UNIVERSITY OF THESSALY

POLYTECHNIC SCHOOL

DEPARTMENT OF MECHANICAL ENGINEERING

DIPLOMA THESIS

Leakage Detection and Optimal Sensor Placement in a water distribution network

By

Dimitris Katsaros

Supervisor

Prof. Costas Papadimitriou

Submitted in Partial Fulfillment

of the requirements for the degree of

Diploma in Mechanical Engineering

The approval of the Diploma Thesis by the Department of Mechanical Engineering of the University of Thessaly does not imply acceptance of the author's opinions. (Law 5343/32, article 202, paragraph 2).

Certified by the members of the Thesis Committee:

First Examiner (Supervisor)
Prof. Costas Papadimitriou
Professor of Structural Dynamics
Department of Mechanical Engineering,
University of Thessaly

Second Examiner
Prof. Dimitrios Valougeorgis
Professor of Analytical and Computational Methods
Department of Mechanical Engineering,
University of Thessaly

Third Examiner
Associate Prof. Georgios Kozanidis
Professor of Industrial Engineering
Department of Mechanical Engineering,
University of Thessaly

Acknowledgments

First and foremost, I would like to express my deep gratitude to my supervisor Professor Costas Papadimitriou for his support and faith during the developing of this work. Without his continuous guidance and constructive criticism I would not be able to perform this task. His perspective of the science of mechanical engineering has been a constant motivational factor and a much needed future asset for the rest of my life.

Furthermore, I should always thank the teaching staff of the Department of Mechanical Engineering of the University of Thessaly not only for the knowledge that I acquired, during my studies, but for morals and responsibilities that abide by the profession of an engineer.

Last and not least, I owe my gratefulness to my family and friends for their perseverance on influencing me to become a better person in every aspect of life. Their unconditional sacrifices and perpetual encouragement has always been a beacon of hope and an instructive lesson to keep moving forward no matter what the obstacles may be.

Abstract

Designing a water distribution network is the cornerstone of this analysis. Given the data of the network we have built a software that simulates the flow and computes pressures in the nodes of the network and flowrates in the pipes of the network. After the completion of the first and particularly major part, our goal was to examine how to detect a leakage, if one occurs, due to a rupture of a pipe. A methodology for locating a leakage and its amount was developed and implemented in software. Our last objective was to compute the Sensitivity Matrices of the system analyzed for flow and pressure and through that determine the optimal sensor placement in order to determine, most accurately, the leakage characteristics. Results obtained from the software were compared with the ones obtained from EPANET. A high accuracy is observed. Our software gives us the flexibility of developing analytical equations for performing sensitivity analyses, thus making more robust the analyses related to optimization problems involved in leakage detections and optimal sensor placement for large scale water distribution networks.

Table of Contents

1. Theoretical Background

1.1 Introduction

For the last thousand years water distribution networks have played an important role for sustaining and improving life conditions. In recent years, as humanity keeps moving forward, this requirement is even bigger and geometrically increases over time and population rise. To illustrate the need for potable water, in modern society, it is necessary to report that the household consumption per capita varies from 28 to 631 liters per day, a factor of 20, according to a research conducted by the International Water Association, in 2016 containing data from 40 countries and 170 cities. This huge contrast in consumption of clean water by citizens of different countries and the constant effort of attaining more, suggests how inextricably linked is water with the living standard. The most common and efficient way of distributing water is through pipeline networks of different sizes. So it is vital to know how to design and calculate all the parameters that are connected with the operation of the network.

To understand the concept of designing we need first to examine its major parts. A pipeline network is consisted of edges and nodes. The edges are responsible for the transportation of liquid while the nodes, which are connected by links, serve as the places where the consumption of that liquid is carried out. Given the fact that we focus on water distribution networks the liquid that is used is obviously water, at a constant temperature of 20 ^{o}C and density of 998.2$\mathrm{kg/m^3}$. We also presume the acceleration of gravity to be equal to 9.807 $\mathrm{m/s^2}$. The main source of water for a network is a number of huge tanks, filled with clean water, which is then delivered to the consumers. There are also many useful components such as valves, pumps, elbows etc., without the use of whom a variety of water networks wouldn't be possible. So it is crystal clear that the complexity of the network varies according to our needs but the main principles of designing and acquiring results remain the same.

As was mentioned above after formulating the network by a set of data (pipe diameters, roughness etc.) we have to mathematically solve it. Getting credible results is of vital importance because parameters such as Flow in each edge and Pressure at each node are the heart of the system and the cornerstone of every analysis. In order to accomplish that, a source code was constructed in MATLAB that depicts and solves every network using a set of non-linear system of equations. To validate the accuracy of the produced results we have used the EPANET software on the same networks. EPANET is a free program used to analyze water distribution networks and it is a very useful tool for performing various simulations. Furthermore for a more convenient and productive use of EPANET, we managed to call it through our developed software and utilize the source code of that program for serving our needs. Networks with different geometries and complexities were created to make full use of the possibilities provided by our model.

After completing the first but the most vital part of the thesis we turned our attention to a highly important problem that is rather a thorn for every network in the world, the detection of leakage. It is crucial to realize that the maintenance service is of tremendous importance, because a well functional network is both beneficial and financially friendly. To make the things easier we based our analysis on the assumption that we have to deal with a leakage, located only at a single edge of the network.

Sensitivity analysis is also part of this thesis. It is quite obvious that flows and pressures depend on the node demands of the system. That is the reason why we will obtain the derivatives for flow and pressure at a given point. That particular point called θ is the demand of a node. The derivatives which constitute the sensitivities will be produced analytically and with finite difference in order to validate the results.

The final part of the current thesis that profoundly relies on the sensitivity matrices, includes an optimal sensor analysis. The sensors that are used are categorized in two forms. Those that are used in an edge and measure the volume flow of the fluid and those which are put at the nodes and measure the pressure. It is very helpful for the maintenance units to know where to place the sensors, given their number, in order to minimize the cost of the sensor equipment and detect the leakage in less time.

It is necessary to mention that in the whole process we considered that water is an uncompressed fluid and the flow is fully developed. Furthermore a model of fixed demands was considered which makes the problem independent of time. Of course in a real water pipeline network, demands vary every minute and are very hard to predict. To conclude, we established a reliable base on which someone can build on and expand this specific analysis in many directions of more complex background.

1.2 Principles of Network Design

The design of a pipe network is based on the behavior of the fluid in internal flows. It is natural to realize that every moving fluid interacts with the surface that is moving on and that has great influence on its motion. In other words, there are losses due to friction. These losses can be categorized in a) frictional losses due to wall shear in pipes, distributed evenly along the length of each pipe and b) minor losses due to the secondary components that constitute the network such as valves, elbows etc. So it is vital to take a closer look at these terms and understand the impact of their meaning.

1.2.1 Frictional Losses

The Pressure difference in a pipe element due to shear losses caused by the wall of the pipe is given by the relation below

$$DP_L = R \cdot Q^x \tag{1.1}$$

in which, DP_L is the pressure loss over the length, R is the resistance coefficient, Q is the volume flow of the fluid and x is an exponent. Depending on the formulation the R changes into various forms. Taking into account the Darcy-Weisbach, the exponent, x=2 and the resistance R is expressed as

$$R = \frac{8 \cdot f \cdot L}{g \cdot \pi^2 \cdot D^2} \tag{1.2}$$

where f is the friction factor, L is the length of the pipe, g is the gravity acceleration and D is the diameter of the pipe. There are additional formulas that for pipe frictional losses with the most known to be that of Hazen-Williams and Chazy-Manning which are mentioned bellow but it is worth to underline that (1.2) is the relation that we are using in this thesis.

Hazen-Williams:

$$R = \frac{K_1 \cdot L}{C^x \cdot D^m} \tag{1.2.1}$$

where K_1 is a constant that equals to 10.59 for S.I and 4.72 for English Units, the exponents x=1.85 and m=4.87 and last the variable C is the Hazen-Williams coefficient which has a dependence only on the pipe roughness. So the type of pipe, along with its time of use play a major role on C.

Chazy-Manning

$$R = \frac{10.29 \cdot n^2 \cdot L}{K_2 \cdot D^{5.33}} \tag{1.2.2}$$

in which n is the Manning roughness coefficient, the exponent x=2 and K_2=1 for S.I and K_2=2.22 for English Units. It is more commonly associated with open flow channels, sewage or drainage systems.

1.2.2 Friction Factor

The friction factor is a very interesting variable that affects mainly the Resistance coefficient, equation (1.2) and needs a lot of attention. It can be obtained by the Moody diagram that relates it with Reynolds number and surface roughness for fully developed flow in a circular pipe. However we are dealing with equations, so we need a reasonably accurate formula that can do the same procedure as Moody diagram. The Swamee-Jain equation is the one that we particularly used in our analysis and is written below

$$f = \frac{1.35}{\left[\ln\left(0.27 \cdot \left(\frac{e}{D}\right) + 5.74 \cdot \left(\frac{1}{Re}\right)^{0.9}\right)\right]^2} \tag{1.3}$$

Where, e is the roughness of the pipe wall, D the diameter and Re the Reynolds number. The equation (1.4) is valid over the ranges $10^{-8} < {}^{e}/_{D} < 10^{-2}$ & $5 \cdot 10^3 < Re < 10^8$. The Reynolds number is expressed as

$$\mathrm{Re} = \frac{\rho \cdot V \cdot D}{\mu} \tag{1.4}$$

in which ρ is the fluid's density , V is the fluid's velocity, D is the diameter of the pipe and μ the viscosity of the fluid. Given the fact that the volume flow in a pipe equals the velocity of the fluid multiplied by the circular surface of the pipe

$$Q = V \cdot A \tag{1.5}$$

and that surface equals

$$A = \frac{\pi \cdot D^2}{2} \tag{1.6}$$

By substituting (1.5), (1.6) and (1.4) to (1.3) we get the final form of the friction factor which is

$$f = \frac{1.35}{\left[\ln\left(0.27 \cdot \left(\frac{e}{D}\right) + 5.74 \cdot \left(\frac{\mu \cdot \pi \cdot D}{4 \cdot \rho \cdot Q}\right)^{0.9}\right)\right]^2} \tag{1.7}$$

It is obvious that f depends on various quantities, $f(\rho, \mu, Q, e, D)$. Of course there are many more approaches to the friction factor with the most known to be that of Colebrook, Hazen-Williams and Manning. The most reliable is Colebrook's equation but the most complex as well. So given the fact that the computational time matters we decided to use Swamee-Jain, (1.7) which show the best behavior towards Colebrook's while Manning's and Hazen-Williams's equations are valid over a limited range of Reynolds numbers.

1.2.2 Minor Losses

The next category of losses, are called Minor Losses, are those that occur due to the existence of a component in some place along the pipe. The relation that will help us to establish our final equation is

$$DP_c = K \cdot \frac{V^2}{2 \cdot g} \tag{1.8}$$

In which, $K = \sum_i k_i$

DP_c is the pressure difference that accounts for the minor losses in a pipe, k_i is a constant that its values vary according to the type of component that exists in the pipe, K is the sum of k_i , in case that there is more than one component in the pipe, V is the velocity of the fluid in the pipe and g is the gravity acceleration. Taking into account the relations (1.5) & (1.6) we come up with the final form of (1.8) which is

$$DP_c = \frac{8 \cdot K}{g \cdot \pi^2 \cdot D^4} \cdot Q^2 \tag{1.9}$$

Where, K is a constant that was analyzed above, g is the acceleration of gravity, D is the diameter of the pipe and finally Q is the volume flow in the pipe.

Summing up, the relations that describe the frictional losses and the minor losses in a pipe are (1.2) and (1.9) respectively. Furthermore it should be highlighted that the frictional factor in (1.2) is the one stated in (1.3) but written in a way to reveal the dependence of f to Q.

1.2.2 Pumps

In the previous sections it was shown that there is a pressure difference along the length of the pipe caused by losses. That may be a problem in big networks because we need large amounts of water to distribute to many consumers. To deal with that obstacle we use pumps, devices that raise the pressure of the fluid in order to overcome the operating pressure of the system and move it at a required flow rate. It is obvious that pumps are part of every modern network and without the use of whom many of them would not be operational.

In the current thesis we focus on pumps with constant power. That means that the pressure provided by the pump is fixed and does not fluctuate. The relation of pressure rise due to the existence of the pump, in meters, is given by the following equation

$$DP_p = \frac{P_u \cdot eff}{Q \cdot \rho \cdot g} \tag{1.10}$$

Where, P_u is the power of the pump, eff is the pump's efficiency, Q is the volume flow, ρ is the fluid's density and finally g is the gravity acceleration. We presume $\gamma = \rho \cdot g$ which is called specific gravity.

2. Network's System of Equations

After making an extensive report on the design principles of the network it is time to analyze how to solve it and calculate the Flow in each and every edge and the Pressure at each and every node. To achieve that we have to formulate our system and examine what methodology we need to use to find the answers we are looking for. So it is crucial to organize our system into key points in order to understand how we cope with such problems. These 3 key points, which are analyzed with details below, are the loops, the pseudo-loops and the equilibrium of mass in the active nodes of the system. Each key point contributes a set of equations and the total combination of these equations formulates the non-linear system that computes the Flow. It goes without saying that the number of equations that are used is equal to the number of pipes that constitute the network(N_p). This means that the number of equations given by the loops (N_l) plus the number given by the Pseudo-loops (N_{pl}) plus the number given by the equilibrium of mass at the active nodes (N_{an}) equals to the number of pipes which is the number of equations needed to compute the Flow in every pipe of the network, $(N_p = N_l + N_{pl} + N_{an})$. After accomplishing the calculation of the flow values we proceed into finding the node Pressures. In this case we have to formulate a linear system of equations which is a simple process and its solution is gives us the pressure values. Let's begin our analysis with the Flows and then move on to the Pressures.

2.1 Computation of Flows

2.1.1 Finding the loops

The finding of the loops plays a major role in order to form the loop set of equations. By definition, a loop is closed route of pipes, in other words if we start from a node we can come back to it by selecting a path of different pipes with no double matches. The way of achieving that is through the use of the function displayed in the Appendix which spots all the possible loops in the network by a repeated procedure. The functions returns a cell array that contains the nodes that constitute all loops and since we know the nodes we can easily find the pipes of the loops because a single pipe is the joint between two nodes. That particular function was modified from its original form in order to fit our needs be incorporated in our code. The details of its origin and use are expressed in the Appendix section. Having completed the task of spotting all the possible loops we must find a way to understand the actual number of the loops existing in the network. That can be determined by the fact that $N_p = N_l + N_{pl} + N_{an} \rightarrow N_l = N_p - N_{pl} - N_{an}$. The Npl and Nan are known as shown in the next key points along with the number of pipes Np. Furthermore is would be nice to clarify that, in huge nets the size of possible loops may be extremely big, so having found the number loops that we have to keep, in order to formulate the loop equations, it would be counterproductive to use oversized loops. That is the reason why we prefer the smallest available.

2.1.2 Loops - 1st Key Point

We start the formation of the nonlinear system by examining the pressure difference in each pipe of the network. The equations that provide us with that information are based on the principles of network design that we discussed in the previous chapter. So the relations that we will need are (1.2), (1.8) and (1.10) while the friction factor in (1.7) is obtained from Swamee-Jain. Moreover we are aware of the fact that the pressure difference in a closed route of pipes or loop is equal to zero. By combining these relations from chapter 1 we will create a set of equations for the loops of the network. Assuming that we have calculated the number of loops in the network and the nodes that constitute each loop we use the following set of equations based on the equilibrium of energy, each of which represents a single loop.

$$\sum_i (\pm)_i \cdot \left(\frac{8 \cdot f_i \cdot L_i}{g \cdot \pi^2 \cdot {D_i}^5} \cdot {Q_i}^2 + \frac{8 \cdot K_i}{g \cdot \pi^2 \cdot {D_i}^4} \cdot {Q_i}^2 - \frac{P_{ui} \cdot eff_i}{Q_i \cdot \gamma}\right) = 0 \qquad (2.1)$$

where, i corresponds to the pipes of each loop. It must be highlighted that (2.1) is a loop equation. Each loop has its own equation so the relation (2.1) is formulated for different pipes i, that constitute the examined loop. So to be exact, the first term comes from (1.2) and is the frictional losses in each pipe of the loop, the second term symbolizes the minor losses of the pipes that constitute the loop (1.8) and the last term is the pressure bust from the pump (1.10), that's why it has a minus while R and DP_c are positive. It has to be noted that the last term is not zero only if a pump exists in the pipe i, the second if a component or more exists in the pipe i while the first term is positive for every single pipe i with no exceptions. All terms of (2.1) are deeply analyzed in the previous chapter. Last thing that is of great essence is the signs $(\pm)$ of pipe i in the loop. The user can determine a positive direction for the loop examined each time. We deemed as positive that of the input matrix containing the node connections which are the pipes. If the original direction of the pipe, given by the matrix with the node connection, concur with that of the node sequence provided by the loop function then we consider the (+) symbol for the examined pipe of the loop, else we use the (-). In Figure 1 is depicted a single loop and the signs of the pipes to make clear the procedure.

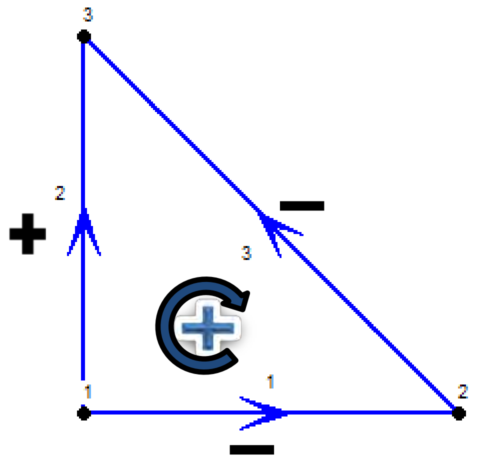

Figure 1: A simple loop of three pipes and the positive clockwise direction

2.1.3 Pseudo-loops - 2st Key Point

After completing the Loop section we focus on the so called Pseudo-loops. With the term Pseudo-loop we mean route of pipes connecting two fixed-grade nodes. To be more precise we mean nodes with constant and unchangeable pressure difference with one another, namely two nodes that contain a Tank. That idea is also based on the equilibrium of energy with the relation for each pseudo-loop to be

$$\sum_{i}(\pm)_i \cdot \left(\frac{8 \cdot f_i \cdot L_i}{g \cdot \pi^2 \cdot D_i{}^5} \cdot Q_i{}^2 + \frac{8 \cdot K_i}{g \cdot \pi^2 \cdot D_i{}^4} \cdot Q_i{}^2 - \frac{P_{ui} \cdot eff_i}{Q_i \cdot \gamma}\right) + \mathrm{DH}_t = 0 \qquad (2.2)$$

As someone can see the equation (2.2) is almost the same with (2.1). The only difference apart from the i pipes that are involved in (2.2) is the last term DH_t which is the value of the head difference between the tank nodes in every pseudo-loop equation. Head is a pressure term clarified in section 2.2. The signs are also determined by the user who assumes the positive and the negative directions. The number of the pseudo-loops equations used in the non-linear system is equal to the number of the Tank nodes minus one $(N_t - 1)$ while the i in (2.2) symbolizes the pipes that form the pseudo-loop.

2.1.4 Equilibrium of mass - 3st Key Point

The last but not least key point has to do with the continuity set of equations which are based on a simple concept. The amount of water that gets in a node is equal to the amount of water that gets out of the node minus the amount of water that stays in the node, in other words the demand of the node or consumption. That principle applies only for the active nodes of the system which includes all the nodes except of those that contain a Tank. The Tank nodes have zero demands so they do not play an important role for this particular set of equations and that is why they get eliminated. We make the assumption that when the flow gets in the node is considered positive while when it leaves the node is deemed negative. To make things clear the equation that is written bellow explains the mentioned above considerations.

$$S_{ji} \cdot Q_i - de_j = 0 \tag{2.3}$$

i=pipe number

j=active node number

where, S_{ji} is a sign matrix that constitutes of $(\pm)\ 1\ \&\ 0$ and represents whether the pipe flow (Q_i) enters the j node (+) or leaves from it (-) while the other elements of S are zero, Q_i is the Flow of pipe i and de_j is the water demand of j node.

In conclusion it would be nice to specify one more time that each key point represents a set of equations that altogether form the final non-linear algebraic system of equations that are solved in our software.

2.1.5 Flow Equations in Matrix form

We will begin with the loops by assuming,

$$Al = (\pm)_i \frac{8 \cdot f_i \cdot L_i}{g \cdot \pi^2 \cdot {D_i}^5} \tag{2.4}$$

$$Bl = (\pm)_i \frac{8 \cdot K_i}{g \cdot \pi^2 \cdot {D_i}^4} \tag{2.5}$$

$$El = (\pm)_i \frac{Pu_i \cdot eff_i}{\gamma} \quad (2.6)$$

In which i corresponds to the pipe of each loop. These matrices have N_l rows and N_p columns. After formulating the matrices (2.4), (2.5) and (2.6) the relations for the loops transforms into,

$$Al \cdot {Q_i}^2 \ + Bl \cdot {Q_i}^2 \ - El \cdot \ \frac{1}{Q_i} = 0 \quad (2.7)$$

We move on to the pseudo-loops by assuming,

$$Apl = (\pm)_i \frac{8 \cdot f_i \cdot L_i}{g \cdot \pi^2 \cdot {D_i}^5} \quad (2.8)$$

$$Bpl = (\pm)_i \frac{8 \cdot K_i}{g \cdot \pi^2 \cdot {D_i}^4} \quad (2.9)$$

$$Epl = (\pm)_i \frac{Pu_i \cdot eff_i}{\gamma} \quad (2.10)$$

In which i corresponds to the pipe of each pseudo-loop. These matrices have N_{pl} rows and N_p columns. From the equation (2.8),(2.9) and (2.10) we get,

$$Al \cdot {Q_i}^2 \ + Bl \cdot {Q_i}^2 \ - El \cdot \ \frac{1}{Q_i} + \mathrm{DH}_t = 0 \quad (2.11)$$

Finally the (2.3) the equilibrium of mass at the active nodes in a matrix form,

$$S \cdot \mathrm{Q}_i - \ de = 0 \quad (2.12)$$

The relation (2.7), (2.11) and (2.12) constitute the non-linear system for flows in matrix form. It would be efficient utilize the capabilities of the matrices by using formulating the system as shown in this subsection.

2.2 Computation of Pressures

Having concluded the Flow section it is vital to calculate the Pressure at the nodes as well. It is an easy procedure that is made up of 3 steps.

1st step

We will use equation (11) for every single pipe, so the formulation is as follows,

$$DH_i = \left| \frac{8 \cdot f_i \cdot L_i}{g \cdot \pi^2 \cdot {D_i}^5} \cdot {Q_i}^2 \; + \; \frac{8 \cdot K_i}{g \cdot \pi^2 \cdot {D_i}^4} \cdot {Q_i}^2 \; - \; \frac{P_{ui} \cdot eff_i}{Q_i \cdot \gamma} \right| \tag{2.4}$$

i=pipe number

where, every term again is known from the previous section and DH_i is the Head difference in each and every pipe. DH_i is always positive even if a pump exists in a pipe because of the use of the absolute value. The Head is also a pressure term that equals the pressure of the node plus the elevation of the node, usually distance from the ground. (See relation (2.6))

2nd step

The second step is based on a linear system to calculate the Head of every node. The linear equation that we are looking for is

$$A_{ij} \cdot \mathrm{H}_j = DH_j \tag{2.5}$$

j=node number

i=pipe number

in which, H is the Head of the nodes, A is a sign matrix with $(\pm)$1 and 0, with (+1) we symbolize the starting node of the pipe i and with (-1) the ending node of the pipe i. DH is the Head difference in every pipe i. It must be noted that the Head in Tank Nodes is constant and equals the water level in the Tank plus the elevation of the node, that simplifies our system even more because the H_j of the Tanks are known.

3rd step

Last we compute the pressure with a simple subtraction since the Heads are known. So,

$$\mathrm{P}_j = H_j - El_j \tag{2.6}$$

j=node number

where, P is the node Pressure, H the node Head and El the elevation of every node.

That is the end of the equation section. Using the constructed code we have managed to compute the Flows in the edges and the Pressures at the Nodes. Other useful information can be also obtained such as the pipe's Velocity, the Reynold's number, the friction factor etc. It must be highlighted that all units are in S.I.

2.3 Illustration of the Equations

The design of the Network is the starting point of every analysis on pipeline networks. Thus it would be nice to illustrate the use of the equations analyzed above by presenting a simple example in order to avoid any misinterpretations.

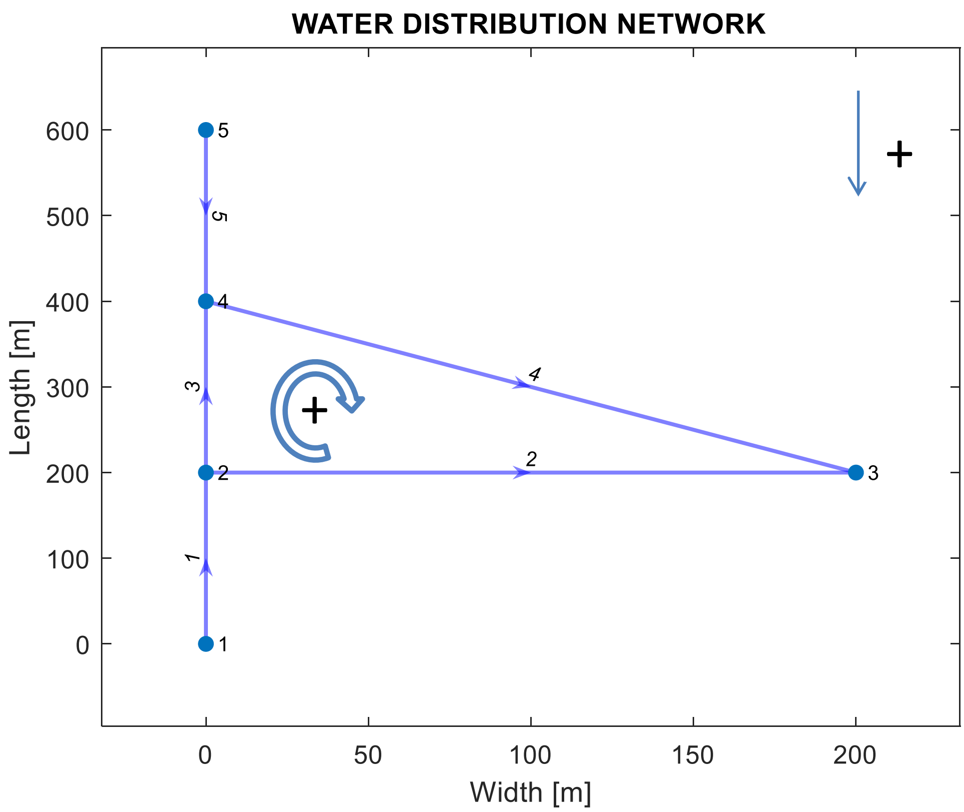

Figure 2 : Illustration Network, simplest case, Tank Node: 1 & 5

We will formulate the nonlinear system for the flows and then the linear system for the pressures. In figure 2 we can see the length of each pipe along with the connections of pipes and their directions as originally assumed. The analysis will begin with the loops, then the pseudo-loops will follow and finally equilibrium of mass at nodes. The units of each and every parameter of the system are in S.I.

Loops

There is only one loop in the Network so the equation (2.1) is used only once as follows

$$\sum_{i=2,3,4}(\pm)_i \cdot (\frac{8 \cdot f_i \cdot L_i}{g \cdot \pi^2 \cdot {D_i}^5} \cdot {Q_i}^2 \ + \frac{8 \cdot K_i}{g \cdot \pi^2 \cdot {D_i}^4} \cdot {Q_i}^2 \ - \ \frac{Pu_i \cdot eff_i}{Q_i \cdot \gamma}) = 0 \rightarrow$$

$$-(\frac{8 \cdot f_2 \cdot L_2}{g \cdot \pi^2 \cdot {D_2}^5} \cdot {Q_2}^2 \ + \frac{8 \cdot K_2}{g \cdot \pi^2 \cdot {D_2}^4} \cdot {Q_2}^2 \ - \ \frac{Pu_2 \cdot eff_2}{Q_2 \cdot \gamma})$$

$$+\frac{8 \cdot f_3 \cdot L_3}{g \cdot \pi^2 \cdot {D_3}^5} \cdot {Q_3}^2 \ + \frac{8 \cdot K_3}{g \cdot \pi^2 \cdot {D_3}^4} \cdot {Q_3}^2 \ - \ \frac{Pu_3 \cdot eff_3}{Q_3 \cdot \gamma}$$

$$+\frac{8 \cdot f_4 \cdot L_4}{g \cdot \pi^2 \cdot {D_4}^5} \cdot {Q_4}^2 \ + \frac{8 \cdot K_4}{g \cdot \pi^2 \cdot {D_4}^4} \cdot {Q_3}^2 \ - \ \frac{Pu_4 \cdot eff_4}{Q_4 \cdot \gamma} = 0 \ \ (1)$$

In case that there is no extra component or pump in the pipe the second or third term of (2.1) is equal to zero.

Pseudo-loops

We consider that the nodes 1 and 5 are Tank Nodes whose purpose is to supply the network with water. So there is only one pseudo-loop in the Network since the number of equations for the pseudo-loops is equal to (number of Tank Nodes-1). There are 2 possible routes of pipes in order to connect node 1 and 5. The first is 1-3-5 and the second is 1-2-4-5. We always prefer the shortest path in order to gain computational time. Therefore, we make use of the equation (2.2) for the path 1-3-5.

$$\sum_{i=1,3,5}(\pm)_i \cdot (\frac{8 \cdot f_i \cdot L_i}{g \cdot \pi^2 \cdot {D_i}^5} \cdot {Q_i}^2 \ + \frac{8 \cdot K_i}{g \cdot \pi^2 \cdot {D_i}^4} \cdot {Q_i}^2 \ - \ \frac{Pu_i \cdot eff_i}{Q_i \cdot \gamma}) \ + \ \mathrm{DPt} = 0 \rightarrow$$

$$-(\frac{8 \cdot f_1 \cdot L_1}{g \cdot \pi^2 \cdot {D_1}^5} \cdot {Q_1}^2 \ + \frac{8 \cdot K_1}{g \cdot \pi^2 \cdot {D_1}^4} \cdot {Q_2}^2 \ - \ \frac{Pu_1 \cdot eff_1}{Q_1 \cdot \gamma})$$

$$-(\frac{8 \cdot f_3 \cdot L_3}{g \cdot \pi^2 \cdot {D_3}^5} \cdot {Q_3}^2 \ + \frac{8 \cdot K_3}{g \cdot \pi^2 \cdot {D_3}^4} \cdot {Q_3}^2 \ - \ \frac{Pu_3 \cdot eff_3}{Q_3 \cdot \gamma})$$

$$+\frac{8 \cdot f_5 \cdot L_5}{g \cdot \pi^2 \cdot {D_5}^5} \cdot {Q_5}^2 \ + \frac{8 \cdot K_5}{g \cdot \pi^2 \cdot {D_5}^4} \cdot {Q_5}^2 \ - \ \frac{Pu_5 \cdot eff_5}{Q_5 \cdot \gamma} \ + \ \mathrm{DH}_{5-1} = 0 \ \ (2)$$

Where $\mathrm{DH}_t = \mathrm{H}_5 - \mathrm{H}_1$ is the head difference between the Tank Nodes 1 and 5. So far we have formulated 2 equations and we need another 3 since the number of pipes is equal to 5.

Equilibrium of Mass

The last part of the system is the equilibrium of water at active nodes. The number of equations that we can extract from the equilibrium, as stated previously, is equal to the number of nodes-number of Tank Nodes which is equal to 3. That equal to the number of the equations that remain since we have used 2 out of 5 until now. Let's see the relation (2.3)

$$S_{ji} \cdot \mathrm{Q}_i - de_j = 0 \rightarrow$$

$$\begin{bmatrix} 1 & -1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 & 1 \end{bmatrix} \cdot \begin{Bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \\ Q_5 \end{Bmatrix} - \begin{Bmatrix} de_2 \\ de_3 \\ de_4 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 0 \end{Bmatrix} \rightarrow$$

$$Q_1 - Q_2 - Q_3 - de_2 = 0 \quad (3)$$

$$Q_2 - Q_4 - de_3 = 0 \qquad (4)$$

$$Q_3 - Q_4 + Q_5 - de_4 = 0 \quad (5)$$

That is the end of the non-linear system for this specific case, formulated by the (1),(2),(3),(4) and (5). If there was only one Tank Node there would be no pseudo-loop equation and the equilibrium of mass would provide us with 4 equations instead of 3. The concept is the same for the larger networks depicted in the next section of this chapter and any other. Let's proceed with the linear system that computes the Pressure at each Node. We begin with (2.4)

$$DH_i = \left| \frac{8 \cdot f_i \cdot L_i}{g \cdot \pi^2 \cdot {D_i}^5} \cdot {Q_i}^2 + \frac{8 \cdot K_i}{g \cdot \pi^2 \cdot {D_i}^4} \cdot {Q_i}^2 - \frac{Pu_i \cdot eff_i}{Q_i \cdot \gamma} \right| \rightarrow$$

$$\begin{Bmatrix} DH_1 \\ DH_2 \\ DH_3 \\ DH_4 \\ DH_5 \end{Bmatrix} = \begin{Bmatrix} \left| \frac{8 \cdot f_1 \cdot L_1}{g \cdot \pi^2 \cdot {D_1}^5} \cdot {Q_1}^2 + \frac{8 \cdot K_1}{g \cdot \pi^2 \cdot {D_1}^4} \cdot {Q_1}^2 - \frac{Pu_1 \cdot eff_1}{Q_1 \cdot \gamma} \right| \\ \left| \frac{8 \cdot f_2 \cdot L_2}{g \cdot \pi^2 \cdot {D_2}^5} \cdot {Q_2}^2 + \frac{8 \cdot K_2}{g \cdot \pi^2 \cdot {D_2}^4} \cdot {Q_2}^2 - \frac{Pu_2 \cdot eff_2}{Q_2 \cdot \gamma} \right| \\ \vdots \\ \left| \frac{8 \cdot f_5 \cdot L_5}{g \cdot \pi^2 \cdot {D_5}^5} \cdot {Q_5}^2 + \frac{8 \cdot K_5}{g \cdot \pi^2 \cdot {D_5}^4} \cdot {Q_5}^2 - \frac{Pu_5 \cdot eff_5}{Q_5 \cdot \gamma} \right| \end{Bmatrix}$$

After that by utilizing (2.5) and (2.6) we get

$$A_{ij} \cdot (\mathrm{P}_j + El_j) = DH_j \rightarrow$$

$$\begin{bmatrix}1 & -1 & 0 & 0 & 0\\0 & 1 & -1 & 0 & 0\\0 & 1 & 0 & -1 & 0\\0 & 0 & -1 & 1 & 0\\0 & 0 & 0 & -1 & 1\end{bmatrix}\cdot\begin{Bmatrix}\mathrm{P}_1\\\mathrm{P}_2\\\mathrm{P}_3\\\mathrm{P}_4\\\mathrm{P}_5\end{Bmatrix}=\begin{Bmatrix}DH_1\\DH_2\\DH_3\\DH_4\\DH_5\end{Bmatrix}-\begin{bmatrix}1 & -1 & 0 & 0 & 0\\0 & 1 & -1 & 0 & 0\\0 & 1 & 0 & -1 & 0\\0 & 0 & -1 & 1 & 0\\0 & 0 & 0 & -1 & 1\end{bmatrix}\cdot\begin{Bmatrix}El_1\\El_2\\El_3\\El_4\\El_5\end{Bmatrix}$$

And we solve the linear system for P_j. Note that the pressure at Tank Nodes is known, so P_1 and P_5 are constant and equal to the water level in the Tank if we calculate the pressure in meters.

2.4 Networks and Results

Using the previously portrayed relations we managed to deal with various distribution water networks and extract sufficient results. Before depicting some of the networks and present their results for flow and pressure it is important to say a few word on how we imported the data into our system.

2.4.1 Data Matrices

The input data in a system is the most important part because the whole theory of the procedure is based on them. Of course the way we import the data into our code depends on the user's way of thinking, programming methods and skills. In this case we made use of three matrices which will be thoroughly analyzed.

1st Pipes

Each pipe has a start and an end which are actually nodes connected by a pipe. So this is a NP×2 matrix which contains the start node and the end node of each pipe. NP is the number of pipes of the network.

Start Node	End Node

2nd Pipe Information

The second matrix contains all the information concerning pipes. It is a NP×7 matrix that has the following columns.

1st column	2st column	3st column	4st column	5st column	6st column
Number of pipe	Diameter [mm]	Roughness [mm]	Loss Coefficient [Unitless]	Pump's Power [Watt]	Pump's Efficiency [Unitless]

3rd Node Information

The final matrix is about the Node information. It is a NN×7 and its content is the one shown below

1st column	2st column	3st column	4st column	5st column	6st column
Number of node	Coordinates x-direction [m]	Coordinates y-direction [m]	Coordinates z-direction [m]	Node's Demand [Liters]	Tank water Level [m]

- It should be mentioned that the length of each pipe is derived from the coordinates of the connected Nodes via Pipes by the relation,

  $$L = ((x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2)^{0.5} \quad (2.7)$$

These three matrices constitute to the data input given which the results are extracted. It is vital to mention that all the data are turned into S.I units when we implement the equations.

2.4.2 Results

To conclude the design section of the Network we have to present a sample of the networks that were examined. It is worth to state again that each network is different in complexity but all rely on the basic principles that have been analyzed so far. Firstly we will depict a figure of the network in 2-D (x-y directions), then we will introduce the data of every network in the form shown previously, then we will and finally we will display the flow and pressure results.

1st Network – (7 Pipes & 6 Nodes)

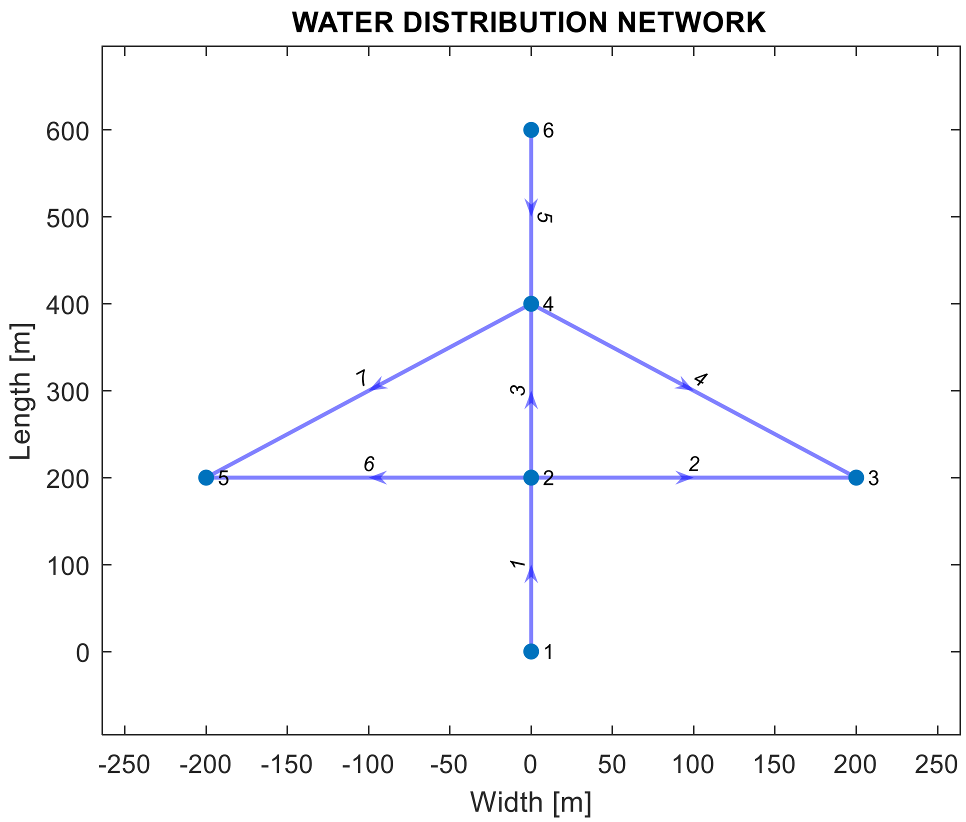

Figure 3 : Water Distribution Network, Water Tanks : Nodes 1 & 6, No Pumps

Now it is time to display the three data matrices:

Table 1 : Pipe Connections for Network 1

Start Node	End Node
1	**2**
2	**3**
2	**4**
4	**3**
6	**4**
2	**5**

4	5

Table 2 : Pipe information for Network 1

Pipe's Number	Diameter	Roughness	Loss Coefficient	Pump's Power	Pump's Efficiency
1	300	0.26	0	0	0
2	200	0.26	0	0	0
3	250	0.26	0	0	0
4	250	0.26	0	0	0
5	300	0.26	0	0	0
6	200	0.26	0	0	0
7	250	0.26	0	0	0
-	[mm]	[mm]	-	[Watt]	-

Table 3 : Node information for Network 1

Node's Number	Coordinates x-direction	Coordinates y-direction	Coordinates z-direction	Node's Demand	Tank water Level
1	0	0	50	0	50
2	200	0	0	10	0
3	200	200	0	12	0
4	400	0	0	15	0
5	200	-200	0	12	0
6	600	0	50	0	50
-	[m]	[m]	[m]	[Liters]	[m]

Finally, the results for Flow and Pressure are presented:

Table 4 : Flow and Velocity Results for Network 1

Pipe's Number	Volumetric Flow	Velocity
1	24.0168	0.3398
2	5.1181	0.1629
3	3.7806	0.0770

4	6.8819	0.1402
5	24.9832	0.3534
6	5.1181	0.1629
7	6.8819	0.1402
-	[Liters/sec]	[m/sec]

Table 5 : Head and Pressure Results for Network 1

Node's Number	Head	Pressure Height	Pressure
1	100	50	489.4625
2	99.9117	99.9117	978.0609
3	99.8759	99.8759	977.9078
4	99.9049	99.9049	977.9936
5	99.8759	99.8759	977.7038
6	100	50	489.4625
-	[m]	[m]	[KPascal]

As someone can see the 1st Network is very simple since it does not have any pump and the loss coefficient in every pipe equals to zero. The second Network that we are going to examine is a little more complicated since it contains all the data parameters and is slightly bigger that the first one.

2nd Network – (14 Pipes & 11 Nodes)

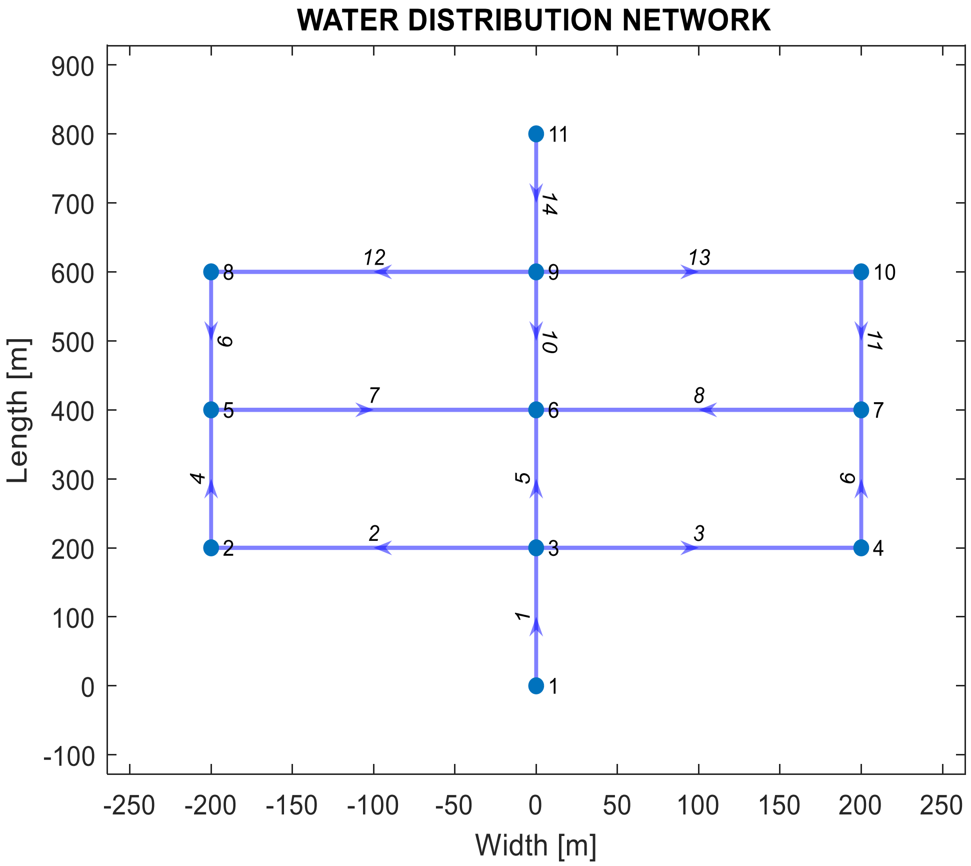

Figure 4 : Water Distribution Network, Water Tanks: Nodes 1 & 11, Pumps: Pipes 1 & 14

We begin with the Data,

Table 6 : Pipe Connections for Network 2

Start Node	End Node
1	3
3	2
3	4
2	5
3	6
4	7
5	6
7	6
8	5

9	6
10	7
9	8
9	10
11	9

Table 7 : Pipe information for Network 2

Pipe's Number	Diameter	Roughness	Loss Coefficient	Pump's Power	Pump's Efficiency
1	300	0.26	10	100000	0.75
2	250	0.26	10	0	0
3	250	0.26	10	0	0
4	250	0.26	10	0	0
5	200	0.26	10	0	0
6	250	0.26	10	0	0
7	200	0.26	10	0	0
8	200	0.26	10	0	0
9	250	0.26	10	0	0
10	200	0.26	10	0	0
11	250	0.26	10	0	0
12	250	0.26	10	0	0
13	250	0.26	10	0	0
14	300	0.26	10	100000	0.75
-	[mm]	[mm]	-	[Watt]	-

Table 8 : Node information for Network 2

Node's Number	Coordinates x-direction	Coordinates y-direction	Coordinates z-direction	Node's Demand	Tank water Level
1	0	0	50	0	5
2	200	-200	50	60	0
3	200	0	55	60	0
4	200	200	60	20	0

5	400	-200	55	60	0
6	400	0	60	200	0
7	400	200	65	20	0
8	600	-200	60	20	0
9	600	0	65	20	0
10	600	200	70	60	0
11	800	0	60	0	10
-	[m]	[m]	[m]	[Liters]	[m]

After formulating the data section we proceed to the results:

Table 9 : Flow and Velocity Results for Network 2

Pipe's Number	Volumetric Flow	Velocity
1	237.1617	3.3552
2	76.2464	1.5533
3	46.9978	0.9574
4	16.2464	0.3310
5	53.9175	1.7162
6	26.9978	0.5500
7	28.4060	0.9042
8	43.2278	1.3776
9	72.1596	1.4700
10	74.3987	2.3682
11	36.2800	0.7391
12	92.1596	1.8775
13	96.2800	1.9614
14	282.8383	4.0013
-	[Liters/sec]	[m/sec]

Table 10 : Head and Pressure Results for Network 2

Node's Number	Head	Pressure Height	Pressure
1	55	5	48.9462

2	70.8774	20.8774	204.3742
3	74.1443	19.1443	187.4082
4	72.8866	12.8866	126.1501
5	70.7192	15.7192	153.8796
6	69.3678	9.3678	91.7042
7	72.4623	7.4623	73.0500
8	73.6490	13.6490	133.6135
9	78.4030	13.4030	131.2052
10	73.2187	3.2187	31.5088
11	70	10	97.8925
-	[m]	[m]	[KPascal]

3rd Network – (52 Pipes & 33 Nodes)

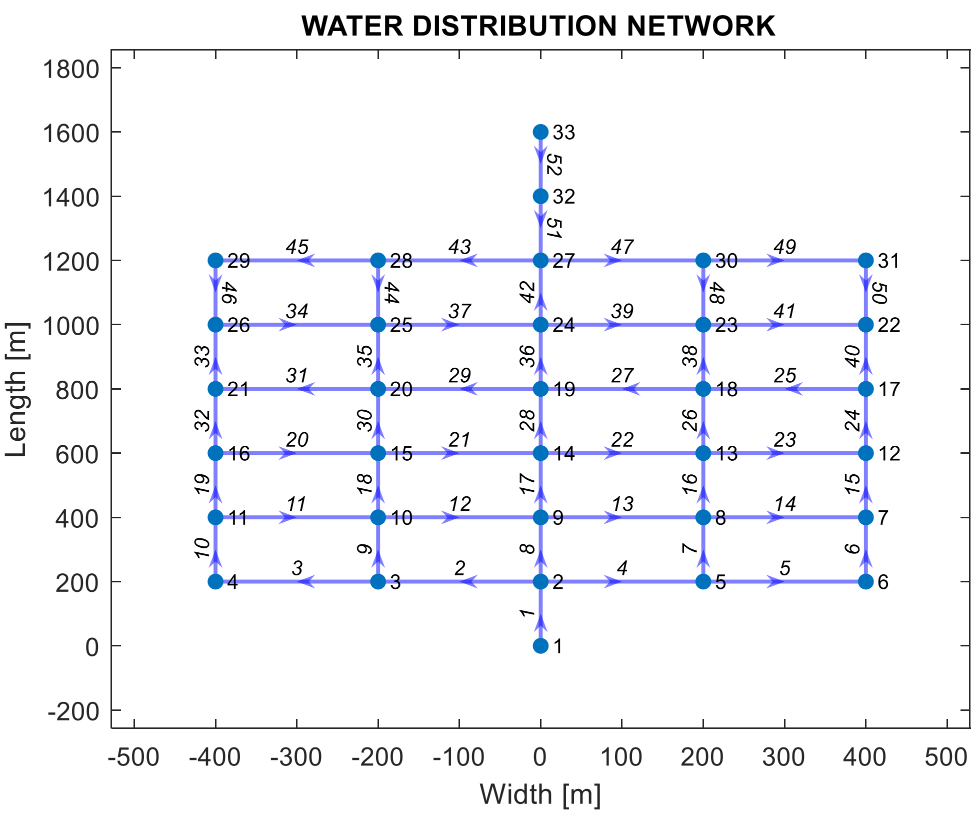

Figure 5 : Network 3, Tank Nodes: 1 & 33, Pumps: Pipes 1 & 52

The Data section is,

Table 11 : Pipe Connections for Network 3

Start Node	End Node
1	3
2	3
3	4
2	5
5	6
6	7
5	8
2	9
3	10
4	11
11	10
10	9
9	8
8	7
7	12
8	13
9	14
10	15
11	16
16	15
15	14
14	13
13	12
12	17
12	18
18	19
14	19
19	20
15	20
20	21
16	21

21	26
26	25
20	25
19	24
25	24
18	23
17	22
23	22
24	27
27	28
28	25
28	29
29	26
27	30
30	31
31	22
32	27
33	32

Table 12 : Pipe information for Network 3

Pipe's Number	Diameter	Roughness	Loss Coefficient	Pump's Power	Pump's Efficiency
1	300	0.26	10	100000	0.75
2	250	0.26	10	0	0
3	250	0.26	10	0	0
4	250	0.26	10	0	0
5	200	0.26	10	0	0
6	250	0.26	10	0	0
7	200	0.26	10	0	0
8	250	0.26	0	0	0
9	250	0.26	0	0	0
10	250	0.26	0	0	0
11	250	0.26	0	0	0

12	250	0.26	0	0	0
13	250	0.26	0	0	0
14	250	0.26	0	0	0
15	250	0.26	0	0	0
16	250	0.26	0	0	0
17	250	0.26	0	0	0
18	250	0.26	0	0	0
19	250	0.26	0	0	0
20	250	0.26	0	0	0
21	250	0.26	0	0	0
22	250	0.26	0	0	0
23	250	0.26	0	0	0
24	250	0.26	0	0	0
25	250	0.26	0	0	0
26	250	0.26	0	0	0
27	250	0.26	0	0	0
28	250	0.26	0	0	0
29	250	0.26	0	0	0
30	250	0.26	0	0	0
31	250	0.26	0	0	0
32	250	0.26	0	0	0
33	250	0.26	0	0	0
34	250	0.26	0	0	0
35	250	0.26	0	0	0
36	250	0.26	0	0	0
37	250	0.26	0	0	0
38	250	0.26	0	0	0
39	250	0.26	0	0	0
40	250	0.26	0	0	0
41	250	0.26	0	0	0
42	250	0.26	0	0	0
43	250	0.26	0	0	0
44	250	0.26	0	0	0

45	250	0.26	0	0	0
46	250	0.26	0	0	0
47	250	0.26	0	0	0
48	250	0.26	0	0	0
49	250	0.26	0	0	0
50	250	0.26	0	0	0
51	250	0.26	0	0	0
52	250	0.26	0	100000	0.75
-	[mm]	[mm]	-	[Watt]	-

Table 13 : Node information for Network 3

Node's Number	Coordinates x-direction	Coordinates y-direction	Coordinates z-direction	Node's Demand	Tank water Level
1	0	0	0	0	50
2	200	0	0	10	0
3	200	-200	0	10	0
4	200	-400	0	10	0
5	200	200	0	10	0
6	200	400	0	10	0
7	400	400	0	10	0
8	400	200	0	10	0
9	400	0	0	10	0
10	400	-200	0	10	0
11	400	-400	0	10	0
12	600	400	0	10	0
13	600	200	0	10	0
14	600	0	0	10	0
15	600	-200	0	10	0
16	600	-400	0	10	0
17	800	400	0	10	0
18	800	200	0	10	0
19	800	0	0	10	0
20	800	-200	0	10	0

21	800	-400	0	10	0
22	1000	400	0	10	0
23	1000	200	0	10	0
24	1000	0	0	10	0
25	1000	-200	0	10	0
26	1000	-400	0	10	0
27	1200	0	0	10	0
28	1200	-200	0	10	0
29	1200	-400	0	10	0
30	1200	200	0	10	0
31	1200	400	0	10	0
32	1400	0	0	10	0
33	1600	0	0	0	0
-	[m]	[m]	[m]	[Liters]	[m]

After formulating the data section we proceed to the results:

Table 14 : Flow and Velocity Results for Network 3

Pipe's Number	Volumetric Flow	Velocity
1	162.2710	3.3058
2	49.8915	1.0164
3	19.8346	0.4041
4	49.8915	1.0164
5	19.8346	0.4041
6	9.8346	0.2003
7	20.0569	0.4086
8	52.4880	1.0693
9	20.0569	0.4086
10	9.8346	0.2003
11	9.4000	0.1915
12	12.2897	0.2504
13	12.2897	0.2504
14	9.4000	0.1915

15	9.2346	0.1881
16	12.9466	0.2637
17	17.9086	0.3648
18	12.9466	0.2637
19	9.2346	0.1881
20	3.2786	0.0668
21	2.7475	0.0560
22	2.7475	0.0560
23	3.2786	0.0668
24	2.5132	0.0512
25	3.3552	0.0684
26	2.4155	0.0492
27	2.7492	0.0560
28	2.4136	0.0492
29	2.7492	0.0560
30	2.4155	0.0492
31	3.3552	0.0684
32	2.5132	0.0512
33	4.1316	0.0842
34	7.7873	0.1586
35	8.1905	0.1669
36	13.0848	0.2666
37	10.3222	0.2103
38	8.1905	0.1669
39	10.3222	0.2103
40	4.1316	0.0842
41	7.7873	0.1586
42	43.7293	0.8908
43	41.9999	0.8556
44	15.6555	0.3189
45	16.3443	0.3330
46	6.3443	0.1292
47	41.9999	0.8556

48	15.6555	0.3189
49	16.3443	0.3330
50	6.3443	0.1292
51	137.7290	2.8058
52	147.7290	3.0095
-	[Liters/sec]	[m/sec]

Table 15 : Head and Pressure Results for Network 3

Node's Number	Head	Pressure Height	Pressure
1	50	50	489.4625
2	88.1782	88.1782	863.1988
3	87.2906	87.2906	854.5091
4	87.1410	87.1410	853.0452
5	87.2906	87.2906	854.5091
6	87.1410	87.1410	853.0452
7	87.1013	87.1013	852.6560
8	87.1378	87.1378	853.0138
9	87.1982	87.1982	853.6048
10	87.1378	87.1378	853.0138
11	87.1013	87.1013	852.6560
12	87.0659	87.0659	852.3099
13	87.0712	87.0712	852.3619
14	87.0751	87.0751	852.3999
15	87.0712	87.0712	852.3619
16	87.0659	87.0659	852.3099
17	87.0626	87.0626	852.2775
18	87.0681	87.0681	852.3317
19	87.0720	87.0720	852.3697
20	87.0681	87.0681	852.3317
21	87.0626	87.0626	852.2775
22	87.0707	87.0707	852.3565
23	87.0964	87.0964	852.6087
24	87.1400	87.1400	853.0348

25	87.0964	87.0964	852.6087
26	87.0707	87.0707	852.3665
27	87.8266	87.8266	859.7564
28	87.1918	87.1918	853.5421
29	87.0883	87.0883	852.5293
30	87.1918	87.1918	853.5421
31	87.0883	87.0883	852.5293
32	94.3584	94.3584	923.6978
33	50	50	489.4625
-	[m]	[m]	[KPascal]

2.5 EPANET

EPANET is the software that was used to validate the results for our analysis. It is free software (see Appendix) with many capabilities on various aspects of water distribution networks. The version that was used is the 2.00.14 but there is even more advanced. For a basic set of input data such as the geometry, node demands, pipe's characteristic, friction factor settings etc. this computer program performs hydraulic simulations to determine the flow in each pipe and the pressure at every node. To be more exact on some basic inputs, we made use of Darcy's friction factor equation, the pumps were of constant power and the system is not time dependent on demands. It is obvious that a partial use of EPANET's capabilities was made due to the fact that our analysis does not extend to time variance.

2.5.1 Networks and Results

In the figures that follow it is clear that all the networks that were shown previously were also replicated in EPANET'S platform as well. Given that, we extracted the Results that EPANET gave to us and displayed them below.

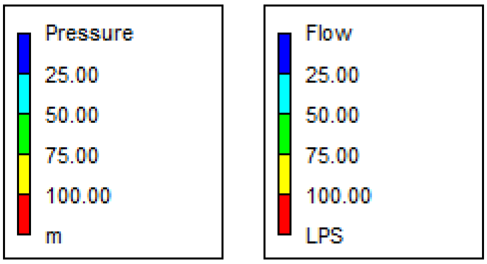

Figure 6 : The Color bars are value indicators for the EPANET Networks

1st Network – (7 Pipes & 6 Nodes)

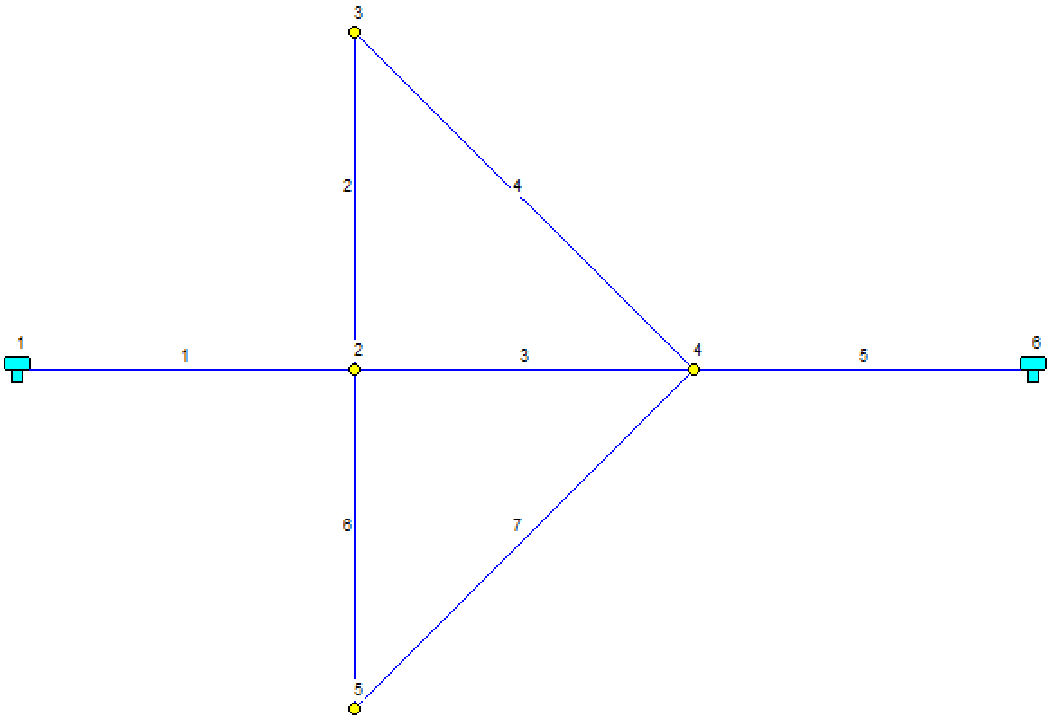

Figure 7 : Network 1 in EPANET

The results for Flow and Pressure in EPANET are presented:

Table 16 : Flow and Velocity Results for Network 1 from EPANET

Pipe's Number	Volumetric Flow	Velocity
1	24.0158	0.3398
2	5.1181	0.1629
3	3.7775	0.0770
4	6.8819	0.1402
5	24.9842	0.3534

6	5.1181	0.1629
7	6.8819	0.1402
-	[Liters/sec]	[m/sec]

Table 17 : Head and Pressure Results for Network 1 from EPANET

Node's Number	Head	Pressure Height	Pressure
1	100	50	489.4625
2	99.9117	99.9117	978.0609
3	99.8757	99.8757	977.9018
4	99.9048	99.9048	977.0028
5	99.8757	99.8757	977.7018
6	100	50	489.4625
-	[m]	[m]	[KPascal]

2nd Network – (14 Pipes & 11 Nodes)

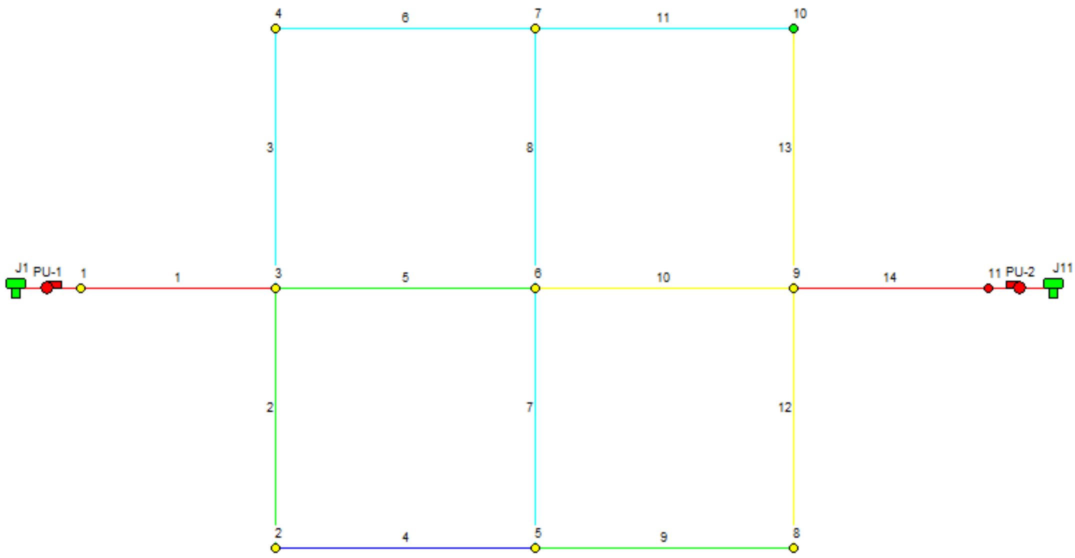

Figure 8 : Network 2 in EPANET

And the results are,

Table 18 : Flow and Velocity Results for Network 2 from EPANET

Pipe's Number	Volumetric Flow	Velocity
1	237.1449	3.3549
2	76.2399	1.5531
3	46.9897	0.9573
4	16.2399	0.3308
5	53.9153	1.7162
6	26.9897	0.5498
7	28.4035	0.9041
8	43.2753	1.3775
9	72.1636	1.4701
10	74.4059	2.3684
11	36.2857	0.7392
12	92.1636	1.8775
13	96.2857	1.9615
14	282.8551	4.0016
-	[Liters/sec]	[m/sec]

Table 19 : Head and Velocity Results for Network 2 from EPANET

Node's Number	Head	Pressure Height	Pressure
1	55	5	48.9462
2	70.8879	20.8879	204.4789
3	74.1531	19.1531	187.4963
4	72.8961	12.8961	126.2444
5	70.7298	15.7298	153.9845
6	69.3786	9.3786	91.8104
7	72.4720	7.4720	73.1460
8	73.6589	13.6589	133.7117
9	78.4114	13.4114	131.0782
10	73.2287	3.2287	31.6069
11	70	10	97.8925
-	[m]	[m]	[KPascal]

3rd Network – (52 Pipes & 33 Nodes)

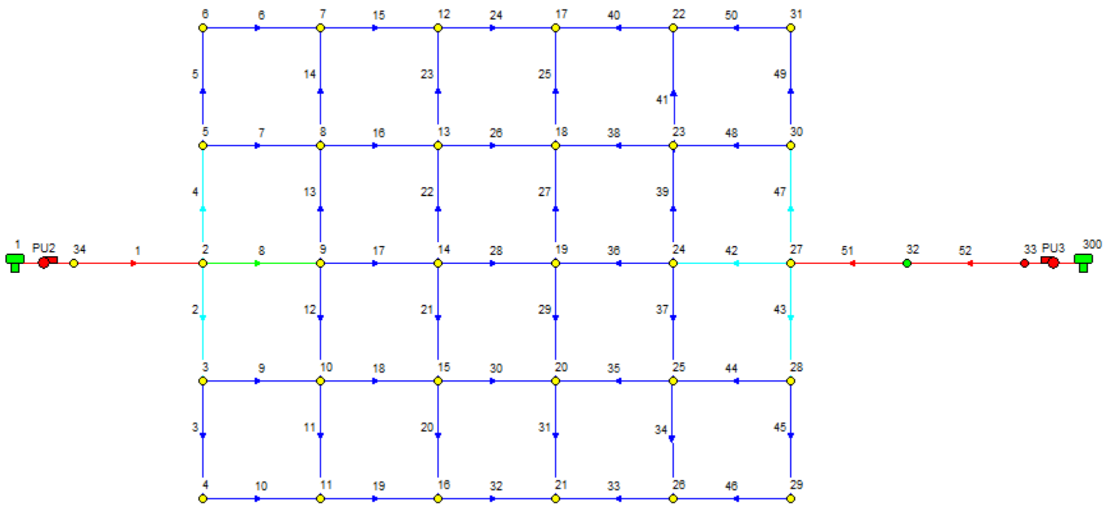

Figure 9 : Network 3 in EPANET

Table 20 : Flow and Velocity Results for Network 3 from EPANET

Pipe’s Number	Volumetric Flow	Velocity
1	162.2713	3.3057
2	49.8909	1.0164
3	19.8341	0.4041
4	49.8908	1.0164
5	19.8339	0.4041
6	9.8339	0.2003
7	20.0568	0.4086
8	52.4897	1.0693
9	20.0568	0.4086
10	9.8341	0.2003
11	9.3993	0.1915
12	12.2894	0.2504
13	12.2885	0.2503
14	9.3985	0.1915
15	9.2324	0.1881
16	12.9468	0.2637
17	17.9118	0.3649

18	12.9469	0.2638
19	9.2334	0.1881
20	3.2775	0.0668
21	2.7433	0.0559
22	2.7585	0.0562
23	3.2956	0.0671
24	2.5281	0.0515
25	3.3139	0.0681
26	2.4097	0.0491
27	2.7431	0.0559
28	2.4100	0.0491
29	2.7543	0.0561
30	2.4127	0.0492
31	3.3570	0.0684
32	2.5109	0.0512
33	4.1320	0.0842
34	7.7874	0.1586
35	8.1900	0.1668
36	13.0874	0.2666
37	10.3226	0.2103
38	8.1911	0.1669
39	10.3203	0.2102
40	4.1281	0.0841
41	7.7845	0.1586
42	43.7303	0.8909
43	41.9994	0.8556
44	15.6548	0.3189
45	16.3447	0.3330
46	6.3447	0.1293
47	41.9990	0.8556
48	15.6553	0.3189
49	16.3436	0.3329
50	6.3436	0.1292

51	137.7287	2.8058
52	147.7287	3.0095
-	[Liters/sec]	[m/sec]

Table 21 : Head and Pressure Results for Network 3 from EPANET

Node's Number	Head	Pressure Height	Pressure
1	50	50	489.4625
2	88.1786	88.1786	863.2109
3	87.2904	87.2904	854.5161
4	87.1407	87.1407	853.0506
5	87.2904	87.2904	854.5161
6	87.1407	87.1407	853.0506
7	87.1018	87.1018	852.6698
8	87.1375	87.1375	853.0193
9	87.1979	87.1979	853.6105
10	87.1375	87.1375	853.0193
11	87.1008	87.1008	852.6600
12	87.0654	87.0654	852.3134
13	87.0707	87.0707	852.3653
14	87.0746	87.0746	852.4035
15	87.0707	87.0707	852.3653
16	87.0654	87.0654	852.3134
17	87.0621	87.0621	852.2811
18	87.0676	87.0676	852.3350
19	87.0715	87.0715	852.3732
20	87.0676	87.0676	852.3350
21	87.0621	87.0621	852.2811
22	87.0702	87.0702	852.3604
23	87.0960	87.0960	852.6130
24	87.1396	87.1396	853.0398
25	87.0960	87.0960	852.6130
26	87.0702	87.0702	852.3604
27	87.8267	87.8267	859.7661

28	87.1915	87.1915	853.5479
29	87.0879	87.0879	852.5337
30	87.1915	87.1915	853.5479
31	87.0879	87.0879	852.5337
32	94.3587	94.3587	923.7101
33	50	50	489.4625
-	[m]	[m]	[KPascal]

2.6 Comparison of Developed Software-EPANET Results

The use of 2 different approaches to extract the needed results for the current study reveals the necessity of valid results. If we pay attention to the result matrices for both the developed software and EPANET we can figure out that they almost perfectly match for flow and pressure. Consequently we are in position to believe that the code constructed is undoubtedly reliable and allow us to proceed.

3. Leakage Detection

After successfully completing the design part of the network, our next objective is to find a way to detect a leakage in some part of the network. Water belongs to the resources that are vital for life. That is why, it is crucial to utilize every single drop and not waste it on account of technical errors. Unfortunately, every man-made system has its flaws and a certain operational duration in time. In the case of water distribution network, the leakage signs the end of a pipe and its immediate replacement or at least a temporary repair, due to cost reasons.

3.1 Basic Principles for Leakage

In order to cope with this problem we have to establish some basic principles for our model. The leakage in this system is nothing more than a demand located in an arbitrary pipe somewhere in the network and that is why is simulated as such. Of course there is no limitation in the number of pipes that present leakage. It is worth to underline again that, by saying demand, we mean a constant consumption of water that an active node requires for its needs. The question that arises is how to identify the location of the leakage (i.e the pipe k that leaks) and the amount of leakage denoted by θ. The answer lies with the optimization field since this is an optimization problem. The next equation is the one used in our analysis and is the one that we wish to minimize in order to obtain k & θ.

$$f(\underline{k},\underline{\theta}) = (L_P \cdot \underline{P}(\underline{k},\underline{\theta}) - \underline{P}_{meas}) \cdot (\underline{P}(\underline{k},\underline{\theta}) - \underline{P}_{meas})^T + (L_Q \cdot \left|\underline{Q}(\underline{k},\underline{\theta})\right| - \left|\underline{Q}_{meas}\right|) \cdot (\left|\underline{Q}(\underline{k},\underline{\theta})\right| - \left|\underline{Q}_{meas}\right|)^T \quad (3.1)$$

Every term of (3.1) will be explained thoroughly in the subsections that follow. The only thing that we should have in mind is that we are interested to the values & θ . In case of multiple leakages k & θ are vectors.

3.1.1 Measurements

The terms $\underline{Q}_{meas}$ and $\underline{P}_{meas}$ are vectors that contain the measurements of flow and pressure in certain pipes and active nodes of the network with the leakage. These measurements are provided as data of the network which needs to be fixed and can be acquired with various techniques like sensor placement. Given the fact that we are not dealing with actual networks the measurements must be obtained by us as explicated below. The way of picking the measurements and form Q_{meas} and P_{meas} is based on the optimal sensor theory which give us the information on where to place a certain number of sensors in order to spot the leakage accurately and with the minimum required equipment since the sensors raise the cost. As we can understand, in real networks the problem gets even more complicated because the demands are not constant due to the fact that they are time dependent.

3.1.2 Simulation of Leakage and Measurements

Since there are no measurements provided we must find a way to run the analysis by producing similar nominal values. This can be achieved by making some speculations about the leakage and then rerun the program according to the new data. We assume that the leakage is a node demand and is simulated by adding a new node in the middle of a pipe we choose. Having considered the pipe of the leakage along with its value, in liters/sec, we put the extra node in the center of that particular pipe. We have previously stated that a pipe is the connection between two neighboring nodes so we 'break' that specific pipe into two sections of equal length and characteristics. Accordingly the newly added node placed in the middle of the pipe has the coordinates and the elevation of the point that is set upon. As we can understand the network changes since we create an extra pipe, by breaking an 'old' one into two and an extra active node, with demand equal to the value of the speculated leakage. After performing all these steps we rerun the analysis and compute the flows and pressures for the reformed network. The values that will be obtained by the program are the measurements we are looking for. It must be clarified that the since the network with the leakage bares an additional pipe and node the results will contain an extra value for flow for the new pipe and an extra value for pressure for the 'leakage' node. These values are eliminated from the measurements because they cannot be used at the original network.

3.1.3 Minimization Process

After acquiring the measurements we should say a few words for the $\underline{Q}(\underline{k},\underline{\theta})$ & $\underline{P}(\underline{k},\underline{\theta})$. The aim of the minimization process is to spot the whereabouts of the leakage k along with its value θ. As we can see the terms of flow and pressure depend on k and θ and that is why their values are affected immensely by these two factors. So our aim is to find the k and θ for which, $f(\underline{k},\underline{\theta})$ is equal to zero or $L_P \cdot \underline{P}(\underline{k},\underline{\theta}) \sim \underline{P}_{meas}$ and $L_Q \cdot \underline{Q}(\underline{k},\underline{\theta}) \sim \underline{Q}_{meas}$. It is obvious that L_P and L_Q are matrices containing of zeros and ones, indicating the locations of the measurements $\underline{P}_{meas}$ and $\underline{Q}_{meas}$. In order to accomplish that an optimization algorithm is used to minimize the objective function $f(\underline{k},\underline{\theta})$. This function requires the solution of the system of network flow and pressure equations developed in chapter 2. It is a repeated procedure for different values of k and θ until it comes up with the optimal solution that minimizes $f(\underline{k},\underline{\theta})$.

We will continue our analysis on leakage detection by using the previously exhibited networks. For the first network we will assume a leakage of 5ltrs/sec in pipe 3 and in the second network we will assume a leakage of 5ltrs/sec in pipe 6. It is the same procedure for the network 3 in which we assume leakage 25ltrs/sec in pipe 33 but it is pointless to display more results since the model is precise as shown before and as displayed bellow.

3.2 Measurement Result

3.2.1 Source Code

1st Network with leakage –(8 Pipes & 7 Nodes)

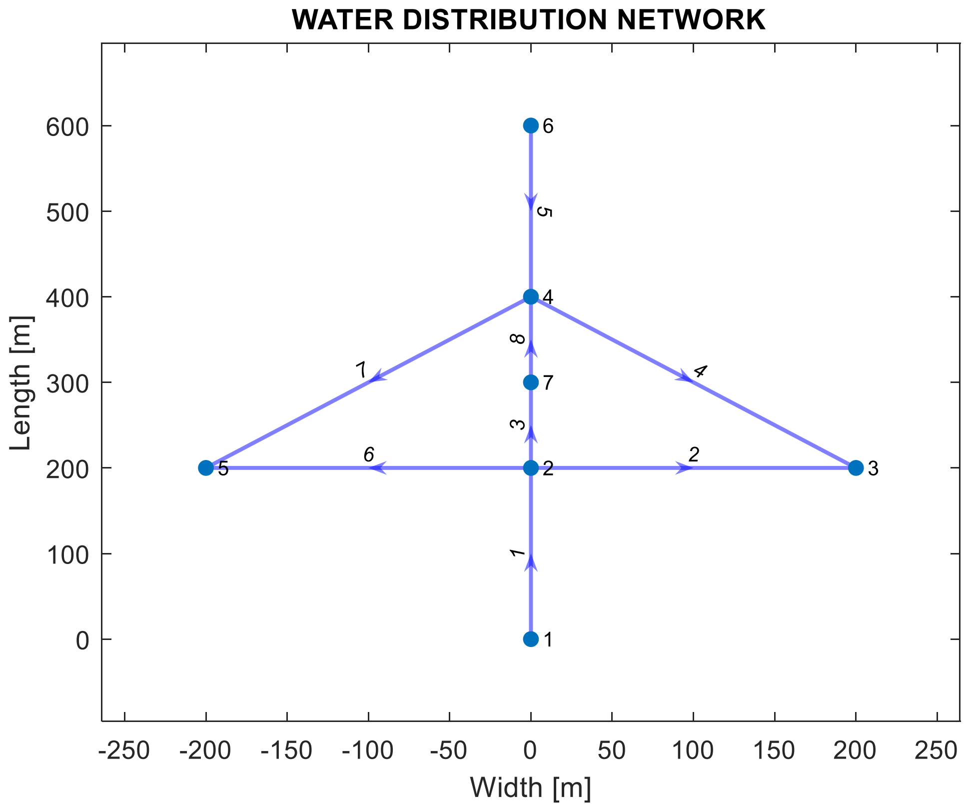

Figure 10 : Network 1 with a leakage

Given the fact that we have added a node and we have broken pipe into two sections, we have a change in our data. That change differentiates the network from its original form. The new results are the measurement that we need for the optimization function. Let's move on to the new data of the Network.

Table 22 : Pipe Connections for Network 1 with leakage

Start Node	End Node
1	2
2	3
2	7
4	3
6	4
2	5
4	5
7	4

Table 23 : Pipe information for Network 1 with leakage

Pipe's Number	Diameter	Roughness	Loss Coefficient	Pump's Power	Pump's Efficiency
1	300	0.26	0	0	0
2	200	0.26	0	0	0
3	250	0.26	0	0	0
4	250	0.26	0	0	0
5	300	0.26	0	0	0
6	200	0.26	0	0	0
7	250	0.26	0	0	0
8	250	0.26	0	0	0
-	[mm]	[mm]	-	[Watt]	-

Table 24 : Node information for Network 1 with leakage

Node's Number	Coordinates x-direction	Coordinates y-direction	Coordinates z-direction	Node's Demand	Tank water Level
1	0	0	50	0	50
2	200	0	0	10	0
3	200	200	0	12	0
4	400	0	0	15	0
5	200	-200	0	12	0
6	600	0	50	0	50

7	300	0	0	5	0
-	[m]	[m]	[m]	[Liters]	[m]

And the new results,

Table 25 : Flow and Velocity Results for Network 1 with leakage

Pipe's Number	Volumetric Flow	Velocity
1	26.4551	0.3743
2	5.1958	0.1654
3	6.0635	0.1235
4	6.8042	0.1386
5	27.5449	0.3897
6	5.1958	0.1654
7	6.8042	0.1386
8	1.0635	0.0217
-	[Liters/sec]	[m/sec]

Table 26 : Head and Pressure Results for Network 1 with leakage

Node's Number	Head	Pressure Height	Pressure
1	100	50	489.4625
2	99.8939	99.9839	977.8864
3	99.8570	99.8570	977.8033
4	99.8854	99.8854	977.5253
5	99.8570	99.8570	977.8033
6	100	50	489.4625
7	99.8858	99.8858	977.8069
-	[m]	[m]	[KPascal]

2nd Network with leakage – (15 Pipes & 12 Nodes)

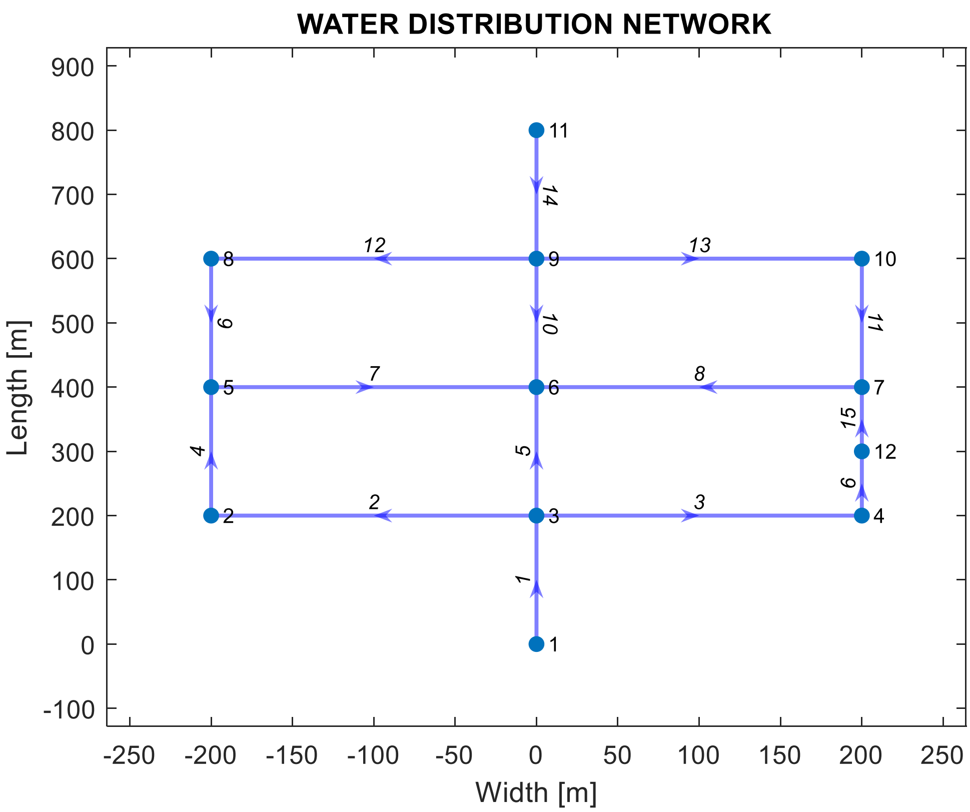

Figure 11 : Network 2 with leakage

Table 27 : Pipe Connections for Network 2 with leakage

Start Node	End Node
1	3
3	2
3	4
2	5
3	6
4	12
5	6
7	6
8	5
9	6
10	7
9	8
9	10
11	9

12	7

Table 28 : Pipe information for Network 2 with leakage

Pipe's Number	Diameter	Roughness	Loss Coefficient	Pump's Power	Pump's Efficiency
1	300	0.26	10	100000	0.75
2	250	0.26	10	0	0
3	250	0.26	10	0	0
4	250	0.26	10	0	0
5	200	0.26	10	0	0
6	250	0.26	10	0	0
7	200	0.26	10	0	0
8	200	0.26	10	0	0
9	250	0.26	10	0	0
10	200	0.26	10	0	0
11	250	0.26	10	0	0
12	250	0.26	10	0	0
13	250	0.26	10	0	0
14	300	0.26	10	100000	0.75
15	250	0.26	10	0	0
-	[mm]	[mm]	-	[Watt]	-

Table 29 : Node information for Network 2 with leakage

Node's Number	Coordinates x-direction	Coordinates y-direction	Coordinates z-direction	Node's Demand	Tank water Level
1	0	0	50	0	5
2	200	-200	50	60	0
3	200	0	55	60	0
4	200	200	60	20	0
5	400	-200	55	60	0
6	400	0	60	200	0
7	400	200	65	20	0

8	600	-200	60	20	0
9	600	0	65	20	0
10	600	200	70	60	0
11	800	0	60	0	10
12	300	200	62.5	5	0
-	[m]	[m]	[m]	[Liters]	[m]

After formulating the data section we proceed to the results:

Table 30 : Flow and Velocity Results for Network 2 with leakage

Pipe's Number	Volumetric Flow	Velocity
1	239.6355	3.3901
2	76.4311	1.5570
3	48.9217	0.9966
4	16.4311	0.3347
5	54.2827	1.7279
6	28.9217	0.5892
7	28.8843	0.9194
8	42.0255	1.3377
9	72.4532	1.4760
10	74.8075	2.3812
11	38.1038	0.7762
12	92.4532	1.8834
13	98.1038	1.9986
14	285.3645	4.0371
15	23.9217	0.4873
-	[Liters/sec]	[m/sec]

Table 31 : Head and Pressure Results for Network 2 with leakage

Node's Number	Head	Pressure Height	Pressure
1	55	5	48.9462
2	70.2538	20.2538	198.2693

3	73.5363	18.5363	181.4567
4	72.1753	12.1753	119.1871
5	70.0921	15.0921	147.7404
6	68.6957	8.6957	85.1243
7	71.6160	6.6160	64.7657
8	73.0455	13.0455	127.7055
9	77.8295	12.8295	125.5913
10	72.4488	2.4488	23.9723
11	70	10	97.8925
12	71.8441	9.3441	91.4720
-	[m]	[m]	[KPascal]

3.2.2 EPANET

In order to validate the results from our model we run the hydraulic simulation for the new networks using the EPANET. So let's review all the mentioned before networks and their results provided by EPANET.

1st Network with leakage – (8 Pipes & 7 Nodes)

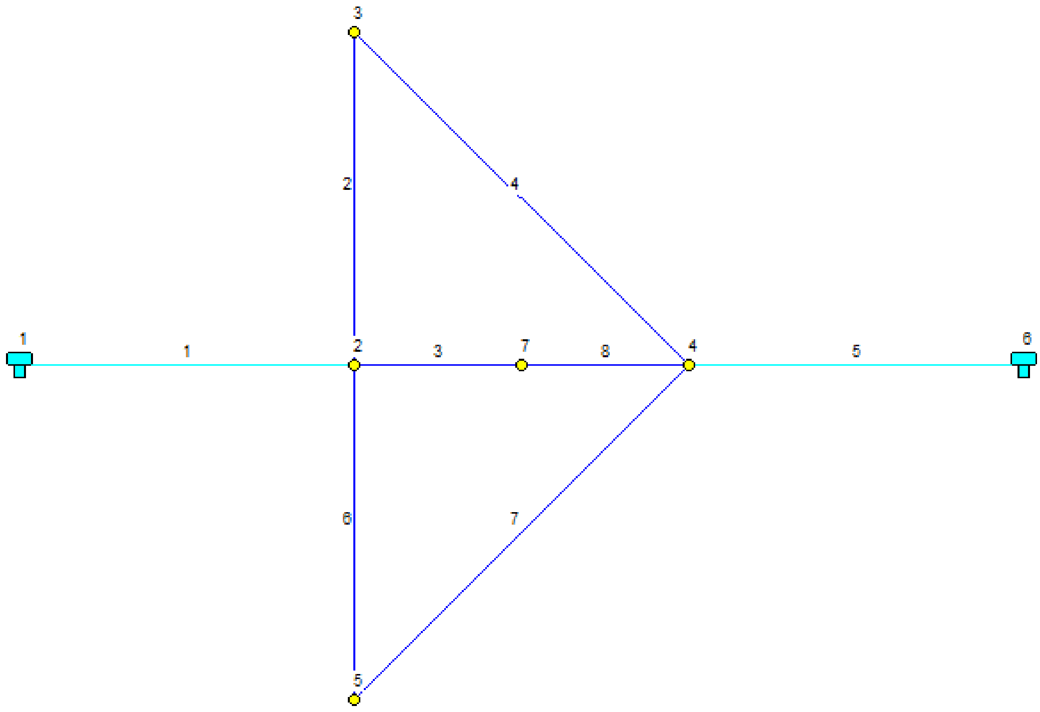

Figure 12 : Network 1 with leakage in EPANET

Table 32 : Flow and Velocity Results for Network 1 with leakage from EPANET

Pipe's Number	Volumetric Flow	Velocity
1	26.4539	0.3742
2	5.1957	0.1654
3	6.0626	0.1235
4	6.8043	0.1386
5	27.5451	0.3897
6	5.1957	0.1654
7	6.8043	0.1386
8	1.0626	0.0216
-	[Liters/sec]	[m/sec]

Table 33 : Head and Pressure Results for Network 1 with leakage from EPANET

Node's Number	Head	Pressure Height	Pressure
1	100	50	489.4625
2	99.8938	99.8938	977.8951

3	99.8569	99.8569	977.5339
4	99.8853	99.8853	977.8119
5	99.8569	99.8569	977.5339
6	100	50	489.4625
7	99.8857	99.8857	977.8158
-	[m]	[m]	[KPascal]

2nd Network with leakage – (15 Pipes & 12 Nodes)

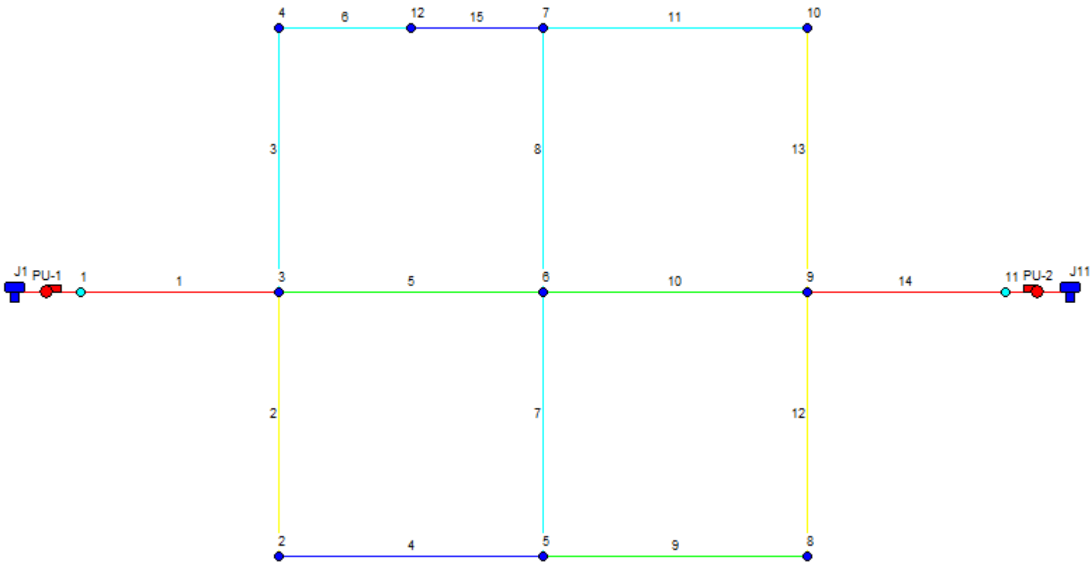

Figure 13 : Network 2 with leakage in EPANET

Table 34 : Flow and Velocity Results for Network 2 with leakage from EPANET

Pipe's Number	Volumetric Flow	Velocity
1	239.6186	3.3899
2	76.4242	1.5569
3	48.9142	0.9965
4	16.4242	0.3346
5	54.2802	1.7278
6	28.9142	0.5890
7	28.8816	0.9193

8	42.0234	1.3376
9	72.4574	1.4761
10	74.8148	2.3814
11	38.1092	0.7763
12	92.4574	1.8835
13	98.1092	1.9987
14	285.3813	4.0373
15	23.9142	0.4872
-	[Liters/sec]	[m/sec]

Table 35 : Head and Pressure Results for Network 2 with leakage from EPANET

Node's Number	Head	Pressure Height	Pressure
1	55	5	48.9462
2	70.2645	20.2645	198.2693
3	73.5453	18.5353	181.4567
4	72.1850	12.1850	119.1871
5	70.1029	15.1029	147.7404
6	68.7067	8.7067	85.1243
7	71.6261	6.6261	64.7657
8	73.0556	13.0556	127.7055
9	77.8381	12.8381	125.5913
10	72.4591	2.4591	23.9723
11	70	10	97.8925
12	71.8540	9.8540	91.4720
-	[m]	[m]	[KPascal]

3.2.3 Comparison of the Measurements

Comparing the source code and the EPANET, it seems clear that we have accurate measurements from the newly formed networks. As was stated the measurements in the second extra pipe and the extra node, for the 1st Network pipe 8 and node 7 and for the 2nd Network pipe 15 and node 12 are not going to be taken into account as measurements, therefore are dismissed.

3.3 Minimization Analysis in EPANET

The same process could be done with EPANET as well. We managed to connect our code with EPANET's libraries and functions through a toolkit. That enabled us to extract all the values needed from EPANET, given the fact that we have already designed the network in EPANET's platform. Similarly with our developed code we called the optimization function for EPANET by instructing the second to break a pipe, add a node and solve the system again and again for all pipes until f goes to zero. It is worth to clarify that in EPANET's case we used measurements from the EPANET software which were produced like analyzed previously in that chapter. The truth is that we did not gain any extra information due to the fact that we already knew where the leakage was in the first place so the algorithm constructed was enough for analyzing detection of leakage. However it is very useful, for scientific purposes, to own a code that gives you the possibility to cope with more complex problems using a credible software apart from the one we have created.

4. Sensitivities

Water distribution systems are not randomly constituted systems but there is a relative hydraulic interaction between their parts. This interconnected flow and pressure effect can be mathematically interpreted and be used in order to observe the system's dependence on fluctuating factors, with the most important to be the active node demands. So it is vital to produce a model that is capable of performing that task, concerning the flow in the pipes and the pressure at the active nodes of the network, using mathematical equations. To achieve our objective we have to look closely the formulas that were utilized in chapter 2 and see the major impact of the demand variable in the solution of the system since it affects both Q and P. Considering the demand as $\underline{\theta}$ in chapter 3 the flow and pressure are functions of $\underline{\theta}$ namely $\underline{Q}(\underline{\theta})$ and $\underline{P}(\underline{\theta})$. Having said that, we will elicit the derivatives as to the active node demand of the network using 2 different methods to calculate sensitivities, analytical and finite difference.

4.1 Analytically Derived Sensitivities

We solve the new network with the leakage and acquire the Flows $\underline{\text{Q}}$ and the Pressure $\underline{\text{P}}$. If we consider one leakage then the new network will have 1 extra pipe and one extra node from the original network. Of course it is rather obvious that all the variables $(D, f, e$ etc.) are provided from the new network as shown in chapter 3.

4.1.1 Flow

Starting with the friction factor equation (1.7) it is obvious that since the equation contains the flow term $\underline{\text{Q}}$. However $\underline{\text{Q}}$ depends on demand $\underline{\theta}$ as $\underline{\text{Q}}(\underline{\theta})$ and that is the reason why f' is a function of θ as well. So the derivative of (1.7) concerning θ is,

$$\underline{f}'(\underline{\theta}) = \frac{2.65}{\left[\ln\left(0.27 \cdot \left(\frac{e}{D}\right) + 5.74 \cdot \left(\frac{\mu \cdot \pi \cdot D}{4 \cdot \rho \cdot \underline{Q}(\underline{\theta})}\right)^{0.9}\right)\right]^3} \cdot \frac{2}{\left(0.27 \cdot \left(\frac{e}{D}\right) + 5.74 \cdot \left(\frac{\mu \cdot \pi \cdot D}{4 \cdot \rho \cdot \underline{Q}(\underline{\theta})}\right)^{0.9}\right)} \cdot$$

$$\cdot [5.166 \cdot \left(\frac{\mu \cdot \pi \cdot D}{4 \cdot \rho}\right)^{0.9} \cdot \underline{Q}(\underline{\theta})^{0.9}] \cdot \underline{Q}'(\underline{\theta}) \qquad (4.1)$$

In which all terms are known but correspond on the new network with the leakage. The final vector $\underline{f}'(\underline{\theta})$ has the same size with $\underline{Q}(\underline{\theta})$. After completing that important part we proceed with the non-linear system of equations for flow which is expressed by the relations (2.1), (2.2) and (2.3). We will repeat the same procedure for these formulas, as we did with the friction factor above and produce the derivatives towards $\underline{\theta}$.

Loops, equation (2.1)

We postulate for ease as we did in subsection 2.1.5,

$$Al = (\pm)_i \frac{8 \cdot L_i}{g \cdot \pi^2 \cdot {D_i}^5} \tag{4.2}$$

$$Bl = (\pm)_i \frac{8 \cdot K_i}{g \cdot \pi^2 \cdot {D_i}^4} \tag{4.3}$$

$$El = (\pm)_i \frac{Pu_i \cdot eff_i}{\gamma} \tag{4.4}$$

Where i corresponds to the pipes in the loops and $(\pm)_i$ signs compared with the defined positive direction.

So,

$$\left((Al \cdot \underline{f}'(\underline{\theta})) \cdot (\underline{Q}(\underline{\theta})^2)^T + 2 \cdot (Al \cdot f(\underline{\theta})) \cdot (\underline{Q}(\underline{\theta}))^T + 2 \cdot Bl \cdot \underline{Q}(\underline{\theta}) + {}^{El}\!/_{\underline{Q}(\underline{\theta})^2} \right) \cdot$$

$$\underline{Q}'(\underline{\theta}) = \begin{Bmatrix} 0 \\ \vdots \\ 0 \end{Bmatrix} \tag{4.5}$$

Or,

$$L \cdot \underline{Q}'(\underline{\theta}) = \begin{Bmatrix} 0 \\ \vdots \\ 0 \end{Bmatrix} \tag{4.5.1}$$

With, $L = (Al \cdot \underline{f}'(\underline{\theta})) \cdot (\underline{Q}(\underline{\theta})^2)^T + 2 \cdot (Al \cdot f(\underline{\theta})) \cdot (\underline{Q}(\underline{\theta}))^T + 2 \cdot Bl \cdot \underline{Q}(\underline{\theta}) + {}^{El}\!/_{\underline{Q}(\underline{\theta})^2}$

It is worth to clarify again that $\underline{Q}(\underline{\theta})$ is the flow for that governs the new network and $f(\underline{\theta})$ the friction factor for the new network with size equal to $\underline{Q}(\underline{\theta})$. In the same manner we formulate the derivatives for equation (2.2)

$$Apl = (\pm)_i \frac{8 \cdot L_i}{g \cdot \pi^2 \cdot {D_i}^5} \tag{4.6}$$

$$Bpl = (\pm)_i \frac{8 \cdot K_i}{g \cdot \pi^2 \cdot {D_i}^4} \tag{4.7}$$

$$Epl = (\pm)_i \frac{Pu_i \cdot eff_i}{\gamma} \tag{4.8}$$

where i corresponds for the pipes in the pseudo-loops and $(\pm)_i$ signs are determined given a positive defined direction.

Consequently,

$$\left((Al \cdot \underline{f}'(\underline{\theta})) \cdot (\underline{Q}(\underline{\theta})^2)^T + 2 \cdot (Al \cdot f(\underline{\theta})) \cdot (\underline{Q}(\underline{\theta}))^T + 2 \cdot Bl \cdot \underline{Q}(\underline{\theta}) + {}^{El}\!/_{\underline{Q}(\underline{\theta})^2} \right) \cdot$$

$$\underline{Q}'(\underline{\theta}) = \begin{Bmatrix} 0 \\ \vdots \\ 0 \end{Bmatrix} \tag{4.9}$$

Or,

$$Lp \cdot \underline{Q}'(\underline{\theta}) = \begin{Bmatrix} 0 \\ \vdots \\ 0 \end{Bmatrix} \tag{4.9.2}$$

With, $Lp = (Al \cdot \underline{f}'(\underline{\theta})) \cdot (\underline{Q}(\underline{\theta})^2)^T + 2 \cdot (Al \cdot f(\underline{\theta})) \cdot (\underline{Q}(\underline{\theta}))^T + 2 \cdot Bl \cdot \underline{Q}(\underline{\theta}) + {}^{El}\!/_{\underline{Q}(\underline{\theta})^2}$

Last but not least the Equilibrium of mass, equation (2.3),

$$\mathrm{S} \cdot \underline{Q}'(\underline{\theta}) = \begin{Bmatrix} 0 \\ \vdots \\ 1 \end{Bmatrix} \tag{4.10}$$

Where, S is the matrix from (2.3) and the derivative of de, second term in (2.3) is a vector with all its elements equal to zero apart from one which equals to 1. That 1 represents the node toward whom we produce our derivatives. That particular node cannot be a Tank node as S does not contain that kind of nodes. Besides, it is pointless to deem a Tank node, given the fact that their demand equals to zero.

If we examine the equations (4.5.1), (4.9.1) and (4.10) we can realize that if combine them we are able to calculate $Q'(\theta)$ by solving a linear system.

$$\begin{bmatrix} L \\ S \\ Lp \end{bmatrix} \cdot \underline{Q}'(\underline{\theta}) = \begin{Bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \end{Bmatrix} \tag{4.11}$$

where L, S and Lp the matrices that mentioned above, $\underline{Q}'(\underline{\theta})$ the matrix containing the sensitivities for flow and the vector in right is a zero vector apart from element which represents a node and equals to 1.

4.1.2 Pressure

The next part, which is equally valuable, is to compute the sensitivity matrix for the node pressures of the network. In order to accomplish that we will use the analysis stated above concerning the flow. We will start with the equation (2.4) which is the absolute value of the head difference in every pipe of the network.

For our ease we postulate,

$$Ap = \frac{8 \cdot L_i}{g \cdot \pi^2 \cdot {D_i}^5} \tag{4.12}$$

$$Bp = \frac{8 \cdot K_i}{g \cdot \pi^2 \cdot {D_i}^4} \tag{4.13}$$

$$Ep = \frac{Pu_i \cdot eff_i}{\gamma} \tag{4.14}$$

Where, i refers to all the pipes. So, the derivative towards $\underline{\theta}$ for the Head difference is,

$$\underline{DH}'(\underline{\theta}) = \left| (Ap \cdot f'(\underline{\theta})) \cdot (\underline{Q}(\underline{\theta})^2)^T + 2 \cdot (Ap \cdot \underline{f}(\underline{\theta})) \cdot (\underline{Q}(\underline{\theta}))^T + 2 \cdot Bp \cdot \underline{Q}(\underline{\theta}) \, {}^{Ep}\!/_{\underline{Q}(\underline{\theta})^2} \right| \cdot \left| \underline{Q}'(\underline{\theta}) \right| \tag{4.15}$$

We will proceed with equations (2.5) and (2.6) and produce the derivatives,

$$A \cdot \underline{H}'(\underline{\theta}) = \underline{DH}'(\underline{\theta}) \tag{4.16}$$

And

$$\underline{P}'(\underline{\theta}) = \underline{H}'(\underline{\theta}) \tag{4.17}$$

If we combine (4.16) and (4.17) we get the linear system for finding $\underline{P}'(\underline{\theta})$.

$$A \cdot \underline{P}'(\underline{\theta}) = \underline{DH}'(\underline{\theta}) \tag{4.18}$$

With this last equation we conclude the theory on analytical derived sensitivities for flow and pressure and we move on to the other way of computing those, which is by finite difference.

4.2 Finite Difference

It is a very easy way of producing the needed sensitivities because we use the definition of derivatives both for Flow and for Pressure. Bellow we present the most precise definition for producing derivatives for a function $'g'$ at a certain point $'a'$ and we will be more thorough on how to adapt it to our system.

$$g'(a) = \frac{g(a + da) \ - g(a - da)}{2 \cdot da} \tag{4.19}$$

where, da is a very small value compared to a.

In our case the value at which we aim to produce the derivatives is θ and as we clarified before, θ is a node demand. Given the fact that θ is a positive number greater than zero, (in liters /sec), we assume that $d\theta \leq 10^{-4}$. So, for flow

$$Q'(\theta) = \frac{Q(\theta + d\theta) \ - Q(\theta - d\theta)}{2 \cdot d\theta} \tag{4.20}$$

And for pressure,

$$P'(\theta) = \frac{P(\theta + d\theta) \ - P(\theta - d\theta)}{2 \cdot d\theta} \tag{4.21}$$

In other words the main point is that we find the flow and pressure in all pipes and active nodes respectively given the fact that we have changed the demand of a certain node θ by $\pm d\theta$. Of course, we underline again that the node of our choice cannot be a Tank Node.

4.3 Networks and Results

The Sensitivity chapter is extremely important for leakage detection and optimal sensor placement analysis. For that reason we need to be certain of the accuracy and the validity of the results of $\underline{Q}'(\theta)$ and $\underline{P}'(\theta)$ and that is why we used 2 methods. Let's compare the Sensitivity Results for the following network considering leakage equal to 5ltrs/sec in pipe 6-(Node 15) for the 2nd network and 25ltrs/sec in pipe 33-(Node 34) for the 3rd network.

2rd Network with leakage – (15 Pipes & 12 Nodes)

See figure 10 to get an exact picture of the network.

Table 36 : Analytical and Finite Difference Sensitivities for Flow, Network 2

Pipe's Number	Analytical	Finite Difference
1	0.5168	0.5168
2	-0.0163	-0.0163
3	0.5065	0.5065
4	-0.0163	-0.0163
5	0.0267	0.0267
6	0.5065	0.5065
7	0.0674	0.0674
8	-0.0839	-0.0839
9	0.0838	0.0838
10	0.0898	0.0898
11	0.3096	0.3096
12	0.0838	0.0838
13	0.3096	0.3096
14	0.4832	0.4832
15	-0.4935	-0.4935

Table 37 : Analytical and Finite Difference Sensitivities for Flow, Network 2

Node's Number	Analytical	Finite Difference
1	0	0
2	-0.1252	-0.1252
3	-0.1266	-0.1266
4	-0.1543	-0.1543
5	-0.1249	-0.1249
6	-0.1313	-0.1313
7	-0.1565	-0.1565
8	-0.1181	-0.1181
9	-0.1096	-0.1096
10	-0.1432	-0.1432
11	0	0
12	-0.1657	-0.1657

3rd Network with leakage – (53 Pipes & 34 Nodes)

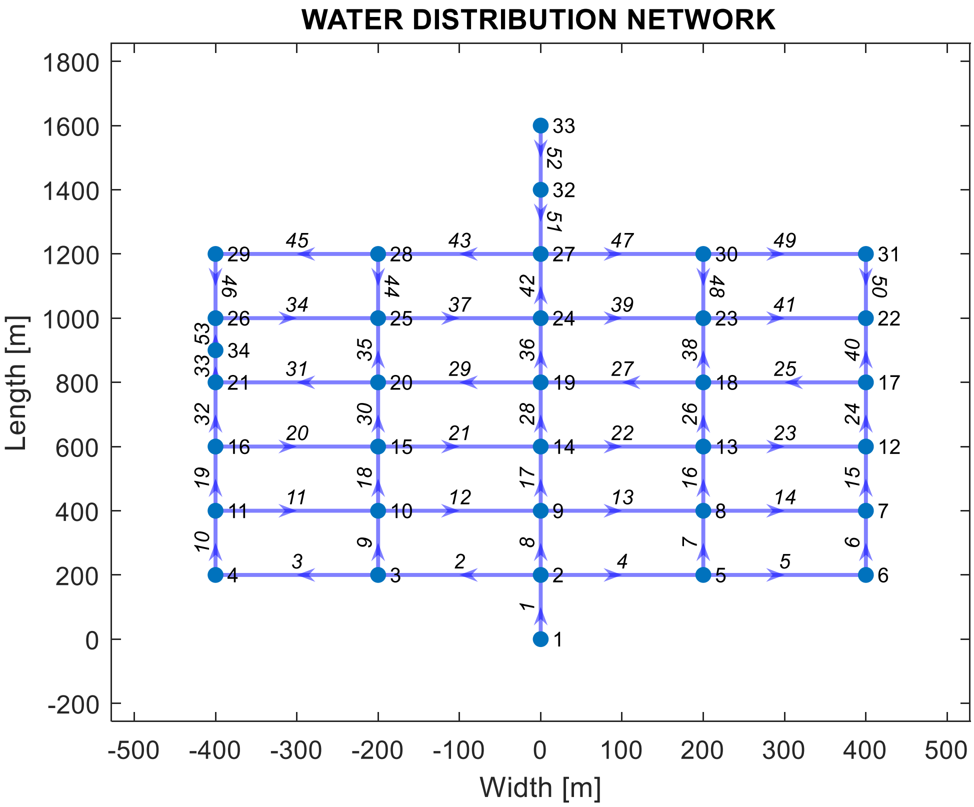

Figure 14 : Network 3 with leakage 25ltrs/sec in pipe 33-(Node 34)

Table 38 : Analytical and Finite Difference Sensitivities for Flow, Network 3

Pipe's Number	Analytical	Finite Difference
1	0.5785	0.5785
2	0.1951	0.1951
3	0.0911	0.0911
4	0.1784	0.1784
5	0.0756	0.0756
6	0.0756	0.0756
7	0.1028	0.1028
8	0.2051	0.2051
9	0.1039	0.1039
10	0.0911	0.0911
11	-0.0697	-0.0697

12	-0.0860	-0.0860
13	0.0223	0.0223
14	0.0235	0.0235
15	0.0991	0.0991
16	0.1015	0.1015
17	0.0968	0.0968
18	0.1203	0.1203
19	0.1608	0.1608
20	-0.1359	-0.1359
21	-0.1597	-0.1597
22	-0.1292	-0.1292
23	-0.0633	-0.0633
24	0.0357	0.0357
25	0.0698	0.0698
26	0.0357	0.0357
27	0.1351	0.1351
28	0.0663	0.0663
29	0.2076	0.2076
30	0.1441	0.1441
31	0.2868	0.2868
32	0.2968	0.2968
33	0.5836	0.5836
34	-0.2700	-0.2700
35	0.0649	0.0649
36	-0.0063	-0.0063
37	-0.1712	-0.1712
38	-0.0296	-0.0296
39	-0.0424	-0.0424
40	-0.0341	-0.0341
41	-0.0075	-0.0075
42	-0.1350	-0.1350
43	0.1804	0.1804
44	0.0340	0.0340

45	0.1465	0.1465
46	0.1465	0.1465
47	0.1061	0.1061
48	0.0645	0.0645
49	0.0415	0.0415
50	0.0415	0.0415
51	0.4215	0.4215
52	0.4215	0.4215
53	-0.4164	-0.4164

Table 39 : Analytical and Finite Difference Sensitivities for Pressure, Network 3

Node's Number	Analytical	Finite Difference
1	0	0
2	-0.2114	-0.2114
3	-0.2188	-0.2188
4	-0.2203	-0.2203
5	-0.2181	-0.2181
6	-0.2193	-0.2193
7	-0.2200	-0.2200
8	-0.2198	-0.2198
9	-0.2196	-0.2196
10	-0.2205	-0.2205
11	-0.2211	-0.2211
12	-0.2208	-0.2208
13	-0.2210	-0.2210
14	-0.2210	-0.2210
15	-0.2219	-0.2219
16	-0.2227	-0.2227
17	-0.2209	-0.2209
18	-0.2211	-0.2211
19	-0.2212	-0.2212
20	-0.2226	-0.2226
21	-0.2251	-0.2251

22	-0.2208	-0.2208
23	-0.2208	-0.2208
24	-0.2212	-0.2212
25	-0.2230	-0.2230
26	-0.2257	-0.2257
27	-0.2167	-0.2167
28	-0.2225	-0.2225
29	-0.2246	-0.2246
30	-0.2200	-0.2200
31	-0.2206	-0.2206
32	-0.1742	-0.1742
33	0	0
34	-0.2277	-0.2277

4.4 Conclusions on Sensitivities

It is obvious that our results match. This is clear evidence that the sensitivity analysis was correct analytically, which was the most difficult method among the two that were developed. The next chapter is the final of the current thesis and concerns the optimal sensor placement for detecting leakage in water distribution networks.

5. Optimal Sensor Placement.

Water is a valuable commodity around the globe, so we try to minimize its losses from the moment it gets into our possession. In real networks where demands are very high, having a leakage may be devastating especially if exists for a long period of time. It is important how to spot it accurately and quickly without the use of extravagant means. Sensors are expensive equipment which may be useless if we cannot determine where to place them, particularly in a large water distribution network. In this chapter we will pay our attention on how we can point out the pipe with the leakage or the proximity area of the leakage. To achieve the optimal sensor placement we will make use a partial use of the Bayesian Analysis.

5.1 Bayesian Analysis on Optimal Sensor Placement

The Bayesian Analysis is based on the results derived from the Network by solving the systems of equations described in chapter 2 and chapter 4 in order to find out the best sensor location for a variety of leakages and networks. Consider $\underline{\theta} \in R^{N_\theta}$ to be the vector of model parameters to be estimated given a set of data $\underline{d} \equiv \underline{d}(\underline{\delta}) \in R^N$ of flow or pressure quantities at locations $\underline{\delta}$. The vector $\underline{\delta}$ contains the numbers of pipes or nodes in which we can place a sensor. Let $\underline{g}(\underline{\theta};\underline{\delta})$ be the vector of the measurements of the same flow or pressure calculated by our model for specific values of the parameter $\underline{\theta}$. The prediction error equation is introduced,

$$\underline{d} = \underline{g}(\underline{\theta};\underline{\delta}) + \underline{e} \tag{5.1}$$

where, $\underline{e}$ is the additive prediction error due to the measurement error. The prediction error is modeled as a Gaussian vector, whose mean is equal to zero and its covariance equal to $\Sigma(\underline{\sigma}) \in R^{N\times N}$, where $\underline{\sigma}$ contains the parameters that define the correlation structure of Σ. Applying the Bayesian theorem , the Posterior Probability Density function or PDF of $\underline{\theta}$,given the measured data $\underline{d}$, is given by,

$$p(\underline{\theta}|\underline{\sigma},\underline{d},\underline{\delta}) = c \cdot \frac{1}{(\sqrt{2\cdot\pi})^N \cdot \sqrt{det(\Sigma(\underline{\sigma}))}} \cdot \exp\left[-\frac{N}{2}\cdot \mathrm{J}(\underline{\theta};\underline{\sigma},\underline{d},\underline{\delta})\right] \tag{5.2}$$

Where,

$$\mathrm{J}(\underline{\theta};\underline{\sigma},\underline{d},\underline{\delta}) = \frac{1}{N}\left[\underline{d} - \underline{g}(\underline{\theta};\underline{\delta})\right]^T \cdot \Sigma^{-1}(\underline{\sigma}) \cdot \left[\underline{d} - \underline{g}(\underline{\theta};\underline{\delta})\right] \tag{5.3}$$

which, expresses the deviation between the measured and the model predicted quantities. The PDF $\pi(\underline{\theta})$ is the prior distribution for $\underline{\theta}$ and c is a normalization constant that ensures that the posterior PDF $p(\underline{\theta}|\underline{\sigma},\underline{d},\underline{\delta})$ integrates to 1.

5.2 Information Entropy

The PDF $(\underline{\theta}|\underline{\sigma},\underline{d},\underline{\delta})$, given by the equation (5.2), quantifies the posterior uncertainty in the parameter values $\underline{\theta}$ based on the information contained in the measured data. The information entropy is given by,

$$h_{\underline{\theta}}(\underline{\delta};\underline{\sigma},\underline{d}) = -\int ln\ p(\underline{\theta}|\underline{\sigma},\underline{d},\underline{\delta}) \cdot p(\underline{\theta}|\underline{\sigma},\underline{d},\underline{\delta})\ d\underline{\theta} \tag{5.4}$$

Is a scalar measure of the uncertainty of the model parameters $\underline{\theta}$. It depends on the location of vector $\underline{\delta}$ of the sensors, the correlation structure of the prediction error and the details in $\underline{d}$. The multidimensional integral shown in (5.4) is a Laplace-type integral that can be asymptotically approximated for a large number of data as displayed,

$$h_{\underline{\theta}}(\underline{\delta};\underline{\sigma},\underline{d}) \sim H(\underline{\delta};\underline{\theta}_0,\underline{d}) = \frac{1}{2} \cdot N_\theta \cdot ln(2 \cdot \pi) - \frac{1}{2} ln\, det\left[Q(\underline{\delta};\underline{\theta}_0,\underline{\sigma}) + Q_\pi(\underline{\theta}_0)\right] \tag{5.5}$$

Where, $\underline{\theta}_0$ are the values of $\underline{\theta}$ that minimize $J(\underline{\theta};\underline{\sigma},\underline{d},\underline{\delta})$; $Q(\underline{\delta};\underline{\theta}_0,\underline{\sigma})$ is asymptotically approximated by,

$$Q(\underline{\delta};\underline{\theta},\underline{\sigma}) = \nabla_{\underline{\theta}}\, \underline{g}(\underline{\theta};\underline{\delta})^T \cdot \Sigma^{-1}(\underline{\sigma},\underline{\delta}) \cdot \nabla_{\underline{\theta}}{}^T \underline{g}(\underline{\theta};\underline{\delta}) + Q_\pi(\underline{\theta}_0) \tag{5.6}$$

Computed at N locations where the sensors are placed; and $Q_\pi(\underline{\theta}_0) = -\nabla_{\underline{\theta}}{}^T \nabla_{\underline{\theta}} ln(\pi(\underline{\theta})$, evaluated by the value $\underline{\theta}_0$, represents the negative hessian of the Hessian of the natural logarithm of the prior distribution of the model parameters. For Uniform prior the term above equals to zero but for the specific case of a Gaussian distribution $Q_\pi(\underline{\theta}_0) = Q_\pi$,which is the inverse of the covariance matrix of the Gaussian distribution and thus it is constant, independent of $\underline{\theta}$. Having concluded the explanation of (5.6), if we go back to relation (5.5) and consider that $\frac{1}{2} \cdot N_\theta .\, ln(2 \cdot \pi)$ is a small term of no importance we can assume that,

$$H(\underline{\delta};\underline{\theta}_0,\underline{d}) = -\frac{1}{2} ln\, det\left[\nabla_{\underline{\theta}}\, \underline{g}(\underline{\theta};\underline{\delta})^T \cdot \Sigma^{-1}(\underline{\sigma},\underline{\delta}) \cdot \nabla_{\underline{\theta}}{}^T \underline{g}(\underline{\theta};\underline{\delta}) + Q_\pi(\underline{\theta}_0)\right] \tag{5.7}$$

Based on (5.7) we introduce the Utility function,

$$U(\underline{\delta};\underline{\theta}_0) + c = H(\underline{\delta};\underline{\theta}_0,\underline{d}) \rightarrow U(\underline{\delta};\underline{\theta}_0) = H(\underline{\delta};\underline{\theta}_0,\underline{d}) - c \tag{5.8}$$

The main objective of our analysis is to minimize the Utility function and through that minimization process we will spot the optimal $\underline{\delta}$, sensor location. By looking closely at (5.8) we can realize that minimizing $U(\underline{\delta};\underline{\theta}_0)$ has the same result as minimizing $H(\underline{\delta};\underline{\theta}_0,\underline{d})$. Let's examine the terms of equation (5.7) for our case.

$$\nabla_{\underline{\theta}}\, \underline{g}(\underline{\theta};\underline{\delta})^T = \begin{bmatrix} {dg_1}/{d\theta_1} & \cdots & {dg_n}/{d\theta_1} \\ \vdots & \ddots & \vdots \\ {dg_1}/{d\theta_n} & \cdots & {dg_n}/{d\theta_n} \end{bmatrix}$$

where, $\underline{g}(\underline{\theta};\underline{\delta})$ represents the sensitivities produced given demand θ at pipes or nodes δ, depending on whether the sensitivities are produced for flows or pressures. As happened in the case of leakage in the measurements section, the extra node and pipe are dismissed in the sensitivities as well. Last but not least, n is the number of θ. Moving on,

$$\nabla_{\underline{\theta}}{}^T\, \underline{g}(\underline{\theta};\underline{\delta}) = \left[\nabla_{\underline{\theta}}\, \underline{g}(\underline{\theta};\underline{\delta})^T\right]^T$$

while *Σ* is

$$\Sigma(\sigma,\delta) = \begin{bmatrix} {s_1}^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & {s_n}^2 \end{bmatrix}$$

where, $s_i = b \cdot \underline{g}(\underline{\theta};\underline{\delta})$, in which $\underline{g}(\underline{\theta};\underline{\delta})$ represents the measurements acquired by solving the network at $\underline{\delta}$ which, as was mentioned, is a vector containing the pipes or nodes that we place the sensors. In other words, a location vector. Moreover b is a constant and for our problem takes the value of 0.05.

Finally,

$$Q_\pi(\underline{\theta}_0) = \begin{bmatrix} {1}/{{\sigma_1}^2} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & {1}/{{\sigma_n}^2} \end{bmatrix}$$

In which, $\sigma_i = a \cdot \mu$ with μ to be the mean. Due to the fact that the data are not available the optimal value $\underline{\theta}_0$ is an assumed nominal value. So μ is equal to the value of leakage if we perform the analysis for flows or is equal to the pressure at the leakage node in case we are performing our analysis for pressures, a is a constant that we appoint its value to be 100.

5.3 Results

In order to produce the results for optimal sensor placement for flow and pressure respectively, we constructed a code that minimizes the Utility function and presents $\underline{\delta}$ for that achieves it. There are 2 different approaches to accomplish the minimization of the Utility function in accordance to $\underline{\delta}$. The first is called FSSP, which is simple step by step procedure. To be more explicit we start by finding in which pipe or node, according to the case examined, the Utility function takes the minimum value. That is the place for the first sensor. Then we repeat the same procedure for the second sensor given the fact that the first sensor is placed somewhere and through the minimization of the Utility we spot the best location for the second sensor. The process goes on for a number of sensors that we have already determined. The second way is called BSSP, that function uses the modal identification method. In this particular case we perform the opposite process by randomly placing the sensors in various parts of the network. Then we remove the sensor that maximizes the Utility function, namely with the worst Utility function, until we find the optimal place for the first sensor. The procedure goes on until we fill all the best sensor location given our initial number of sensors. These both techniques were applied in the water distribution networks that we analyzed in previous sections in order to verify the validity of our results for the optimal sensor placement, for flow and pressure, in order to detect the location of the leakage.

2rd Network (See figure 4)

We assume a leakage in pipe 6 equal to 5 liters/sec. Given the measurements and Sensitivities acquired for that amount of leakage we minimize the Utility function using FSSP and BSSP. Below, are displayed the results for both methods for flow and pressure,

Leakage in pipe 6-Flow

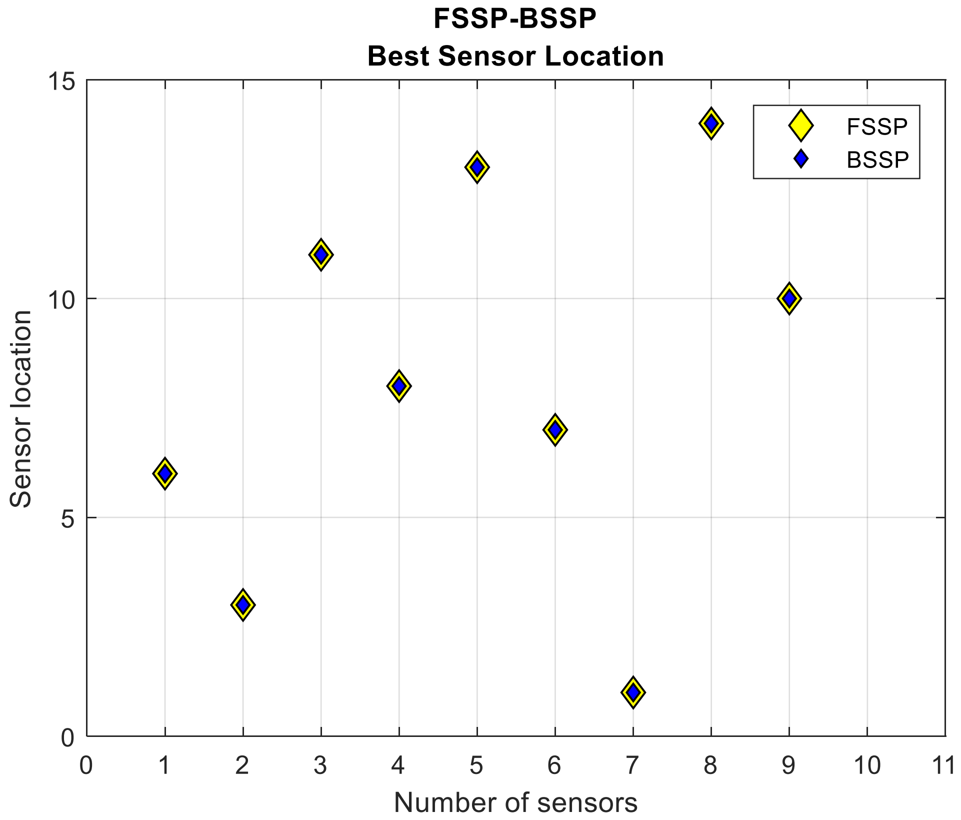

Figure 15 : Best sensor location for flow

Table 40: First 4 Flow sensors from figure 15

Pipes	**6**	**3**	**11**	**8**

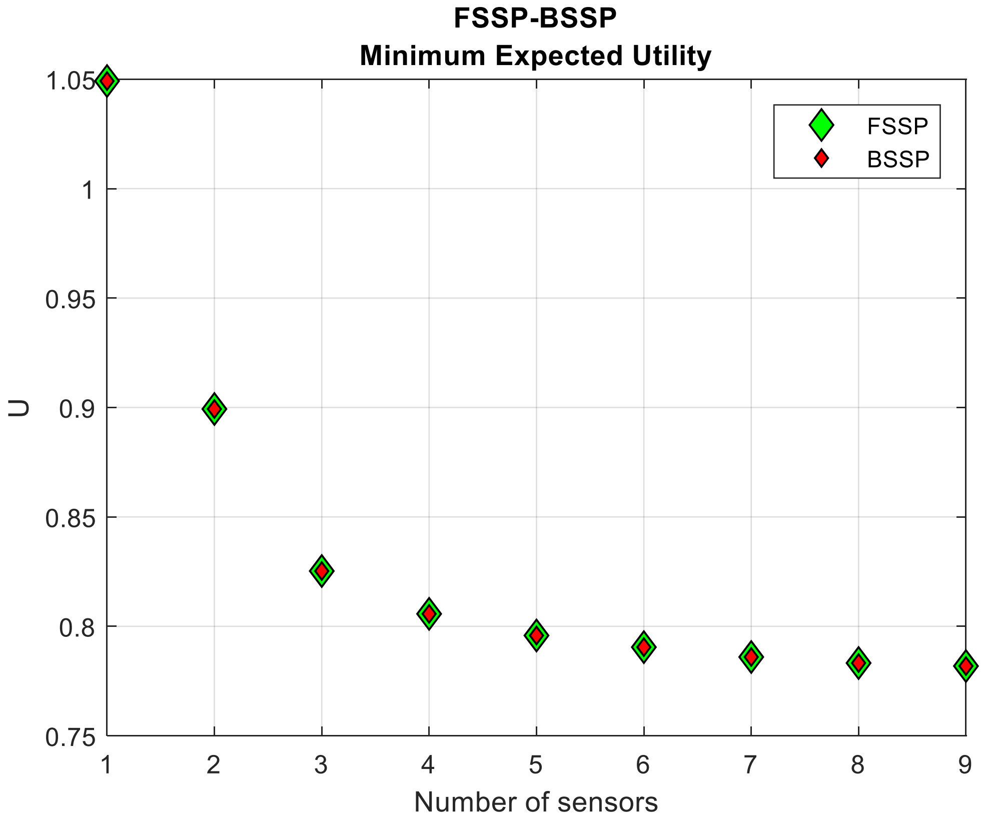

Figure 16 : Utility function values as to the number of sensors for Flow

Leakage in pipe 6-Pressure

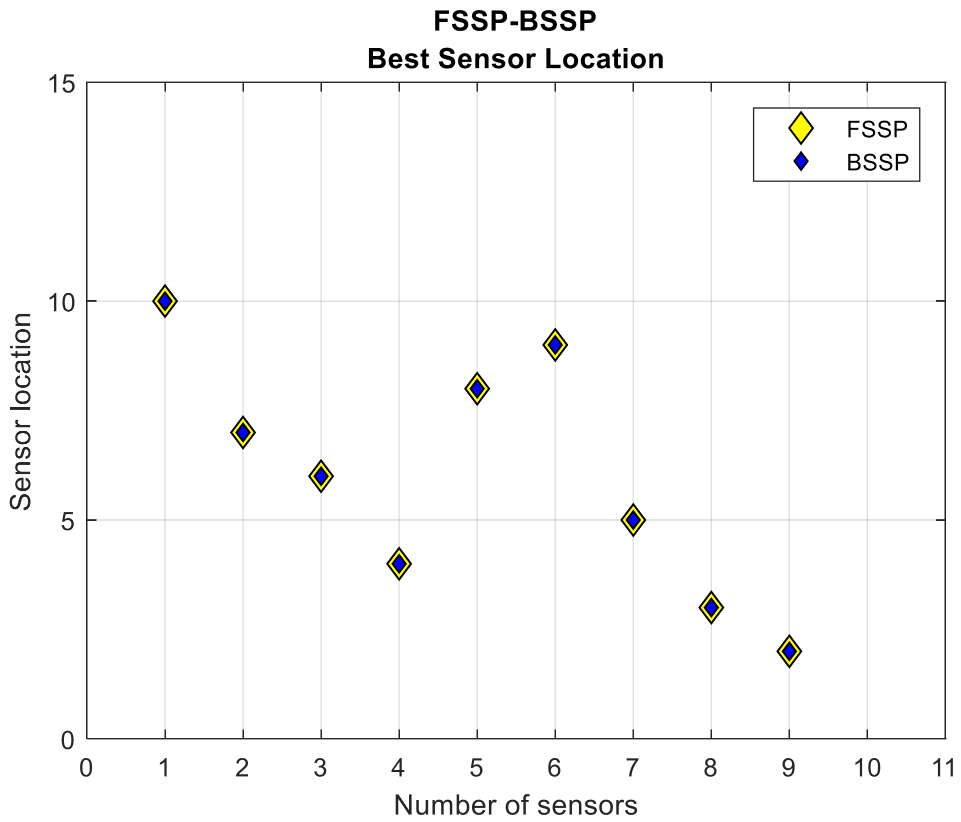

Figure 17 : Best sensor location for Pressure

Table 41: First 4 Pressure sensors form Figure 17

Nodes	**10**	**7**	**6**	**4**

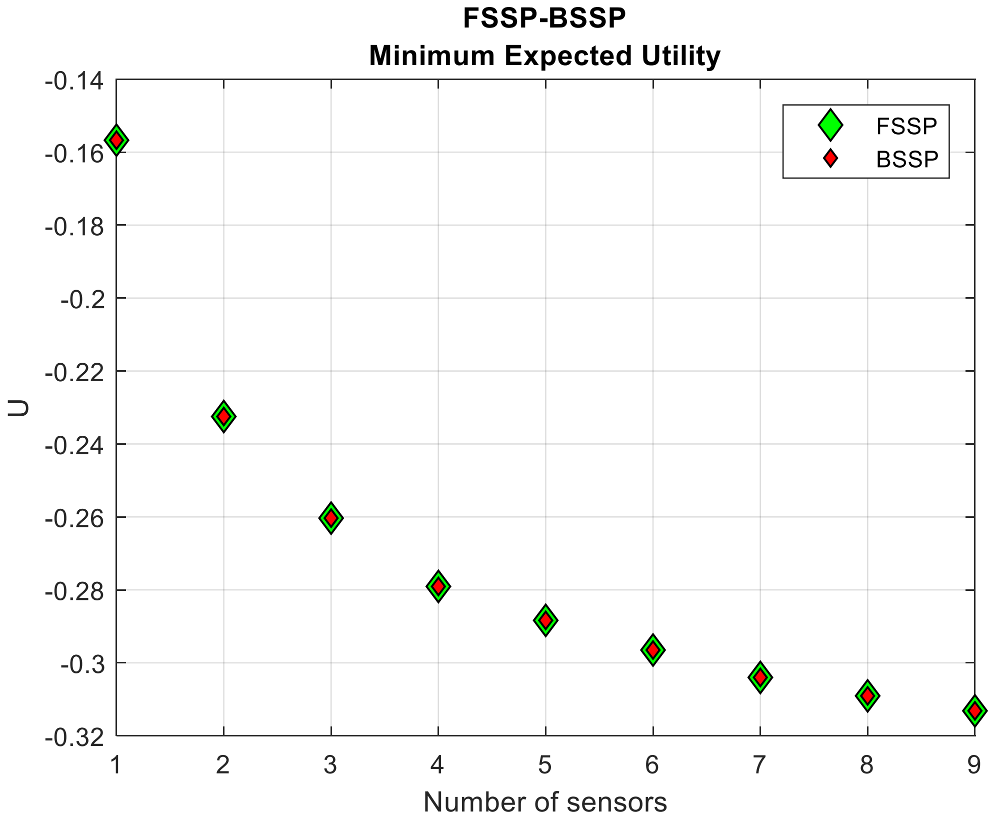

Figure 18 : Utility function values as to the number of sensors for Pressure

3rd Network (See figure 5)

We will investigate 2 cases of leakage in the big network separately. We assume a leakage in pipe 25 equal to 5liters/sec and a leakage in pipe 33 equal to 25 liters/sec. The measurements and sensitivity matrices are known for this case as well. We perform the same task as described. Consequently,

Leakage in pipe 25-Flow

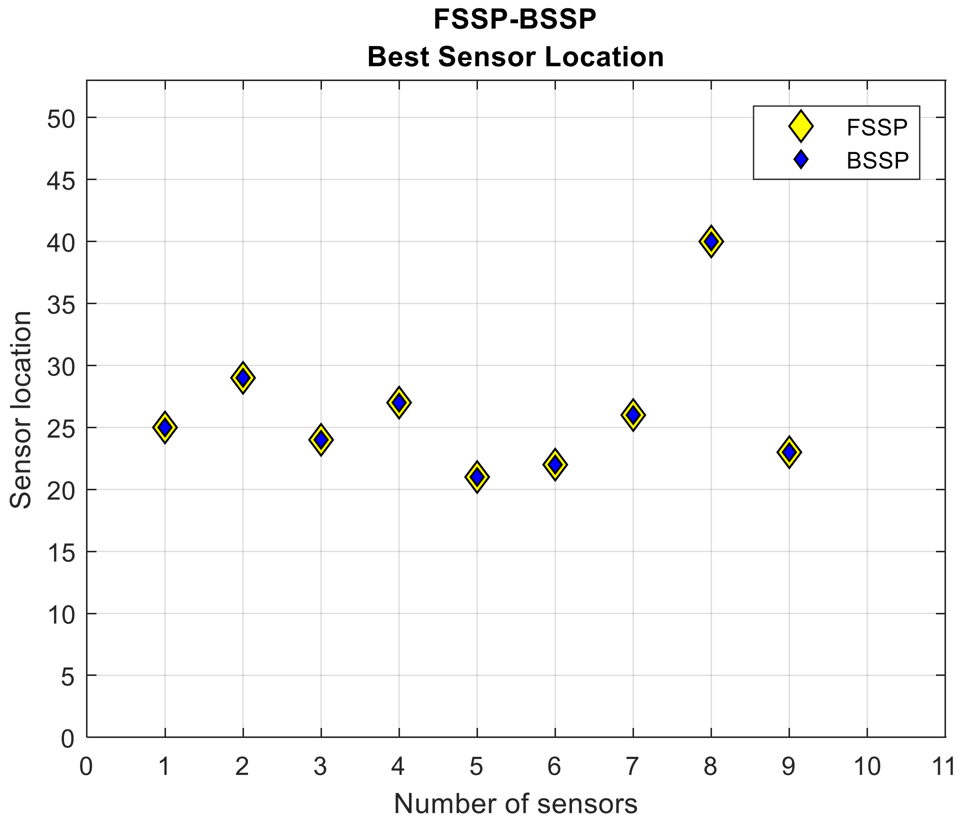

Figure 19 : Best sensor location for Flow in Network 3 with leakage in pipe 25

Table 42: First 4 Flow sensors from figure 19

Pipes	25	29	24	27

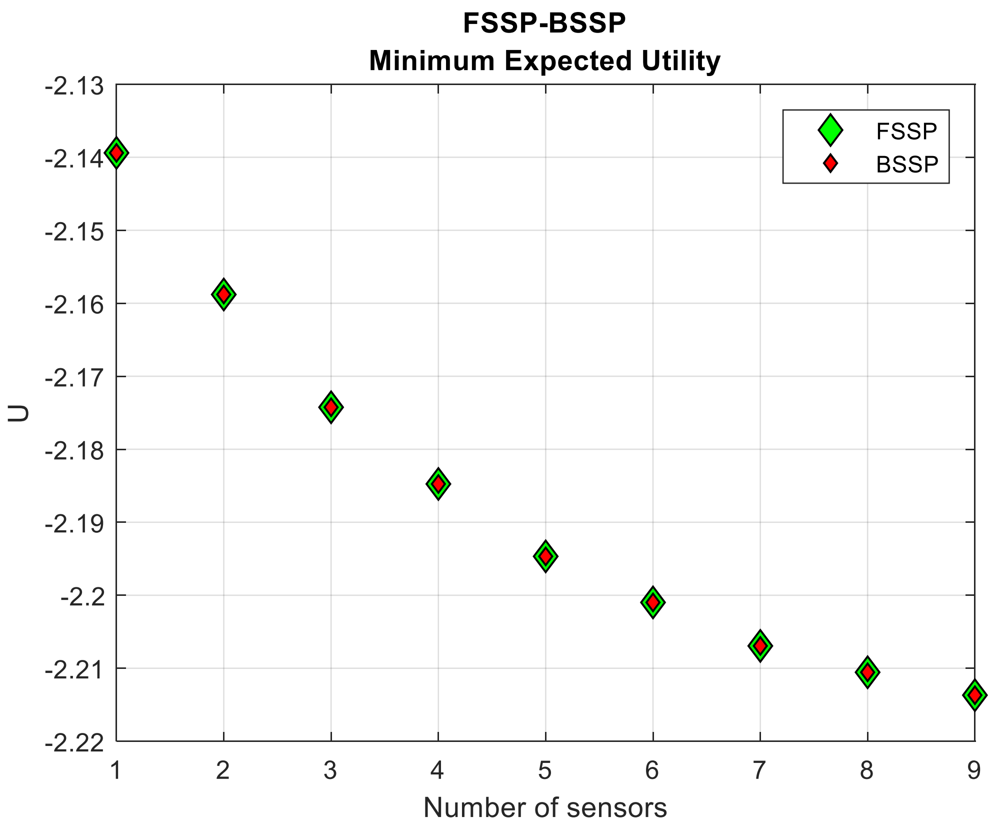

Figure 20 : Utility function as to the number of sensors for Flow in Network 3 with leakage in pipe 25

Leakage in pipe 25-Pressure

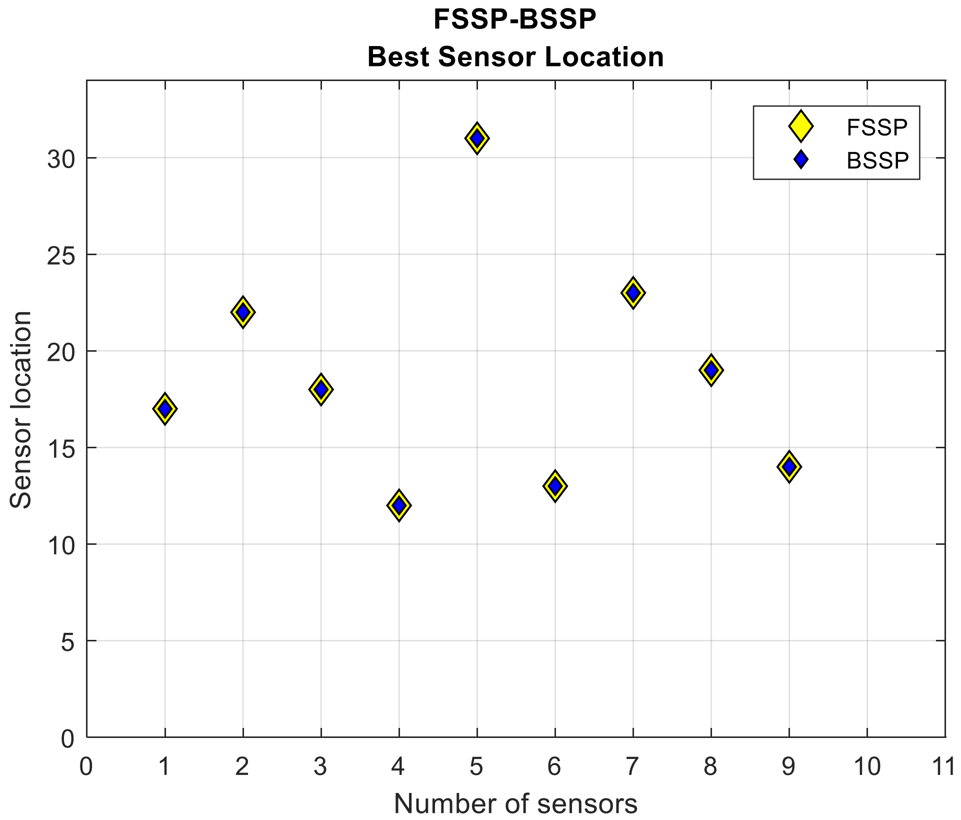

Figure 21 : Best sensor location for Pressure in Network 3 for leakage in pipe 25

Table 43 : First 4 Flow sensors from figure 21

Nodes	17	22	18	12

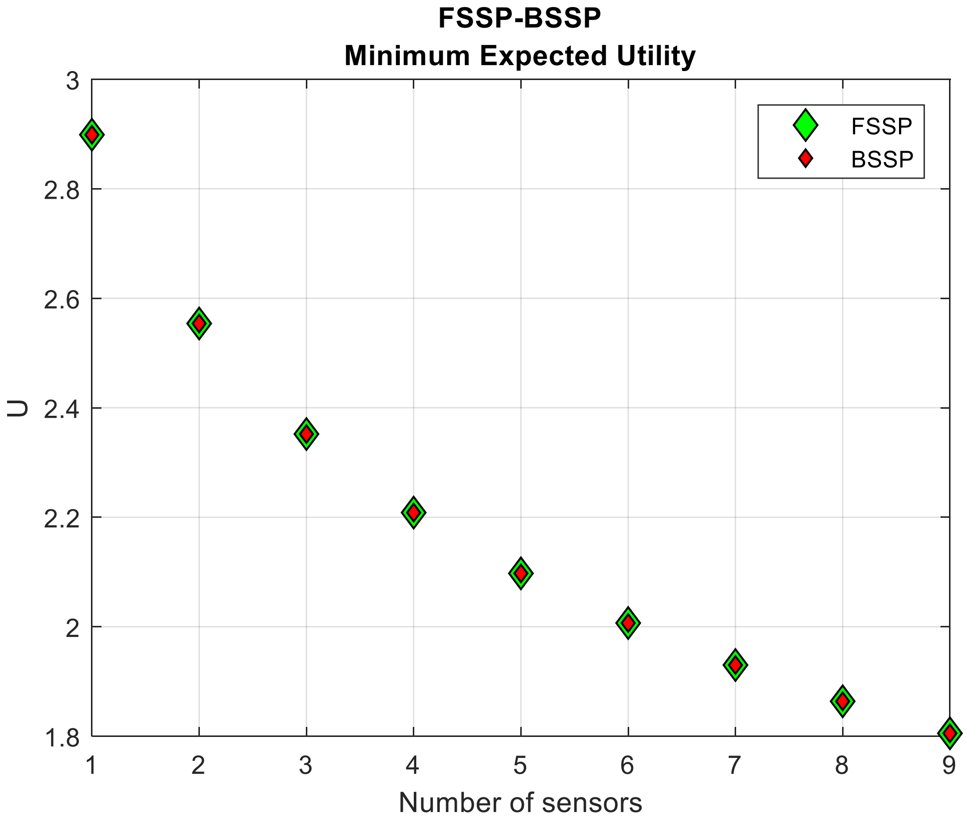

Figure 22 : Utility function values as to the number of sensors for Pressure in Network 3 with leakage in pipe 25

Leakage in pipe 33-Flow

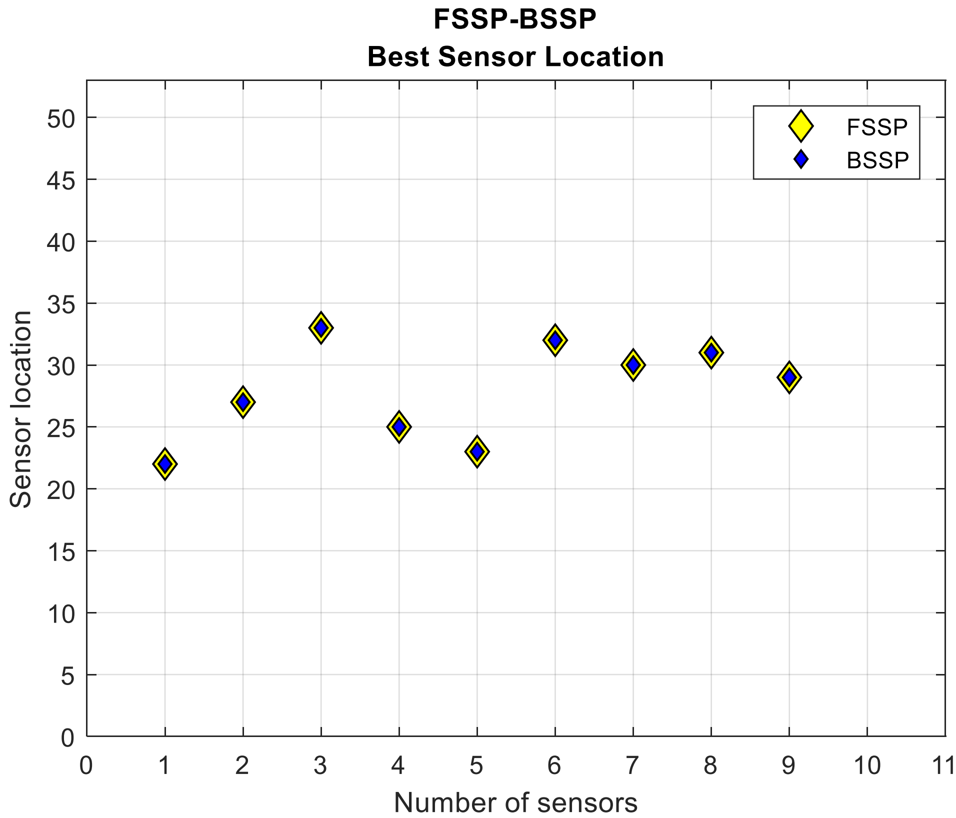

Figure 23 : Best sensor location for Flow in Network 3 with leakage in pipe 33

Table 44: First 4 Flow sensors from figure 23

Pipes	22	27	33	25

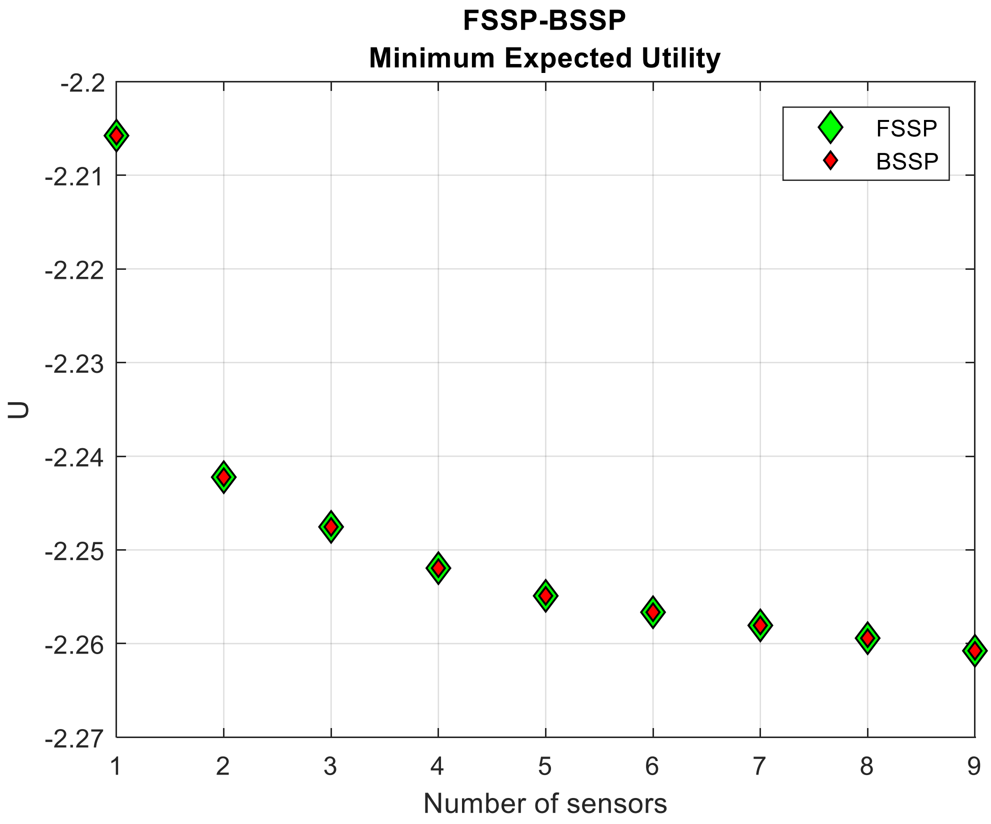

Figure 24 : Utility function values as to the number of sensors for Flow for Network 3 with leakage in pipe 33

Leakage in pipe 33-Pressure

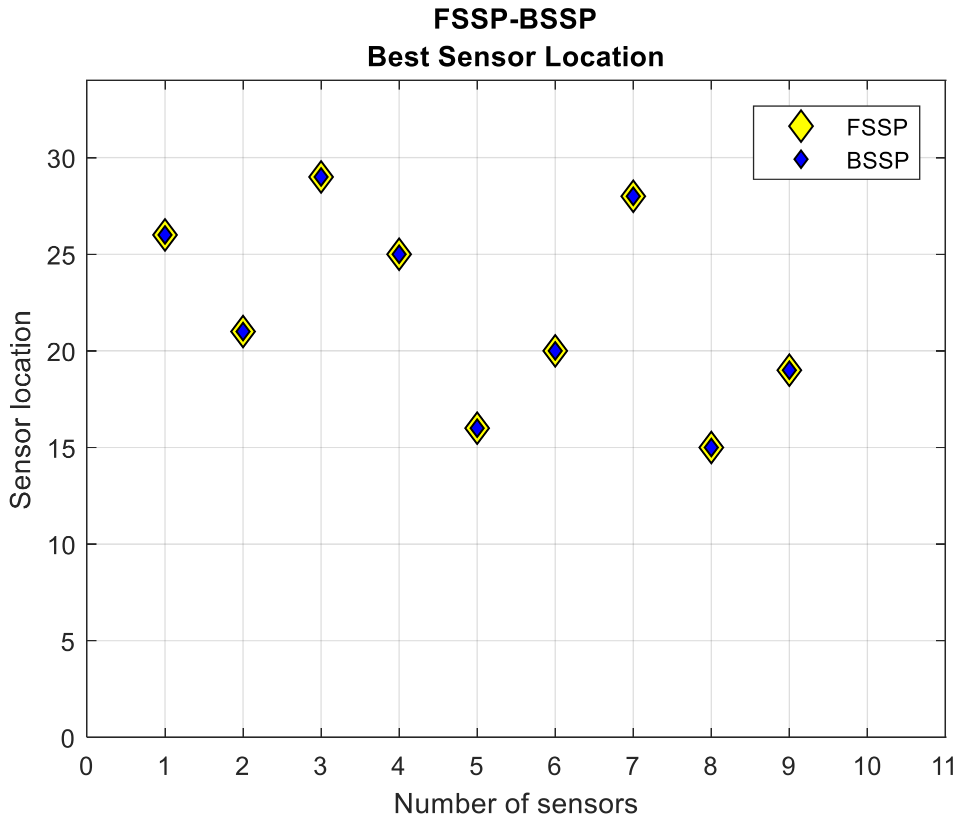

Figure 25 : Best sensor location for Pressure for Network 3 with leakage in pipe 33

Table 45: First 4 Pressure sensors from figure 25

Nodes	**26**	**21**	**29**	**25**

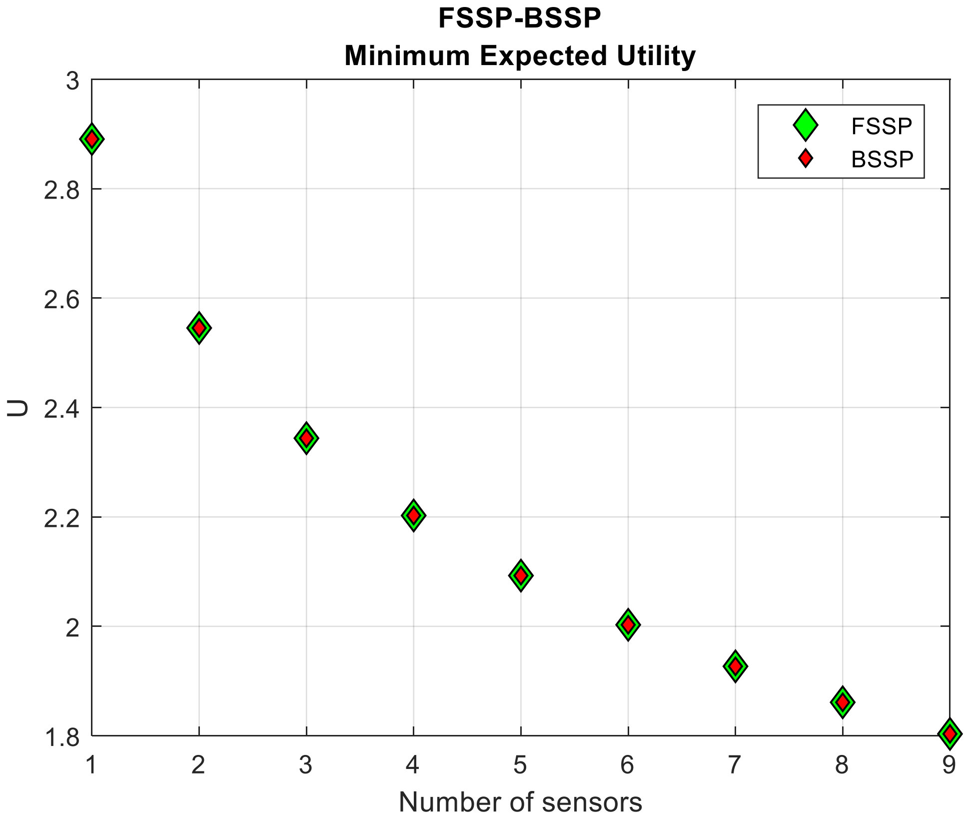

Figure 26 : Utility function values as to sensor number for Pressure in Network 3 with leakage in pipe 33

6. Conclusions and Future Work

A software for simulating the flow in water distribution networks was developed. Results from the software compare very well with the ones obtained by the EPANET software. Results obtained from the software for a number of different networks compare well with those obtained from EPANET. The software can be used for designing water distribution networks, for leakage detection as well as optimal sensor placement. Herein the main use was leakage detection using simulated measurements and optimal sensor placement for leakage detection.

Proceeding to the Detection of Leakage, we could undoubtedly state that the minimization procedure runs perfectly for the constructed software. We were able to create credible measurements and then through the optimization process spot the leakage value and its pipe's location. Moreover we managed to accomplish the optimization process by using the EPANET and we obtained the same results.

After completing the Detection of Leakage chapter we moved on to the Sensitivities. The sensitivities are extremely useful for all the systems whose parts affect each other. We were able to produce these derivatives both analytically and by finite difference. We used these methods in order to validate their accuracy which is incredibly high by examining the results. The Sensitivities are the major pillar of the Optimal Sensor placement so much attention was paid in order to certain of the outcome.

Concerning the Optimal Sensor analysis it is obvious that the FSSP and BSSP offer the same results. By various tests, it became clear that we are able to predict sufficiently the location of the leakage or its proximity area with both flow and pressure sensors. Furthermore it goes without saying that while we increase the number of sensors the Utility function decreases so the system becomes more accurate for Detecting the Leakage.

There are many more scientific topics that may enable the expansion of this study or even the improvement of the current one. As was stated the EPANET use for the detection of leakage may be very credible but if very inefficient due to the large computational time that the program requires. In addition the Detection of Leakage was examined in the case that we have a single leakage. So it is vital to expand this study on detecting multiple leakages. Last but not least, the system is assumed time independent but this isn't the case in real networks whose demands constantly change. Let's keep in mind that we set the foundations for further investigation by developing a reliable software on water distribution networks.

7. References

1. POTTER M. C. AND WIGGERT, M. C. (1997). *MECHANICS OF FLUIDS* (Vol. 2nd EDITION) PRENTICE HALL.

2. Rossman, L. a. (2008). Users manual EPANET 2. *Social Studies of Science*, *38*(4), 483–508. https://doi.org/10.1177/0306312708089715

3. Sarrate, R., Blesa, J., Nejjari, F., & Quevedo, J. (2014). Sensor placement for leak detection and location in water distribution networks. *Water Science and Technology: Water Supply*, *14*(5), 795–803.

4. Papadimitriou, D. I., & Papadimitriou, C. (2015). Optimal Sensor Placement for the Estimation of Turbulence Model Parameters in Cfd. *International Journal for Uncertainty Quantification*, *5*(6), 545–568.

5. Kiij, J. (2011). Darcy Friction Factor Formulae in Turbulent Pipe Flow. *Lunowa∗Fluid Mechanics Paper 110727*, 1–11.

6. ΖΑΝΝΗΣ, Π. (2000). *ΣΧΕΔΙΑΣΜΟΣ, ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ΚΑΙ ΔΙΑΓΝΩΣΗ ΒΛΑΒΩΝ ΣΕ ΔΙΚΤΥΑ ΣΩΛΗΝΩΣΕΩΝ* ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ.

7. Δημήτρης Γ. Παπανίκας. (2010). *ΕΦΑΡΜΟΣΜΕΝΗ ΡΕΥΣΤΟΜΗΧΑΝΙΚΗ* (4th ed.).

8. Appendix

The address for acquiring EPANET software and examine the details of the program is:

- https://www.epa.gov/water-research/epanet

Below, is displayed the modified code that we used in order to obtain the loops in the network. 'Pipes' is the matrix with the node connections and loop_list is a cell array containing all the possible loops. The details of the function are written in the first lines of the code as comments. Indicatively, it must be said that the original function is called run_loops.m and it can be found in following address:

- https://ch.mathworks.com/matlabcentral/fileexchange/10722-count-loops-in-a-graph?s_tid=prof_contriblnk

```
function [loop_list]=looptrialcode(Pipes)

%->run_loops.m file's address:
https://ch.mathworks.com/matlabcentral/fileexchange/10722-
count-loops-in-a-graph?s_tid=prof_contriblnk

%RUN_LOOPS  Counts the number of loops in a network
```

```
% This code counts the number of loops (cycles) in a network
(graph) that
% is composed of nodes and edges. It employs an iterative
algorithm that
% transforms the network into a tree (the ILCA - Iterative
Loop Counting
% Algorithm). This is a "brute force" technique as there are
no known (to
% my knowledge anyway) algorithms for providing a good
estimation.
%
% AUTHOR:   Joseph Kirk,2/2007
% EMAIL:    <jdkirk630@gmail.com>
% USAGE:    >> run_loops;
% NOTES:    Refer to the README and the DETAILS files for more
info
%---------------------------------------------------------------
------
% MODIFIED: Dimitris Katsaros(pregraduate mechanical engineer
UTH)
% EMAIL:    <dimkats007@gamil.com>
% Usage:    >>find net loops automatically without the command
window

%Using run_loops.m file we managed to delete the command
window and import
%the Node Connections through the double matrix Pipes and then
we compute all possible Loops
%which we pass in a cell array matrix called looplist.

%Procedure:
% STEP 1: OBTAIN A NETWORK
         edge_list=Pipes; %Pipes is a matrix (dimensions:
number of pipes x 2)that contains the Node connections
        if isempty(edge_list)
            disp('Error:N is an empty matrix,please check it
again!!!')
        end
        usnet = edge_list2net(edge_list); % format the
edgelist for the loop counting process
        net = sort_net(usnet);

num_nodes = length(net); %number of nodes
num_edges = calc_num_edges(net); %number of edges
disp(['  Net:  Nodes = ' num2str(num_nodes) ',' ' Edges = '
num2str(num_edges)]);

% STEP 2: SETUP (INITIALIZE THE STARTING NODE)
n = get_starting_node(net);        % give the path a nearly
optimal starting node
path = net(n).node;                % initialize the path
current_edge = net(n).edges(1);    % initialize the first edge
loop_list = [];                    % initialize the loop list
```

```
iterations = 0;                     % initialize the number of
algorithm steps

% STEP 3: COUNT LOOPS (SEARCH THE GRAPH USING THE ILCA)
answer2 ={'20'};                                % initial guess of
loop estimation
if ~isempty(answer2)
    num_est_loops = cell2mat(answer2);
else
    disp('Give an initial guess of loop estimation(line 41)')
end
wb = waitbar(0,['Searching Tree for Loops ... ' num2str(0) '
found']);
while (length(path)>1 || ~isempty(current_edge))
    [net,path,current_edge,loop_list] =
iterate_tree(net,path,current_edge,loop_list);
    iterations = iterations+1;

waitbar(length(loop_list)/str2double(num_est_loops),wb,['Searc
hing Tree for Loops ... ' num2str(length(loop_list))
'found']);

end
close(wb);
num_loops = length(loop_list);
disp(['    It took ' num2str(iterations) ' steps to complete
the ILCA']);
disp(['     There are ' num2str(num_loops) ' loops in the
net']);

%--------------------------------------------------------------
-------------
%------- SUBFUNCTIONS ----------------------------------------
-------------
%--------------------------------------------------------------
-------------
%--------------------------------------------------------------
-------------
%--------------------------------------------------------------
-------------
function net = edge_list2net(edge_list)
% PURPOSE:  Transform an edge list into a network structure
% USAGE:    >> net = edge_list2net(edge_list);
% INPUTS:   edge_list  - Nx2 matrix of nodes where each row
represents an edge connection
% OUTPUTS:  net  - network structure containing two fields:
'node' and 'edges'
%                  'node' is the ID of the current node
%                  'edges' is a vector that lists all the
nodes connected to 'node'

net = [];
if isempty(edge_list)
```

```
    return
end
edge_list = abs(round(real(edge_list)));
ne = size(edge_list);
net(1).node = edge_list(1,1); net(1).edges = edge_list(1,2);
net(2).node = edge_list(1,2); net(2).edges = edge_list(1,1);
for idx = 2:ne(1)
    node_exists = 0;
    % if the node is already part of the net, update the list
of edges
    for k = 1:length(net)
        if (edge_list(idx,1) == net(k).node)
            % do not update the edge list if the edge already
exists
            if isempty(find([net(k).edges net(k).node] ==
edge_list(idx,2),1))
                net(k).edges = [net(k).edges
edge_list(idx,2)];
            end
            node_exists = 1;
            break
        end
    end
    % if the node is new, add it to the end of the net along
with the edge
    if ~node_exists
        net(k+1).node = edge_list(idx,1);
        net(k+1).edges = edge_list(idx,2);
    end
    node_exists = 0;
    % if the node is already part of the net, update the list
of edges
    for k = 1:length(net)
        if (edge_list(idx,2) == net(k).node)
            % do not update the edge list if the edge already
exists
            if isempty(find([net(k).edges net(k).node] ==
edge_list(idx,1),1))
                net(k).edges = [net(k).edges
edge_list(idx,1)];
            end
            node_exists = 1;
            break
        end
    end
    % if the node is new, add it to the end of the net along
with the edge
    if ~node_exists
        net(k+1).node = edge_list(idx,2);
        net(k+1).edges = edge_list(idx,1);
    end
end
```

```
%-------------------------------------------------------------
-------------
function net = sort_net(net)
% PURPOSE:  Puts all of the nodes in order from least to
greatest
% USAGE:    >> net = sort_net(net);
% INPUTS:   net  - network structure containing two fields:
'node' and 'edges'
%                  'node' is the ID of the current node
%                  'edges' is a vector that lists all the
nodes connected to 'node'
% OUTPUTS:  net  - sorted network structure containing two
fields: 'node' and 'edges'
%                  'node' is the ID of the current node
%                  'edges' is a vector that lists all the
nodes connected to 'node'

tmp = [];
nodes_list = zeros(1, length(net));
for k = 1:length(net)
    nodes_list(k) = net(k).node;
end
[sorted, order] = sort(nodes_list);
for k = 1:length(net)
    tmp(k).node = net(order(k)).node;
    tmp(k).edges = sort(net(order(k)).edges);
end
net = tmp;

%-------------------------------------------------------------
-------------
function num_edges = calc_num_edges(net)
% PURPOSE:  Calculates the number of edges in an undirected
network
% USAGE:    >> num_edges = calc_num_edges(net);
% INPUTS:   net  - network structure containing two fields:
'node' and 'edges'
%                  'node' is the ID of the current node
%                  'edges' is a vector that lists all the
nodes connected to 'node'
% OUTPUTS:  num_edges  - number of edges in the network

num_edges = 0;
for k = 1:length(net)
    num_edges = num_edges + length(net(k).edges);
end
num_edges = num_edges/2;

%-------------------------------------------------------------
-------------
function n = get_starting_node(net)
% PURPOSE:  Pick the (nearly) optimal starting node
% USAGE:    >> n = get_starting_node(net);
```

```
% INPUTS:   net  - network structure containing two fields:
'node' and 'edges'
%                   'node' is the ID of the current node
%                   'edges' is a vector that lists all the
nodes connected to 'node'
% OUTPUTS:  n  - index to the optimal network starting node

n = 1;
for k = 2:length(net)
    if (length(net(k).edges) > length(net(n).edges))
        n = k;
    end
end

%--------------------------------------------------------------
-------------
function [net,path,current_edge,loop_list] =
iterate_tree(net,path,current_edge,loop_list)
% PURPOSE:  Execute the current iterative step in the loop
counting algorithm
% USAGE:    >> [net,path,current_edge,loop_list] =
iterate_tree(net,path,current_edge,loop_list);
% INPUTS:   net  - network structure containing two fields:
'node' and 'edges'
%                   'node' is the ID of the current node
%                   'edges' is a vector that lists all the
nodes connected to 'node'
%           path  - an ordered vector of node values that are
connected
%           current_edge  - the node ID of the current edge
%           loop_list  - a structure with one field named
'loop' containing a list of all loops found
% OUTPUTS:  net  - same as net input
%           path  - same as path input,potentially modified
%           current_edge  - the node ID of the next edge to be
considered
%           loop_list  - same as loop_list input,potentially
ammended

path_size = length(path);
% DONE - finished searching tree
if (path_size == 1 && isempty(current_edge))
    return
% CURRENT EDGE LIST FINISHED - go up tree
elseif (isempty(current_edge))
    current_edge = get_next_edge(net,path(path_size-
1),path(path_size));
    path(path_size) = [];
% CURRENT EDGE IS THE SAME AS PREVIOUS VERTEX - move to next
edge
elseif (length(path) > 1 && path(path_size-1) == current_edge)
    current_edge =
get_next_edge(net,path(path_size),current_edge);
```

```
% LOOP FOUND!
elseif (check_path4loop(path,current_edge))
    loop = loop2std_form(path,current_edge);
    if ~compare_loop(loop,loop_list)
        loop_list = append_loop_list(loop_list,loop);
    end
    current_edge =
get_next_edge(net,path(path_size),current_edge);
% NO LOOP FOUND - keep going down tree
else
    path = [path current_edge];
    current_edge = get_next_edge(net,path(path_size+1),[]);
end

%--------------------------------------------------------------
-------------
function loop_list = append_loop_list(loop_list,loop)
% PURPOSE:  Adds a loop to the end of a loop_list structure
% USAGE:    >> loop_list = append_loop_list(loop_list,loop);
% INPUTS:   loop_list  - a structure with one field named
'loop' containing
%                        a list of all previously found loops
%           loop  - 1xM vector containing a list of nodes that
make a loop
% OUTPUTS:  loop_list  - the modified loop_list structure

if isempty(loop_list)
    loop_list.loop = loop;
else
    num_loops = length(loop_list);
    loop_list(num_loops+1).loop = loop;
end

%--------------------------------------------------------------
-------------
function status = check_path4loop(path,current_edge)
% PURPOSE:  Check to see if the current edge is in the path
% USAGE:    >> status = check_path4loop(path,current_edge);
% INPUTS:   path  - an ordered vector of node values that are
connected
%           current_edge  - a node connected to the last node
in path
% OUTPUTS:  status  - 1 if a loop has been found,0 otherwise

status = 0;
if find(path == current_edge,1)
    status = 1;
end

%--------------------------------------------------------------
-------------
function status = compare_loop(loop,loop_list)
```

```
% PURPOSE:  Check to see if the loop already exists in the
loop_list
% USAGE:    >> status = compare_loop(loop,loop_list);
% INPUTS:   loop  - 1xM vector containing nodes that are
connected in a loop
%           loop_list  - a structure with one field named
'loop' containing a list
%                        of all previously found loops
% OUTPUTS:  status  - equals 1 if 'loop' already exists,0
otherwise

status = 0;
if isempty(loop_list)
    return
end
for k = 1:length(loop_list)
    m = length(loop_list(k).loop);
    n = length(loop);
    % if the two loops have the same length,check if they are
identical
    if (m == n)
        status = 1;
        for kk = 1:n
            if (loop_list(k).loop(kk) ~= loop(kk))
                status = 0;  % loops are different,move on to
next
                break
            end
        end
        % loops are identical
        if status
            return
        end
    end
end

%---------------------------------------------------------------
-------------
function next_edge =
get_next_edge(net,current_node,current_edge)
% PURPOSE:  Find the next edge of the current node in the
network structure
% USAGE:    >> next_edge =
get_next_edge(net,current_node,current_edge);
% INPUTS:   net  - network structure containing two fields:
'node' and 'edges'
%                  'node' is the ID of the current node
%                  'edges' is a vector that lists all the
nodes connected to 'node'
%           current_node  - the ID of the current node in the
path
%           current_edge  - the node ID of the current edge
```

```
% OUTPUTS:  next_edge  - the node ID of the next edge in the
edges list for the current node

next_edge = [];
for k = 1:length(net)
    if (current_node == net(k).node)
        if isempty(current_edge)    % start with the first
edge of the node
            next_edge = net(k).edges(1);
        else    % get the next edge in the list,if there is
one
            kk = find(net(k).edges == current_edge);
            if kk < length(net(k).edges)
                next_edge = net(k).edges(kk+1);
            end
        end
        return
    end
end
%-------------------------------------------------------------
-------------
function loop = loop2std_form(path,current_edge)
% PURPOSE:  Take a loop found in the path and return the loop
vector in *standard form*
% USAGE:    >> loop = loop2std_form(path,current_edge);
% INPUTS:   path  - an ordered vector of node values that are
connected
%           current_edge  - the node ID of the current edge
% OUTPUTS:  loop  - 1xM vector of standard form loop,where M
is the length of the loop
% NOTES:    Standard form is defined as having the smallest
node ID at the front
%           of the list,and the smaller of the two neighbors
listed second

ii = find(path == current_edge);
% get the loop from the path
loopy = path(ii:end);
n = length(loopy);
jj = find(loopy == min(loopy));
% order the loop with the smallest value first
loop = loopy([(jj:n) (1:jj-1)]);
% order the rest of the loop with the smaller of the two
neighbors second
if loop(2) > loop(n)
    loop = [loop(1) fliplr(loop(2:n))];
end
```