

Raúl Siverio Quintero

*Programación óptima de los sistemas de  
distribución de agua*

Optimal Scheduling of Water Distribution  
Systems

Trabajo Fin de Grado  
Grado en Matemáticas  
La Laguna, Marzo de 2023

DIRIGIDO POR  
*Hipólito Hernández Pérez*

*Hipólito Hernández Pérez*

*Departamento de Matemáticas,  
Estadística e Investigación  
Operativa*

*Universidad de La Laguna  
38200 La Laguna, Tenerife*

---

## Agradecimientos

A mi tutor Hipólito por dirigir este trabajo y porque sin su ayuda y comprensión este proyecto no hubiera sido posible. A mi compañera de vida y a mi padre y madre apoyarme en todo momento.

Raúl Siverio Quintero  
La Laguna, 9 de marzo de 2023



---

## Resumen · Abstract

### *Resumen*

---

*En este trabajo se presenta la modelización del problema de optimización de los sistemas de distribución de agua, donde la demanda de los clientes debe ser satisfecha tratando de minimizar los costos operativos. Basándonos en artículo “Optimal Scheduling of Water Distribution Systems”, relajamos su modelo no lineal, generando así un modelo alternativo que simplifica el problema abordado. Implementamos en PYTHON el modelo alternativo y probamos su eficiencia con distintas redes de distribución de agua.*

### *Abstract*

---

*This paper presents the modeling of the optimization problem of water distribution systems, where customer demand must be satisfied trying to minimize operating costs. Based on the article “Optimal programming of water distribution systems”, we relax its non-linear model, generating an alternative model that simplifies the problem addressed. We implemented the alternative model in PYTHON and tested its efficiency with different water distribution networks.*



---

# Contenido

<b>Agradecimientos</b> .....	III
<b>Resumen/Abstract</b> .....	V
<b>Introducción</b> .....	IX
<b>1. Herramientas matemáticas e hidráulicas</b> .....	1
1.1. Teoría de Grafos .....	1
1.2. Redes de distribución de agua y el software EPANET .....	2
1.3. Introducción a la Optimización Combinatoria .....	3
<b>2. Planteamiento y modelización del problema</b> .....	5
2.1. Modelo no lineal desarrollado en [5] .....	6
2.2. Modelo alternativo lineal .....	13
2.3. Modelo alternativo lineal más ajustado .....	16
<b>3. Resultados computacionales</b> .....	19
<b>4. Conclusiones</b> .....	33
<b>A. Código en PYTHON</b> .....	35
A.1. Funciones auxiliares .....	35

A.2. Preproceso .....	35
A.3. Modelo Alternativo .....	39
<b>Bibliografía</b> .....	43
<b>Poster</b> .....	45



---

## Introducción

Las redes de distribución de agua son una parte vital de cualquier sistema de suministro de agua potable en todo el mundo. Estas redes son un conjunto de tuberías, válvulas y otros elementos diseñados para transportar agua desde una fuente de suministro, como un embalse o una planta de tratamiento de agua, hasta los hogares y negocios de una comunidad. Las redes de distribución de agua son una infraestructura crítica para el bienestar de las personas y la economía de las ciudades, por lo que su optimización es crucial.

Una red de distribución de agua bien diseñada y optimizada puede garantizar que el agua potable llegue a su destino de manera segura y eficiente. El objetivo principal de una red de distribución de agua es satisfacer la demanda de agua de una comunidad en particular, ya sea residencial, comercial o industrial.

La optimización de una red de distribución de agua puede mejorar la eficiencia del suministro de agua, reducir los costos operativos, minimizar las pérdidas de agua y mejorar su calidad.



## Herramientas matemáticas e hidráulicas

En este capítulo se definirán una serie de elementos fundamentales relacionados con la teoría de grafos, la topología de una red de distribución de agua, el software EPANET y la optimización combinatoria necesarios para comprender el desarrollo posterior de nuestro trabajo. No obstante, si el lector desea indagar en alguno de estos temas, puede hacer uso del libro de Bondy y Murty [2] para teoría de grafos, el manual del usuario de EPANET [1], el libro de Salazar [4] para programación matemática y en particular optimización combinatoria, y, el artículo de D'Ambrosio et al. [3] para ver como se puede aplicar la programación matemática en la optimización de redes de distribución de aguas.

### 1.1. Teoría de Grafos

En este primer apartado se introducirán una serie de definiciones de elementos de teoría de grafos, que serán útiles en el desarrollo de nuestro trabajo.

**Definición 1.** Sea el conjunto  $G = G(V, A)$ , decimos que es un **grafo dirigido** si está compuesto por un conjunto de vértices  $V$  y un conjunto de arcos  $A \subseteq \{(i, j) : i, j \in V, i \neq j\}$ , en esta definición se interpreta que el arco  $(i, j)$  es distinto del arco  $(j, i)$ .

**Definición 2.** Sea el conjunto  $G = G(V, E)$ , decimos que es un **grafo no dirigido** si está compuesto por un conjunto de vértices  $V$  y un conjunto de aristas  $E \subseteq \{[i, j] : i, j \in V, i \neq j\}$ , donde la arista  $[i, j]$  se entiende que es la misma que  $[j, i]$ .

**Definición 3.** Una **cadena** es una secuencia de arcos con un vértice en común cada dos arcos consecutivos.

**Definición 4.** Se llama **camino** a una cadena tal que el vértice de destino coincide con el vértice de origen de cada dos arcos consecutivos. Para el caso de

*grafos no dirigidos, un **camino** es una secuencia de aristas donde cada par de aristas son ambas incidentes en el mismo nodo.*

*Decimos que un **camino es simple**, si sus nodos intermedios son distintos.*

**Definición 5.** *Decimos que un grafo es **conexo** si todos sus vértices están conectados por un camino.*

## 1.2. Redes de distribución de agua y el software EPANET

Las redes de distribución de agua son sistemas complejos de tuberías, válvulas y otros componentes, diseñados para transportar agua de un punto a otro en una ciudad o área geográfica determinada. Estas redes pueden ser modeladas como grafos, donde los nodos representan los puntos de entrada y salida de agua (embalses, tanques o puntos de demanda de agua como hogares y empresas) y las aristas representan las tuberías y bombas de agua que conectan los nodos y que permiten transportar el agua a cualquier lugar de la red que lo requiera.

A lo largo de este trabajo mencionaremos en numerosas ocasiones al software EPANET. Es un software de simulación hidráulica de redes de distribución de agua, fue desarrollado por la Agencia de Protección Ambiental de los Estados Unidos (EPA, por sus siglas en inglés) y se lanzó originalmente en 1993. Este programa se utiliza para modelar redes y calcular el flujo y la presión del agua en cada punto de la red. Es una herramienta valiosa para ingenieros y diseñadores de sistemas de distribución de agua, ya que permite predecir el comportamiento del sistema bajo diferentes condiciones y evaluar el rendimiento y la eficiencia de la red.

El software permite a los usuarios ingresar la topología de una red de distribución de agua como un fichero de texto, incluyendo la ubicación y las características de las tuberías, válvulas, bombas y nodos, así como los patrones de demanda de agua en cada punto de la red. EPANET utiliza algoritmos hidráulicos avanzados para simular el funcionamiento de la red bajo las características introducidas, permitiendo al usuario observar en cada momento el caudal y la presión del agua en cada punto de la red, así como la energía consumida por cada una de las bombas. En caso de introducir una red cuyas características conduzcan a futuros problemas como roturas de tuberías, fallos en las bombas o demanda insatisfecha, el software nos lo hará saber, convirtiéndose así en una herramienta fundamental no solo para el desarrollo de futuras redes sino también para comprender el comportamiento de las redes bajo diferentes condiciones y optimizar la gestión del sistema para mejorar la eficiencia y la calidad del agua.

EPANET es un código fuente escrito en C y se ha desarrollado entorno a él una API en PYTHON. La API de PYTHON de EPANET (epanettools) es un conjunto de funciones y métodos que permiten interactuar con el software utilizando el lenguaje de programación PYTHON. La API se basa en un archivo de extensión compartida (DLL) que contiene las funciones de EPANET escritas en C.

La API de PYTHON permite realizar operaciones como la carga y ejecución de modelos de red de distribución de agua, la obtención de información sobre los nodos y las tuberías de la red y la simulación del flujo de agua y la calidad del agua. El uso de esta API puede ser útil para desarrolladores que desean integrar la funcionalidad de EPANET en sus propias aplicaciones de software, o para analistas de redes de agua que desean automatizar y/o optimizar la simulación y el análisis de redes utilizando PYTHON.

### 1.3. Introducción a la Optimización Combinatoria

La programación matemática es la rama de la investigación operativa cuyo objetivo es resolver problemas del tipo:

$$\text{mín}\{f(x) : x \in S\}$$

O bien,

$$\text{máx}\{f(x) : x \in S\}$$

Donde  $S \subseteq \mathbb{R}^n$  es lo que llamamos región factible, ya que cada uno de sus elementos será una solución de la función objetivo  $f : S \rightarrow \mathbb{R}$ . Esto es lo que llamamos un problema de optimización.

La programación matemática se puede dividir en dos grandes bloques, dependiendo de si la función  $f$  es lineal o no y de si  $S$  es un conjunto de desigualdades lineales o no. Así, tenemos que la programación matemática puede ser lineal o no lineal.

A su vez, la programación lineal puede ser continua o entera, dependiendo de si existe o no la condición de que alguna de las variables sea entera. En el caso de la programación lineal entera encontramos problemas que pueden llegar a ser muy difíciles de resolver.

Por otro lado, la optimización combinatoria es una rama de la programación matemática, que trata de resolver problemas en los que el número de soluciones factibles es finito. De esta forma, obtenemos la siguiente definición:

**Definición 6.** *Un problema de optimización combinatoria se puede formular como  $\min\{f(x) : x \in S\}$  donde  $|S| < \infty$ .*

En muchas ocasiones, aunque el número de soluciones es finito (es decir,  $|S| < \infty$ ), estamos hablando de un número tan grande que es necesario desarrollar algoritmos matemáticos capaces de determinar al menos alguna de ellas.

Estos algoritmos a su vez se dividen en polinomiales y no polinomiales. Sin entrar en conceptos de teoría de complejidad computacional podemos decir, a groso modo, que un problema es polinomial si existe un algoritmo polinomial que determina una solución óptima, estos problemas se consideran “fáciles” de resolver. En caso contrario, estaremos hablando de un problema no polinomial y será un problema “difícil” de resolver.

## Planteamiento y modelización del problema

En este capítulo trataremos los sistema de distribución de agua *Water Distribution System*, *WDS* y su optimización. Para explicar con claridad el desarrollo y las metas de este Trabajo de Fin de Grado resumiremos parte del artículo científico “Optimal Scheduling of Water Distribution Systems” de Singh y Kekatos [5]. Nuestro objetivo principal será minimizar los costes de la electricidad exigiendo que la demanda de todos los consumidores de agua de nuestra red sea satisfecha. La tarifa dinámica del precio de la electricidad conlleva a que la operación de la distribución de agua en cualquier tipo de red sea variable. Las bombas que mueven el agua de los depósitos a los tanques y a los consumidores, pueden servir como alternativas de almacenamiento de energía si se opera de la forma correcta. Por otra parte es importante considerar que el correcto funcionamiento de los sistemas de distribución de agua está guiado por las leyes de la hidráulica de acuerdo a las cuales la presión a lo largo de una tubería cae proporcionalmente al cuadrado de su caudal de agua, añadiendo así restricciones no lineales a nuestro modelo y complicando por tanto su resolución.

En Singh y Kekatos [5], el problema de sistemas de distribución de agua, está formulado como un problema entero-mixto no convexo que incorpora restricciones de caudal y presión necesarias para el correcto funcionamiento de las bombas de velocidad fija, tanques, embalses y tuberías. Las restricciones hidráulicas son relajadas a restricciones de un cono de segundo orden. Para restaurar la factibilidad de las restricciones no convexas originales, se añade un término de penalización a la función objetivo del problema relajado. Este nuevo problema se puede resolver como un problema entero-mixto de cono de segundo orden.

Posteriormente mostramos un modelo lineal desarrollado por nosotros, en el que se estudian diferentes escenarios que son resueltos por EPANET. Es más, en la última sección de este capítulo se muestra una extensión de este modelo.

## 2.1. Modelo no lineal desarrollado en [5]

Un sistema de distribución de agua puede ser representado como un grafo dirigido  $G = (V, A)$ , donde  $V$  representa al conjunto de embalses, tanques, uniones de tuberías y puntos de demanda. Los embalses son la principal fuente de agua y constituyen el subconjunto  $V_E \subset V$ . Del mismo modo, los nodos que alojan los tanques comprenden el subconjunto  $V_T \subset V$ . Los nodos en el conjunto  $V_T \cup V_E$  serán reservas y no sirven directamente a los consumidores de agua. Sin pérdida de generalidad, en el caso de que tengamos un consumidor potencial en un nodo  $i \in V_T \cup V_E$  este se puede conectar a un nodo auxiliar conectado al nodo  $i$  a través de una tubería sin pérdidas. Consideraremos también al conjunto de nodos de demanda  $V_D \subset V$  y al conjunto de nodos que representan uniones  $V_U \subset V$ .

Sea  $d_i^t$  la cantidad de agua que el nodo  $i$  proporciona a la red durante el periodo  $t$ . Tenemos que, para embalses (que solo proporcionan agua a la red y no reciben nada)  $d_i^t \geq 0, \forall i \in V_E$ ; para los nodos de demanda con consumidores asociados  $d_i^t \leq 0, \forall i \in V_D$ ; en las uniones  $d_i^t = 0, \forall i \in V_U$ ; y por último, los tanques pueden estar vaciando o llenando por lo que  $d_i^t$  puede tomar ambos signos.

Cabe destacar que a pesar de considerarlo un grafo dirigido, las tuberías en determinadas condiciones pueden ser bidireccionales, tomando en ocasiones caudales negativos que simplemente expresan sentido contrario del agua al pre-establecido.

Los elementos del conjunto de arcos  $A$  de  $G$  representan las tuberías y bombas de agua. Todos los arcos tienen una dirección asignada. El arco  $(i, j) \in A$  modeliza la tubería que conecta los nodos  $i$  y  $j$ . Si  $(i, j) \in A$ , entonces  $(j, i) \notin A$ . El caudal o flujo de agua que pasa por la tubería  $(i, j)$  en el periodo  $t$  se denota por  $d_{ij}^t$ . Si el agua fluye desde el nodo  $i$  al nodo  $j$  en el instante  $t$ ,  $d_{ij}^t \geq 0$ , en otro caso el flujo será negativo. Por la conservación del flujo tenemos que:

$$d_i^t = \sum_{j:(i,j) \in A} d_{ij}^t - \sum_{j:(j,i) \in A} d_{ji}^t, \quad \forall i, t \quad (2.1)$$

Los sistemas de distribución de agua aparte de la cantidad de agua proporcionada y del flujo están gobernados por la presión. La presión del agua es normalmente sustituida por la carga de presión (en inglés *Pressuare head*), que se mide en metros y está linealmente relacionada con la presión del agua. Para entrar un poco más en detalle, una carga de presión de  $h$  metros corresponde con una presión del agua de  $hp\tilde{g}$ , donde  $p$  es la densidad del agua en  $kg/m^3$  que es una constante conocida y  $\tilde{g}$  es la aceleración a causa de la gravedad en  $m/s^2$ .



La carga de presión en un nodo es igual a su elevación geográfica más la altura de la columna de agua presente en el nodo.

La carga de presión en el nodo  $i$  en el tiempo  $t$  será denotada por  $h_i^t$ . Para el correcto funcionamiento de las redes de agua se requiere de una presión manométrica mínima en todos los nodos (la presión manométrica o presión relativa es la diferencia entre la presión absoluta o real y la presión atmosférica, se aplica tan solo en aquellos casos en los que la presión es superior a la presión atmosférica). Añadiendo este mínimo valor de presión manométrica a la elevación geográfica conocida de cada nodo  $i \in V$  obtenemos una cota inferior de la presión:

$$h_i^t \geq \underline{h}_i^t \quad (2.2)$$

El movimiento del agua en una tubería se traduce en una caída cuadrática de la presión. La caída de la carga de presión a lo largo de la tubería  $(i, j) \in A$  está descrita por la ecuación de *Darcy-Weisbach*:

$$h_i^t - h_j^t = c_{ij} \text{sign}(d_{ij}^t) (d_{ij}^t)^2 \quad (2.3)$$

donde el coeficiente  $c_{ij} := \frac{l_{ij} f_{ij}}{4\pi^2 r_{ij}^5 \bar{g}}$  depende de la longitud de la tubería  $l_{ij}$ ; de su radio interior  $r_{ij}$ ; y del factor de fricción de Darcy  $f_{ij}$ . Aunque el factor  $f_{ij}$  realmente depende del flujo  $d_{ij}$  de una forma no lineal continua, este es normalmente aproximado como una constante. La función signo se asegura de que la presión caiga en la dirección del flujo del agua. Para eliminar la discontinuidad de la función signo, se propone un modelo entero-mixto usando el truco de la  $M$  grande para la caída de presión en la tubería  $(i, j) \in A$ , usando variables binarias  $x_{ij}^t$  para  $t = 1, \dots, T$ , tal que  $x_{ij}^t = 1$  si hay flujo de agua de  $i$  a  $j$ ,  $x_{ij}^t = 0$  si hay flujo de agua de  $j$  a  $i$  y el valor que tome la variable es indiferente cuando no hay flujo de agua. De tal forma que la ecuación de la caída de presión (2.3) puede ser expresada de forma equivalente a partir de las siguientes restricciones:

$$-M(1 - x_{ij}^t) \leq d_{ij}^t \leq Mx_{ij}^t \quad (2.4)$$

$$-M(1 - x_{ij}^t) \leq h_i^t - h_j^t - c_{ij}(d_{ij}^t)^2 \leq M(1 - x_{ij}^t) \quad (2.5)$$

$$-Mx_{ij}^t \leq h_i^t - h_j^t + c_{ij}(d_{ij}^t)^2 \leq Mx_{ij}^t \quad (2.6)$$

$$x_{ij}^t \in \{0, 1\} \quad (2.7)$$

para una  $M > 0$ . Si  $x_{ij}^t = 1$ , la restricción (2.4) garantiza que  $d_{ij}^t \geq 0$ ; la restricción (2.5) pasa a ser una igualdad,  $h_i^t - h_j^t = c_{ij}(d_{ij}^t)^2$ ; y la restricción (2.6) se vuelve trivial. Por el contrario si  $x_{ij}^t = 0$ , en (2.4) vemos que el flujo cambia su dirección,  $d_{ij}^t \leq 0$ ; la restricción (2.6) pasa a ser una igualdad,  $h_i^t - h_j^t = -c_{ij}(d_{ij}^t)^2$ ; y (2.5) se vuelve trivial. Cabe destacar que si  $d_{ij}^t = 0$ , la variable

$x_{ij}^t$  se convierte en inconsecuente, y observando (2.5) y (2.6) concluiremos que  $h_i^t = h_j^t$  para cualquier valor de  $x_{ij}^t$ .

Para mantener la presión en los nodos en niveles deseables, las empresas encargadas de la distribución del agua usan bombas instaladas en determinadas tuberías para aumentar la presión. A una tubería de agua equipada con una bomba se puede modelar como una bomba ideal (sin pérdidas) seguida de una tubería con caída de la carga de presión dictada por (2.4)–(2.7). El subconjunto de arcos que representan las bombas ideales es denotado por  $A_b \subset A$ . Los arcos restantes comprende el conjunto  $\bar{A}_b := A \setminus A_b$  y representan a las tuberías con pérdidas, que están sujetas a las restricciones (2.4)–(2.7). A partir de ahora cualquier referencia a la bomba  $(i, j)$  se referirá al segmento ideal de esta.

Si la bomba  $(i, j) \in A_b$  está activa durante el periodo  $t$ , su flujo está obligado a estar en el rango  $\underline{d}_{ij} \leq d_{ij}^t \leq \bar{d}_{ij}$  con  $\underline{d}_{ij} \geq 0$  debido a limitaciones físicas de la propia red. La bomba  $(i, j)$  añade una presión denotada por  $g_{ij}^t \geq 0$  por lo que :

$$h_j^t - h_i^t = g_{ij}^t \quad (2.8)$$

La presión añadida  $g_{ij}^t$  depende de la velocidad de la bomba y del flujo del agua. La dependencia de  $g_{ij}^t$  sobre el flujo de agua es bastante débil y puede ser ignorada sin una pérdida significativa de precisión. De esta forma, para una bomba de velocidad constante, la presión que dicha bomba añade a la tubería  $g_{ij}$  es constante cuando la bomba está activada, y cero en el caso contrario. En ocasiones, cuando una bomba no está funcionando, el agua puede fluir libremente en cualquier dirección a lo largo de la tubería mediante una válvula de derivación o *bypass* conectada en paralelo a la bomba y sin incurrir ninguna diferencia de presión. (Las válvulas de derivación de presión sirven para controlar la presión en un sistema al desviar una parte del flujo. Por lo general, desvían el fluido desde la salida de una bomba hasta el depósito). El funcionamiento de una bomba junto con su válvula *textitbypass* se puede modelar usando el truco de la  $M$  grande a través del modelo entero-mixto,  $\forall (i, j) \in A_b$ , sea  $\tilde{d}_{ij}^t$  el flujo a través de la bomba  $(i, j)$  a tiempo  $t$ :

$$h_i^t - h_j^t = -g_{ij}x_{ij}^t \quad (2.9)$$

$$-M(1 - x_{ij}^t) \leq d_{ij}^t - \tilde{d}_{ij}^t \leq M(1 - x_{ij}^t) \quad (2.10)$$

$$\underline{d}_{ij}x_{ij}^t \leq \tilde{d}_{ij}^t \leq \bar{d}_{ij}x_{ij}^t \quad (2.11)$$

$$x_{ij}^t \in \{0, 1\} \quad (2.12)$$

La variable binaria  $x_{ij}^t$  indica si la bomba está en funcionamiento o no en el tiempo  $t$ . Cuando la bomba está activa ( $x_{ij}^t = 1$ ), la restricción (2.9) implica

(2.8); en otro caso ( $x_{ij}^t = 0$ ), esta restricción verifica que  $h_i^t = h_j^t$ . Para  $x_{ij}^t = 1$ , las restricciones (2.10)-(2.11) implican que  $d_{ij}^t = \tilde{d}_{ij}^t$  y el flujo de agua en la bomba se mantiene entre los límites positivos  $[\underline{d}_{ij}, \bar{d}_{ij}]$ . Para  $x_{ij}^t = 0$ , la variable  $\tilde{d}_{ij}^t$  siguiendo la restricción (2.11) se establece en cero y  $d_{ij}^t$  representa el agua que fluye a través de la válvula de derivación de la bomba. Debemos tener en cuenta que solo consideraremos bombas de velocidad fija y no bombas de velocidad variable por simplicidad.

La presión en un embalse puede ser asumida como constante a lo largo de días o semanas. Considerando el embalse  $i \in V_E$ , denotaremos su presión constante como  $\bar{h}_i$ . Para extraer el agua de este embalse, su presión nodal  $h_i^t$  debe ser menor que la carga de presión constante  $\bar{h}_i$  del embalse. Esto se cumple a través de las restricciones:

$$0 \leq d_i^t \leq M\alpha_i^t \quad (2.13)$$

$$h_i^t \leq \bar{h}_i + P(1 - \alpha_i^t) \quad (2.14)$$

$$\alpha_i^t \in \{0, 1\} \quad (2.15)$$

para todo  $i \in V_E$  y para todo tiempo  $t = 1, \dots, T$ . La variable binaria  $\alpha_i^t$  indica si se extrae agua del embalse  $i$  en el instante de tiempo  $t$ . Si  $\alpha_i^t = 1$ , el embalse  $i$  está conectado a nuestro sistema de distribución de agua y las restricciones (2.13)–(2.15) nos aseguran que  $d_i^t \geq 0$  y  $h_i^t \leq \bar{h}_i$ . En cambio, cuando  $\alpha_i^t = 0$ , el embalse se encuentra desconectado de la red  $d_i^t = 0$  y por lo tanto la restricción (2.14) se verifica de forma trivial.

En contraposición a los embalses, el volumen de agua en los tanques varía significativamente a lo largo del día. Las variaciones en el volumen de agua se traducen en variaciones en el nivel de agua lo que provoca a su vez variaciones de presión en el fondo del tanque. Para modelizar el funcionamiento de los tanques, denotaremos por  $l_i^t$  al nivel de agua del tanque  $i \in V_T$  al final del periodo  $t$ . Para ser coherentes con la carga de presión, el nivel de agua  $l_i^t$  incluye la elevación geográfica del tanque  $i$ . Si  $\delta$  es la duración de un período de control y  $B_i$  es el área de la base del tanque  $i$  (suponiendo que todos los tanques considerados son cilíndrico), el nivel del agua en el tanque  $i$  satisface que :

$$l_i^t = l_i^{t-1} - \frac{d_i^t \delta}{B_i} \quad (2.16)$$

Debido al volumen finito de los tanques, el nivel de agua de un tanque  $i$ , está restringido en todo instante  $t$  por:

$$l_i \leq l_i^t \leq \bar{l}_i \quad (2.17)$$

Por lo general, el intercambio de agua de los tanques se mantiene a cero durante todo el período de funcionamiento, es decir, el nivel del agua del tanque es igual al inicio que al final del período de funcionamiento:

$$l_i^0 = l_i^T \quad (2.18)$$

Cabe destacar que en una red real la condición (2.18) es muy restrictiva, puesto que a la hora de trasladar el modelo a la realidad es casi imposible llegar a una igualdad absoluta en los niveles de agua, luego a nivel computacional lo óptimo sería dar un intervalo para estos niveles iniciales y finales de agua para tener así una región factible un más amplia.

Cada tanque tiene dos tuberías independientes una para llenarlo y otra para vaciarlo. El tubo de llenado o entrada está conectado cerca de la parte superior del tanque, mientras que el tubo de vaciado o salida se conecta por la parte inferior. Las dos tuberías están controladas por dos válvulas independientes. La presión de salida de las válvulas puede ser igual o menor que la presión de entrada. Por lo tanto cuando el tanque  $i$  está siendo llenado con agua en el instante de tiempo  $t$ , debe verificar que  $h_i^t \geq \bar{l}_i$ . Por el contrario, cuando el flujo de agua se dirige hacia fuera del tanque, verificará que  $h_i^t \leq l_i^t$ . Al cerrar tanto el válvulas de entrada y salida, la presión  $h_i^t$  en el nodo  $i$  se desvincula de la presión del fondo del tanque,  $l_i^t$ .

Para tratar de modelizar el funcionamiento del tanque ya mencionado, definimos dos variables binarias  $(\alpha_i^t, \beta_i^t)$  y una variable continua auxiliar  $\tilde{h}_i^t$ . El funcionamiento del tanque  $i$  en el instante  $t$  está descrito por las restricciones siguientes:

$$-M(1 - \alpha_i^t) \leq \tilde{h}_i^t - h_i^t \leq M(1 - \alpha_i^t) \quad (2.19)$$

$$-M\alpha_i^t \leq d_i^t \leq M\alpha_i^t \quad (2.20)$$

$$-M\beta_i^t \leq d_i^t \leq M(1 - \beta_i^t) \quad (2.21)$$

$$\bar{l}_i - M(1 - \beta_i^t) \leq \tilde{h}_i^t \leq l_i^t + M\beta_i^t \quad (2.22)$$

$$\alpha_i^t, \beta_i^t \in \{0, 1\} \quad (2.23)$$

Al igual que en el caso de los embalses la variable  $\alpha_i^t$  nos indica si el tanque  $i$  está conectado a la red en el tiempo  $t$ ; por otra parte, la variable  $\beta_i^t$  nos indica si el tanque está llenando o vaciando. Cuando el tanque está conectado ( $\alpha_i^t = 1$ ), la restricción (2.19) sostiene  $\tilde{h}_i^t = h_i^t$  y (2.20) se hace trivial. Si además el tanque está llenándose de agua ( $\beta_i^t = 1$ ), entonces  $d_i^t \leq 0$  por (2.21) y  $\tilde{h}_i^t = h_i^t \geq \bar{l}_i$  por (2.22). Si el tanque está conectado y vaciando ( $\alpha_i^t = 1, \beta_i^t = 0$ ), entonces  $d_i^t \geq 0$  por (2.21) y  $\tilde{h}_i^t = h_i^t \leq l_i^t$  por (2.22). Cuando el tanque no está conectado

( $\alpha_i^t = 0$ ), la restricción (2.20) exige que  $d_i^t = 0$ , por lo tanto la presión del tanque en este caso no está relacionada con la presión de la red y los valores tanto de  $\beta_i^t$  como de  $\tilde{h}_i^t$  son inconsecuentes.

Con precios dinámicos de la energía, el objetivo principal es minimizar el costo de la electricidad consumida por las bombas de agua. Tras definir todas las restricciones de red ahora las recopilaremos y definiremos el problema de optimización de los sistemas de distribución de agua .

La energía consumida por la bomba  $(i, j) \in A_b$  durante el periodo  $t$  en vatios viene dada por el producto de la diferencia de presión inducida  $g_{ij}$  medida en pascales por el flujo de agua  $\tilde{d}_{ij}^t$  en  $m^3/sec$ . Si la eficiencia energética total de la bomba es  $\eta_{ij}$ , su consumo de energía eléctrica  $\frac{\delta p \tilde{g} g_{ij}}{\eta_{ij}} \tilde{d}_{ij}^t$  a lo largo del periodo  $t$  de duración  $\delta$ . Para las bombas de velocidad fija consideradas aquí, como ya mencionamos con anterioridad, la presión ganada  $g_{ij}$  es constante y podemos definir el coeficiente de consumo de electricidad como:

$$c_{ij} := \frac{\delta p \tilde{g} g_{ij}}{\eta_{ij}} \quad \forall (i, j) \in A_b$$

Dado el nivel inicial de agua en los tanques  $\{l_i^0\}_{i \in V_T}$ , las demandas de agua en los nodos de consumo  $\{d_i^t\}_{i \in V_D}$ , los precios de la electricidad  $\{\pi_t\}_{t=1}^T$  y parámetros de red (capacidades de tanques, dimensiones de la tuberías, ganancias de presión de la bomba y presión mínima requerida, alturas de los tanques); la tarea de este modelo es minimizar el costo de la electricidad para hacer funcionar las bombas mientras la demanda de los consumidores es satisfecha y respetando las limitaciones de las redes de distribución de agua. Concretamente, el coste de bombear el agua puede ser formulado como.

$$f(\tilde{d}) := \sum_{t=1}^T \sum_{(i,j) \in A_b} c_{ij} \pi_t \tilde{d}_{ij}^t \quad (2.24)$$

donde el vector  $\tilde{d}$  contiene los flujos de agua  $\{\tilde{d}_{ij}^t\}_t$  de todas las bombas  $(i, j) \in A_b$  y para todos los tiempos  $t$ . Para simplificar la presentación, el precio de la electricidad  $\pi_t$  se asume invariable a través del sistema de distribución de agua para todo  $t$ . El problema en cuestión se puede plantear como la minimización :

$$\begin{aligned} & \text{mín} && f(\tilde{d}) \\ & \text{sujeto a:} && (2.1), (2.2), (2.4) - (2.7), (2.9) - (2.23). \end{aligned} \quad (P1)$$

El problema (P1) envuelve a las variables continuas  $\{h_i^t, d_i^t, d_{ij}^t, \tilde{h}_i^t, \tilde{d}_{ij}^t\}$  y a las variables binarias  $\{x_{ij}^t, \alpha_i^t, \beta_i^t\}, \forall i, j \in V$ . Para bombas de velocidad fija , el

costo en (P1) es lineal. Aunque la mayoría de las restricciones son lineales gracias al truco de la constante grande, las restricciones (2.5)–(2.6) que modelizan la caída de la presión, son no lineales. De hecho, esas restricciones contienen dos desigualdades cuadráticas, una convexa y otra no convexa. Para obtener soluciones abordables para nuestros sistemas de distribución de agua, procedemos a relajar la restricción no convexa y derivar así en un problema entero-mixto que será convexo con respecto a las variables continuas.

Relajación convexa del problema. La caída de presión a través de una tubería con pérdidas  $(i, j) \in A$  depende de su flujo de agua  $d_{ij}^t$ , relación expresada en (2.3), que puede ser relajada a una desigualdad convexa como :

- $h_i^t - h_j^t \geq c_{ij}(d_{ij}^t)^2$  for  $d_{ij}^t \geq 0$ ;
- $h_j^t - h_i^t \geq c_{ij}(d_{ij}^t)^2$  for  $d_{ij}^t \leq 0$ .

Dado que el signo de  $d_{ij}^t$  es controlado por la variable binaria  $x_{ij}^t$ , la relajación en (2.4)–(2.7) se puede realizar alternativamente de la siguiente forma:

$$-M(1 - x_{ij}^t) \leq d_{ij}^t \leq Mx_{ij}^t \quad (2.25)$$

$$-M(1 - x_{ij}^t) \leq h_i^t - h_j^t - c_{ij}(d_{ij}^t)^2 \quad (2.26)$$

$$h_i^t - h_j^t + c_{ij}(d_{ij}^t)^2 \geq Mx_{ij}^t \quad (2.27)$$

Comparando (2.4)–(2.7) con (2.25)–(2.27), la desigualdad hacia la derecha de (2.5) y la desigualdad hacia la izquierda de (2.6) han sido eliminadas en (2.25)–(2.27). Esas son exactamente las restricciones no convexas. Reemplazando las restricciones (2.4)–(2.7) por (2.25)–(2.27), tenemos el problema relajado:

$$\begin{aligned} & \text{mín} && f(\tilde{d}) \\ & \text{sujeto a:} && (2.1), (2.2), (2.9) - (2.23), (2.25) - (2.27). \end{aligned} \quad (P2)$$

El problema (P2) es convexo con respecto a las variables continuas, por lo que podrá ser manejado por solucionadores entero-mixtos existentes. Al ser una relajación de (P1), el valor óptimo de (P2) sirve como límite inferior para el valor óptimo de (P1). Si los minimizadores de (P2) satisfacen (2.26) ó (2.27) con una igualdad  $\forall (i, j) \in \bar{A}_b$ , la relajación se considera exacta. En este caso, los minimizadores de (P2) coinciden con los minimizadores de (P1). No obstante, en otros casos la relajación no es necesariamente exacta. En el resto del artículo se dan condiciones necesarias y suficientes para relacionar las soluciones de los problemas (P1) y (P2), dando lugar a un nuevo problema (P3), pero no entraremos más en detalle.

## 2.2. Modelo alternativo lineal

Con el objetivo de facilitar la resolución del problema y optimizar la velocidad de ejecución de este, hemos desarrollado en este trabajo un nuevo modelo que hemos llamado *modelo alternativo lineal*. Para este modelo usaremos sobre todo los parámetros ya vistos en el Modelo 1, aún así los expondré a continuación para mayor claridad.

$$A_b \equiv \text{Conjunto de bombas}, |A_b| = k$$

Definimos  $b = (b_1, b_2, \dots, b_k) \in \{0, 1\}^k$  como un vector de 0 y 1 donde cada elemento del vector hace referencia a una bomba de la red de estudio. Si la bomba  $i$  está en funcionamiento entonces el elemento  $b_i = 1$  o bien si se encuentra apagada  $b_i = 0$ , para todo  $i = 1, \dots, k$ . Así, tenemos  $2^k$  patrones diferentes de bombas y denotamos por  $PB$  a este conjunto de patrones diferentes.

$$M_D \equiv \text{Conjunto de nodos de demanda}, |M_D| = m$$

El vector  $d^t = (d_1^t, d_2^t, \dots, d_m^t) \in \mathbb{R}^m \times \mathbb{R}^t$  determina la demanda de los clientes en un instante de tiempo  $t$ . Nos referiremos a este vector como patrón de demanda. Debemos tener en cuenta que hay como mucho  $T$  patrones diferentes de demanda (donde  $T$  es el número de periodos).

$$M_T \equiv \text{Conjunto de tanques}, |M_T| = n$$

El vector  $l^t = (l_1^t, l_2^t, \dots, l_n^t) \in (R^+)^n \times \mathbb{R}^t$  nos indica el nivel de los depósitos en el instante de tiempo  $t$ . Donde consideramos  $l^0 \equiv$  niveles iniciales de los tanques.

Ahora bien, para cada patrón de bombas  $b = (b_1, \dots, b_k)$  y para cada patrón de demandas  $d^t = (d_1^t, d_2^t, \dots, d_m^t)$ , el software hidráulico EPANET nos puede decir si la solución con estas características mencionadas es factible o no. Es decir el propio EPANET se va a encargar de verificar algunas restricciones físicas e hidráulicas de la red de estudio. En caso de tener factibilidad de nuestra posible solución, podemos obtener:

$$C_{b,d^t} \equiv \text{Consumo total al utilizar los patrones } b \text{ y } d^t$$

Una vez obtenemos el consumo eléctrico total de todas aquellas combinaciones factibles según EPANET obtendremos también las variaciones de los niveles de los tanques para cada una de estas combinaciones.

$$\Delta_{b,d^t} = (\Delta_{b,d^t}^1, \dots, \Delta_{b,d^t}^n) \equiv \text{Variaciones de los niveles de los tanques.}$$

Es decir, una vez obtenemos todas las combinaciones de  $(b, d^t)$  podemos descartar las no factibles y para las factibles obtenemos su consumo total  $C_{b,d^t}$  y las variaciones en los niveles de los tanques  $\Delta_{b,d^t}$ .

Por lo tanto comenzaremos el problema con un conjunto de  $2^k$  posibilidades para las bombas, con todos los patrones de demanda diferentes, que como ya comentamos serán como máximo  $T$ . Consideramos que se asume en para cada  $t$  instante de tiempo los niveles medios de los tanques aunque si los niveles fueran otros, el resultado no debería de ser muy diferente en  $C_{b,d^t}$  y  $\Delta_{b,d^t}$ . Luego a lo sumo tendremos  $2^k \cdot T$  posibilidades diferentes, donde cada una de esas posibilidades factibles nos asegura gracias a EPANET que la demanda y las condiciones físicas e hidráulicas necesarias para el correcto funcionamiento de la red han sido satisfechas es decir EPANET nos aporta soluciones factibles para nuestro problema, pero no óptimas.

Tras este preproceso, con el fin de construir un modelo sencillo para encontrar la mejor de estas soluciones, definimos las variables binarias  $x_b^t$  para cada  $b \in PB$  y  $t = 1, \dots, T$ , de forma que  $x_b^t = 1$  si en el periodo  $t$  se aplica la distribución de bombas  $b$  y vale cero en otro caso. Adicionalmente, definimos las variables continuas  $l_i^t$  para  $i = 1, \dots, n$  y  $t = 1, \dots, T$  como el nivel del deposito  $i$  en el periodo  $t$ .

Partiendo de estas definiciones y teniendo en cuenta que  $\pi$  es el vector que indica el precio de la energía en los distintos periodos, vamos a proponer el siguiente modelo que nos resuelve “aproximadamente” el problema de optimización de la red de distribución de agua:

$$\text{mín} \sum_{t=1}^T \sum_{b \in PB} \pi_t C_{b,d(t)} x_b^t \quad (2.28)$$

sujeto a:

$$\sum_{b \in PB} x_b^t = 1 \quad t = 1, \dots, T \quad (2.29)$$

$$l_i^t = l_i^{t-1} + \Delta_{b,d(t)}^i x_b^t \quad t = 1, \dots, T, i = 1, \dots, n \quad (2.30)$$

$$\underline{l}_i \leq l_i^t \leq \bar{l}_i \quad t = 1, \dots, T, i = 1, \dots, n \quad (2.31)$$

$$|l_i^T - l_i^0| \leq L \quad i = 1, \dots, n \quad (2.32)$$

$$x_b^t \in \{0, 1\} \quad t = 1, \dots, T, b \in PB \quad (2.33)$$

La primera familia de restricciones (2.29) exige que para cada instante de tiempo  $t$  exista una única distribución de bombas que tomaremos como óptima. Las siguientes restricciones (2.30) nos dicen simplemente que el nivel de agua



de un tanque en un periodo de tiempo  $t$  será igual a su nivel en el periodo anterior más la variación en el nivel de los tanques de la distribución de bombas seleccionada. La tercera familia restricciones (2.31) exige que el nivel de nuestros tanques se encuentre entre su nivel máximo permitido  $\bar{l}_i$  y su nivel mínimo  $\underline{l}_i$ . Por último, la cuarta familia de restricciones (2.32) exige que la diferencia del nivel de los tanques al inicio y al final de nuestra simulación sea menor que una constante  $L$ .

Como podemos observar el modelo alternativo se trata de un problema de programación lineal entera-mixta, simplificando considerablemente el proceso de resolución del problema abordado. Sin embargo, debemos hacer notar que hemos supuesto unos niveles intermedios de los tanques para ver si una combinación de bombas y demandas  $(b, d^t)$  es factible y si es factible que compute el consumo eléctrico  $C_{b,d^t}$  y las variaciones de nivel de los tanques  $\Delta_{b,d^t}$ . Por lo tanto, la factibilidad, el consumo y los cambios de nivel también dependen de los niveles de los tanques. Afortunadamente, como se puede ver en los resultados computacionales, esta dependencia es muy pequeña, no obstante presentamos en la siguiente sección un modelo en el que se tiene en cuenta de forma más precisa los niveles de los tanques de un periodo para otro.

### 2.3. Modelo alternativo lineal más ajustado

Como se ha mencionado en la sección anterior, el modelo anterior considera diferentes posibilidades de funcionamiento de las bombas y demandas para el cálculo de la factibilidad, consumo y cambios de niveles de los tanques. A continuación presentamos un modelo que también tiene en cuenta diferentes niveles de los tanques al comienzo de un periodo, ya que estos niveles iniciales influyen (aunque ligeramente) en la factibilidad, consumo y cambios de niveles de los tanques.

Consideremos que para cada tanque  $i = 1, \dots, n$  podemos tener  $q$  posibilidades diferentes de niveles, espaciadas uniformemente entre el nivel mínimo  $\underline{l}_i$  y el nivel máximo  $\bar{l}_i$ . Para ser más precisos tomemos los posibles niveles para cada tanque  $i = 1, \dots, n$ :

$$l_i(j) = \underline{l}_i + \frac{2j-1}{2q}(\bar{l}_i - \underline{l}_i) \quad \text{para } j = 1, \dots, q.$$

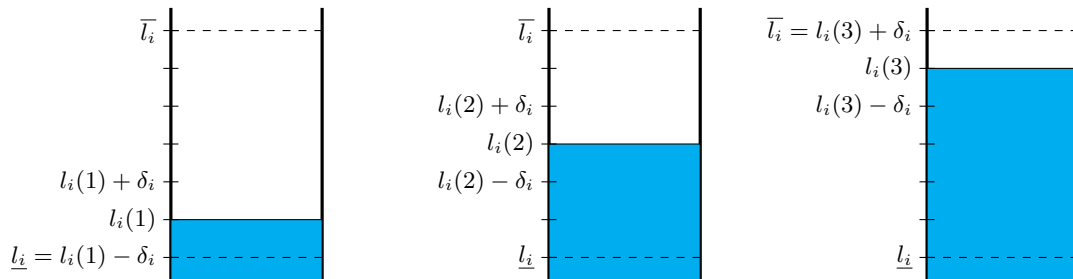
La idea es que si al principio del periodo  $t$  el nivel del estanque es  $l_i^{t-1}$  trataremos de utilizar una simulación de EPANET para ese periodo con el nivel del tanque  $l_i(j)$  lo más próxima posible a  $l_i^{t-1}$  para algún  $j = 1, \dots, q$ . Es decir, si denotamos por

$$\delta_i = \frac{1}{2q}(\bar{l}_i - \underline{l}_i),$$

entonces tomaremos para el periodo  $t$  una configuración para un  $j = 1, \dots, q$  que verifique que:

$$l_i^{t-1} \in [l_i(j) - \delta_i, l_i(j) + \delta_i].$$

La Figura 2.1 muestra los tres niveles posibles  $l_i(1)$ ,  $l_i(2)$  y  $l_i(3)$  si tomamos  $q = 3$ .



**Figura 2.1.** Los tres niveles de referencia de un tanque cuando  $q = 3$ .

Consideremos por el vector  $p = (l_1(j_1), \dots, l_n(j_n))$  una combinación de niveles de tanques y por  $PT$  al conjunto de las  $q^n$  posibilidades distintas que puede valer el vector  $p$ . Para hacer referencia a un elemento de  $p$ , utilizaremos la notación  $p_i$  por simplicidad.

De forma similar a lo realizado en la sección anterior, ahora podemos considerar con EPANET la factibilidad, el consumo y los cambios de nivel que se pueden producir para las  $2^k$  combinaciones de bombas,  $T$  periodos de tiempos y  $q^n$  niveles de tanques. Así, podemos definir la variable  $x_{bp}^t$  para cada  $b \in PB$ , para cada  $p \in PT$  y cada periodo  $t = 1, \dots, T$ , de forma que  $x_{bp}^t = 1$  si en el periodo  $t$  se aplica la distribución de bombas  $b$  partiendo de unos niveles de los tanques  $p$ , y vale cero en otro caso. Adicionalmente, definimos las variables continuas  $l_i^t$  para  $i = 1, \dots, n$  y  $t = 1, \dots, T$  como el nivel del deposito  $i$  en el periodo  $t$ . Partiendo de estas definiciones y conceptos vamos a proponer el siguiente modelo lineal, que es más ajustado que el anterior:

$$\text{mín} \sum_{t=1}^T \sum_{b \in PB} \sum_{p \in PT} \pi_t C_{b,p,d(t)} x_{bp}^t \quad (2.34)$$

sujeto a:

$$\sum_{b \in PB} \sum_{p \in PT} x_{bp}^t = 1 \quad t = 1, \dots, T \quad (2.35)$$

$$l_i^t = l_i^{t-1} + \Delta_{b,p,d(t)}^i x_{bp}^t \quad t = 1, \dots, T, i = 1, \dots, n, p \in PT \quad (2.36)$$

$$(p_i - \delta_i) - (1 - x_{bp}^t) \cdot (p_i - \delta_i - \underline{l}_i) \leq l_i^t \quad t = 1, \dots, T, i = 1, \dots, n, p \in PT \quad (2.37)$$

$$(p_i + \delta_i) + (1 - x_{bp}^t)(\bar{l}_i - p_i - \delta_i) \geq l_i^t \quad t = 1, \dots, T, i = 1, \dots, n, p \in PT \quad (2.38)$$

$$|l_i^T - l_i^0| \leq L \quad i = 1, \dots, n \quad (2.39)$$

$$x_{bp}^t \in \{0, 1\} \quad t = 1, \dots, T, b \in PB, p \in PT \quad (2.40)$$

La función objetivo (2.34) es muy similar a la del modelo anterior, sin embargo, en esta ocasión el consumo ( $C_{b,p,d(t)}$ ) y la variable binaria ( $x_{bp}^t$ ) dependen también del nivel inicial de los tanques  $p$ . Por ello añadimos un sumatorio extra para todo  $p \in PT$ . El concepto de la primera familia de restricciones (2.35) sigue siendo el mismo, exigir que para el instante de tiempo  $t$  solamente una de las posibilidades (combinando las distribuciones de bombas y los distintos niveles iniciales de los tanques) sea la válida para nuestra restricción. La segunda familia (2.36) no varía salvo en que debemos tener en cuenta que las variaciones en el nivel de los tanques ( $\Delta_{b,p,d(t)}^i$ ) ahora dependerán también de los niveles iniciales

$p$ . Las restricciones (2.37)–(2.38) las consideramos como una única familia pero las hemos separado debido a su tamaño. Si  $x_{bp}^t = 0$ , son equivalentes a (2.31), sin embargo, si  $x_{bp}^t = 1$  entonces tendremos que  $p_i - \delta_i \leq l_i^t \leq p_i + \delta_i$  estableciendo de esta forma que el nivel del tanque  $i$  en el instante  $t$  se encuentre entorno a  $p_i$ . Las dos últimas familias de restricciones son idénticas a las del ejemplo anterior.

Al igual que en modelo anterior tenemos un modelo de programación lineal entera-mixta, que considera más posibilidades aún que nuestro modelo alternativo.

## Resultados computacionales

Tras el desarrollo de nuestro modelo alternativo, lo implementamos en PYTHON, para probar su desempeño con varias redes de distribución de agua. He de destacar que las redes en su mayoría tienen formato de fichero ".inp". Este tipo de ficheros son con los que trabaja EPANET y básicamente nos proporcionan toda la información necesaria de nuestra red. En los siguientes ejemplos utilizaremos el precio de la luz en España el día 13 de Febrero de 2023 que se presenta en la Figura 3.1.



**Figura 3.1.** Precio de la luz durante el día 13 de febrero

Comenzaremos por una red simple, la red “Ex1.inp”. Esta red consta de un embalse, una bomba, un tanque y 6 nodos de paso o demanda, como vemos en la Figura 3.2. Nuestro objetivo es encontrar la distribución de bombas (en este caso de una sola bomba) tal que el gasto total al satisfacer la demanda de los clientes para cada hora sea mínimo.

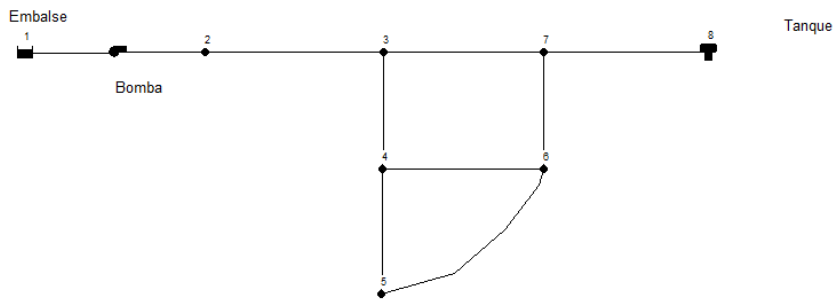


Figura 3.2. Ejemplo 1: Red simple Ex1.inp

Observando el fichero de esta red podemos destacar algunos parámetros cuyo valor afectará directamente a la solución obtenida. Por ejemplo, en nuestro fichero “Ex1.inp” nos viene dado el número de horas que quiero realizar la simulación (24h) y también podemos observar la demanda base de los nodos y su patrón de demanda. Para entender todo esto de forma sencilla adjunto a continuación varias capturas del fichero “Ex1.inp”.

```
[TITLE]

[JUNCTIONS]
;ID      Elev      Demand      Pattern
2        700      0           ;
3        710      150        patt_1     ;
4        700      150        patt_1     ;
5        650      200        patt_1     ;
6        700      150        patt_1     ;
7        700      0           ;

[RESERVOIRS]
;ID      Head      Pattern
1        700           ;

[TANKS]
;ID      Elevation  InitLevel  MinLevel  MaxLevel  Diameter  MinVol  VolCurve
8        830      10.3      10        20        60        0       ;

[PIPES]
;ID      Node1      Node2      Length   Diameter  Roughness  MinorLoss  Status
1        2        3          3000    14        100       0          Open ;
2        2        7          5000    12        100       0          Open ;
3        3        4          5000    8         100       0          Open ;
4        4        6          5000    8         100       0          Open ;
5        4        7          5000    8         100       0          Open ;
```

Figura 3.3. Ejemplo 1: fichero Ex1.inp

En la Figura 3.3 observamos la primera parte del fichero. Aquí podemos ver algunos de los distintos elementos presentes en toda red de distribución de agua como pueden ser los nodos de demanda, los tanques, tuberías, etc. Si prestamos atención a la parte inicial, en el apartado *Juntions*, vemos las distintas características de cada uno de los nodos. En la columna *Demands* tenemos lo que denominamos la demanda base de los nodos mientras que en la columna *Patterns* tenemos patt\_1, que hace referencia a un vector de proporciones presente en la Figura 3.4. Al multiplicarlo por la demanda base de un nodo en concreto, obtenemos el patrón de demanda de dicho nodo para todos los periodos de nuestra simulación.

```
[PATTERNS]
;ID          Multipliers
patt_1      0.02          0.02          0.02          0.02          0.04          0.06
patt_1      0.07          0.07          0.04          0.02          0.02          0.04
patt_1      0.06          0.05          0.04          0.03          0.04          0.06
patt_1      0.07          0.07          0.06          0.05          0.02          0.02
```

Figura 3.4. Ejemplo 1: Vector de proporciones patt\_1

Teniendo en cuenta nuestro ejemplo actual y focalizando en el nodo con ID 3, podemos ver que su demanda base es 150 y que la proporción necesaria de dicha demanda para la primera hora es de 0.02. Por lo tanto nuestro nodo 3 tendrá en la primera hora una demanda de  $150 \cdot 0,02 = 3 \text{ m}^3/h$  y en la sexta hora su demanda será de  $150 \cdot 0.06 = 9 \text{ m}^3/h$ . Debo destacar que en este caso todos los nodos tenían asociado el mismo vector de proporciones, pero esto no tiene porque ser así, puesto que cada nodo podría tener su propia demanda base y vector de proporciones algo que veremos en ejemplos futuros.

En este primer ejemplo, tenemos un máximo de  $2 * 24 = 48$  distribuciones posibles combinando las distintas demandas y controlando el funcionamiento de la bomba. Al ejecutar el programa el preproceso tarda 0.97 segundos y la resolución del modelo contando que el preproceso ya ha finalizado será de 0.02 segundos, es decir, obtenemos la solución final en 0.99 segundos.

En la Tabla 3.1 se presentan los resultados computacionales obtenidos al tratar de ejecutar el Modelo Alternativo sobre esta red de prueba. Como en este caso solo contamos con una única bomba el proceso se simplifica puesto que deberemos elegir cuando abrir o cerrar la bomba. Como es de esperar la bomba se abre en los dos periodos donde el precio de la luz es menor (periodo 4 y 5), tras estos periodos el tanque se llena llegando a acumular hasta casi 13 metros de agua, al parecer suficiente para abastecer la demanda de la red en el resto de periodos.

**Tabla 3.1.** Resultados Ejemplo 1

Periodos	Bombas	Tanques	Consumo	Precio	Gasto
1	[0]	10.26	0.00	0.19	0.00
2	[0]	10.23	0.00	0.19	0.00
3	[0]	10.19	0.00	0.19	0.00
4	[1]	11.58	20.72	0.18	3.71
5	[1]	12.95	20.73	0.18	3.75
6	[0]	12.84	0.00	0.19	0.00
7	[0]	12.71	0.00	0.18	0.00
8	[0]	12.58	0.00	0.20	0.00
9	[0]	12.51	0.00	0.23	0.00
10	[0]	12.47	0.00	0.22	0.00
11	[0]	12.44	0.00	0.26	0.00
12	[0]	12.36	0.00	0.25	0.00
13	[0]	12.25	0.00	0.25	0.00
14	[0]	12.16	0.00	0.24	0.00
15	[0]	12.08	0.00	0.19	0.00
16	[0]	12.03	0.00	0.19	0.00
17	[0]	11.96	0.00	0.20	0.00
18	[0]	11.85	0.00	0.21	0.00
19	[0]	11.72	0.00	0.27	0.00
20	[0]	11.59	0.00	0.30	0.00
21	[0]	11.48	0.00	0.31	0.00
22	[0]	11.38	0.00	0.29	0.00
23	[0]	11.35	0.00	0.23	0.00
24	[0]	11.31	0.00	0.22	0.00

Cabe destacar que al finalizar el proceso el tanque cuenta con un volumen mayor que al inicio. En el periodo 4 la bomba gastará 3.71 euros mientras que en el periodo 5 el gasto será de 3.75, sumando un total de 7.45 euros gastados en el día para abastecer correctamente la demanda de esta pequeña red.

Debemos tener en cuenta que aun no hemos asegurado la factibilidad de la solución obtenida y que los resultados de esta tabla se obtuvieron considerando que al inicio de cada periodo el nivel de los tanques es el inicial. Es decir, los resultados obtenidos en la primera tabla son aquellos que nos aporta directamente nuestro programa, teniendo en cuenta que se han realizado de forma independiente las simulaciones de cada hora y por ello al comienzo de cada nueva hora toma como altura de los tanques al valor inicial. Por ello ahora ejecutaremos en EPANET el fichero ‘resultados.inp’, puesto que este fichero contiene las distribuciones de bombas que optimizan el gasto en nuestra red. Así probaremos si la solución obtenida a partir de nuestro modelo alternativo es factible o no (puesto que EPANET verificará las condiciones físicas e hidráulicas de la red) y observaremos también las diferencias en el nivel de los tanques y como consecuencia directa en el consumo de las bombas y por la tanto en el gasto. Estos nuevos resultados estarán presentes en la [Tabla 3.2](#).



**Tabla 3.2.** Comprobación Ejemplo 1

Periodos	Bombas	Tanques	Consumo	Precio	Gasto
1	[0]	10.26	0.00	0.19	0.00
2	[0]	10.23	0.00	0.19	0.00
4	[1]	11.58	20.88	0.18	3.74
5	[1]	12.94	20.84	0.18	3.77
6	[0]	12.83	0.00	0.19	0.00
7	[0]	12.70	0.00	0.18	0.00
8	[0]	12.57	0.00	0.20	0.00
9	[0]	12.49	0.00	0.23	0.00
10	[0]	12.46	0.00	0.22	0.00
11	[0]	12.42	0.00	0.26	0.00
12	[0]	12.35	0.00	0.25	0.00
13	[0]	12.24	0.00	0.25	0.00
14	[0]	12.14	0.00	0.24	0.00
15	[0]	12.07	0.00	0.19	0.00
16	[0]	12.01	0.00	0.19	0.00
17	[0]	11.94	0.00	0.20	0.00
18	[0]	11.83	0.00	0.21	0.00
19	[0]	11.70	0.00	0.27	0.00
20	[0]	11.57	0.00	0.30	0.00
21	[0]	11.46	0.00	0.31	0.00
22	[0]	11.37	0.00	0.29	0.00
23	[0]	11.33	0.00	0.23	0.00
24	[0]	11.30	0.00	0.22	0.00

Tras ejecutar el fichero de resultados de este ejemplo en EPANET, verificamos la factibilidad de nuestra solución. Como podemos ver las tablas son casi idénticas, ya que el fichero con los resultados exige que la distribución de bombas en cada momento sea la obtenida en nuestra solución anterior pero considera en cada hora como niveles iniciales del volumen de los tanques a los niveles finales de la hora anterior. Observando las diferencias entre las variaciones del volumen de los tanques y del consumo de las bombas en ambas tablas, se alejan como mucho unas décimas de lo esperado y por lo tanto podemos concluir que en este primer ejemplo la ejecución de nuestro modelo ha sido un éxito, teniendo un gasto real de 7.51 euros o lo que es lo mismo una diferencia de 0.8% con lo obtenido al ejecutar nuestro programa.

Ahora probaremos nuestro programa con una red algo más compleja, la red “Net3.inp”. Esta red consta de dos embalses (un río y un lago), tres tanques de agua, dos bombas que conectan a los embalses con la red y un gran número de nodos de demanda y tuberías. La Figura 3.5 muestra la representación de esta red. El objetivo es el mismo, minimizar el gasto al bombear el agua desde los embalses para que la demanda sea satisfecha.

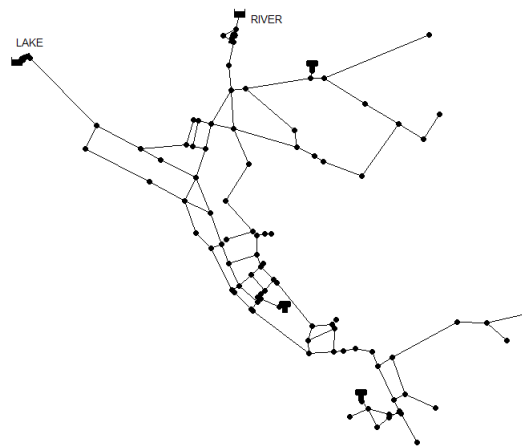


Figura 3.5. Ejemplo 2: Red de distribución de agua Net3.inp

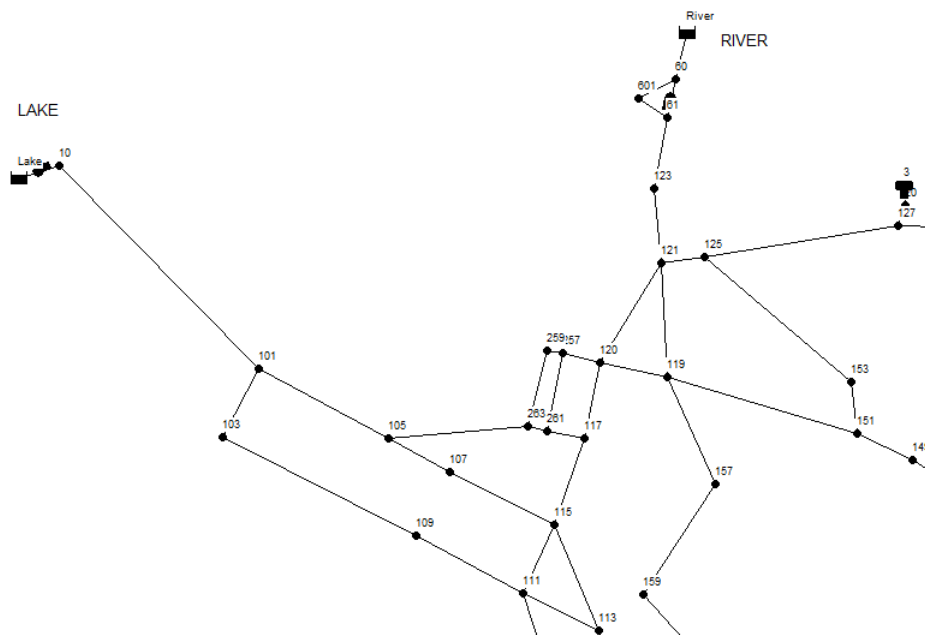


Figura 3.6. Ejemplo 2: Red Net3.inp ampliada

Para entender un poco mejor el funcionamiento de esta red nos centraremos en la localización de las bombas y el sistema de tuberías que lo rodea, mostradas en la Figura 3.6.

En primer lugar, si observamos el embalse *Lake* vemos como está conectado a la red por una bomba que parte de dicho embalse hasta el nodo 10. Esta conexión servirá solamente para aportar agua a la red, pero no para vaciar agua de la propia red hacia el lago puesto que cuando cerramos la bomba el agua no puede ir en ningún sentido.

Por otra parte, prestando atención al siguiente embalse, vemos a simple vista como la red de tuberías entorno a este es algo más compleja. En este caso en concreto tenemos al embalse conectado a uno de los nodos de paso que se encuentra al nivel del mar (nodo 60), a su vez este nodo de paso está conectado por medio de una bomba a otro nodo de paso que también se encuentra al nivel del mar (nodo 61) y finalmente el nodo 61 se conecta por una tubería al nodo 123 (con altura geográfica de 11 metros) conectando así a nuestro embalse *River* con la red. En el caso de tener un exceso de agua en la red (supongamos tanques llenos y demanda satisfecha) el agua fluiría en sentido contrario yendo del nodo 123 al nodo 61. Desde el nodo 61 el agua no puede avanzar directamente al nodo 60 puesto que con las condiciones anteriores la bomba deberá estar cerrada, de ahí la conexión paralela que observamos en la imagen. Por lo tanto en caso de exceso de agua al llegar al nodo 61 el agua tomará la tubería que conecta con el nodo de paso 601 y que a su vez se encuentra conectado con el nodo 60, estableciendo así una vía para el vaciado de la red en caso de que fuese necesario.

También debo resaltar que los patrones de demanda en este problema se diferencian de los del primer ejemplo. En este caso tenemos numerosos nodos con distintos patrones de demanda como observamos en las Figuras 3.7 y 3.8.

```
[TITLE]
EPANET Example Network 3].

[JUNCTIONS]
;ID          Elev          Demand          Pattern          ;
10           147           0               ;
15           32            1               3                 ;
20           129           0               ;
35           12.5         1               4                 ;
40           131.9        0               ;
50           116.5        0               ;
60           0             0               ;
601          0             0               ;
61           0             0               ;
101          42            189.95          ;
103          43            133.2           ;
105          28.5         135.37          ;
107          22            54.64           ;
109          20.3         231.4           ;
111          10            141.94          ;
113          2             20.01           ;
115          14            52.1            ;
117          13.6         117.71          ;
119          2             176.13          ;
120          0             0               ;
121          -2           41.63           ;
123          11            1               2                 ;
```

Figura 3.7. Ejemplo 2: Fichero Net3.inp

```
[PATTERNS]
;ID          Multipliers
;General Default Demand Pattern
1            1.34          1.94          1.46          1.44          .76          .92
1            .85          1.07          .96          1.1          1.08         1.19
1            1.16         1.08          .96          .83          .79          .74
1            .64          .64          .85          .96          1.24         1.67
;Demand Pattern for Node 123
2            0             0             0             0             0             1219
2            0             0             0             1866          1836          1818
2            1818         1822          1822          1817          1824          1816
2            1833         1817          1830          1814          1840          1859
;Demand Pattern for Node 15
3            620          620          620          620          620          360
3            360          0             0             0             0             360
3            360          360          360          360          0             0
3            0             0             0             0             360          360
;Demand Pattern for Node 35
4            1637         1706          1719          1719          1791          1819
4            1777         1842          1815          1825          1856          1801
4            1819         1733          1664          1620          1613          1620
4            1616         1647          1627          1627          1671          1668
;Demand Pattern for Node 203
5            4439         4531          4511          4582          4531          4582
5            4572         4613          4643          4643          4592          4613
5            4531         4521          4449          4439          4449          4460
5            4439         4419          4368          4399          4470          4480
```

Figura 3.8. Ejemplo 2: Patrones de demanda

Como observamos en los extractos de ficheros anteriores los nodos poseen distintas demandas base, así como distintos patrones de demanda. En la Figura 3.7 los nodos que no tienen asignado un patrón toman el patrón 1 por defecto mientras que por ejemplo el nodo 15 tiene un patrón de demanda particular definido en el patrón 3. Esta simulación al igual que la anterior se realizará a

24 horas. Tras conocer algunas características e introducir ligeramente el funcionamiento de la red, procedemos a ejecutar nuestro programa con este segundo ejemplo.

**Tabla 3.3.** Resultados Ejemplo 2

Periodos	Bombas	Tanques			Consumo			Precio	Gastos
		T1	T2	T3	B1	B2	Total		
1	[1 1]	15.20	23.54	30.52	62.70	309.67	372.38	0.19	70.21
2	[0 0]	12.20	22.79	27.99	0.00	0.00	0.00	0.19	0.00
3	[1 1]	14.12	22.74	29.43	62.71	309.66	372.37	0.19	69.25
4	[1 1]	16.03	22.69	30.86	62.71	309.66	372.37	0.18	66.68
5	[1 1]	18.54	22.97	32.65	62.68	309.72	372.40	0.18	67.30
6	[1 1]	20.62	23.12	34.11	62.70	309.67	372.37	0.19	69.45
7	[1 0]	19.67	22.85	32.29	63.52	0.00	63.52	0.18	11.50
8	[1 1]	21.93	23.00	34.08	62.69	309.71	372.40	0.20	73.55
9	[1 0]	20.90	22.68	32.27	63.52	0.00	63.52	0.23	14.64
10	[1 1]	22.70	22.72	33.58	62.72	309.64	372.36	0.22	81.66
11	[0 0]	19.70	22.05	30.87	0.00	0.00	0.00	0.26	0.00
12	[0 0]	16.70	21.38	28.20	0.00	0.00	0.00	0.25	0.00
13	[0 0]	13.70	20.71	25.51	0.00	0.00	0.00	0.25	0.00
14	[0 0]	10.70	20.04	22.77	0.00	0.00	0.00	0.24	0.00
15	[1 1]	12.74	20.19	24.11	62.71	309.65	372.36	0.19	71.85
16	[1 1]	14.93	20.39	25.53	62.70	309.67	372.37	0.19	72.54
17	[1 1]	17.20	20.62	27.08	62.70	309.68	372.38	0.20	74.96
18	[1 1]	19.51	20.87	28.65	62.70	309.69	372.38	0.21	78.09
19	[1 0]	18.42	20.62	26.65	63.52	0.00	63.52	0.27	17.40
20	[0 0]	15.73	20.08	23.92	0.00	0.00	0.00	0.30	0.00
21	[0 0]	12.81	19.46	21.09	0.00	0.00	0.00	0.31	0.00
22	[1 0]	11.32	19.08	18.93	63.52	0.00	63.52	0.29	18.21
23	[1 1]	13.10	19.09	20.11	62.72	309.62	372.35	0.23	86.09
24	[0 0]	10.10	18.37	17.54	0.00	0.00	0.00	0.22	0.00

En este segundo ejemplo, tenemos un máximo de  $2^2 * 24 = 96$  posibilidades, puesto que para cada periodo de tiempo tenemos 4 posibles distribuciones de bombas. Al ejecutar el programa el preproceso tarda 10.18 segundos y la resolución del modelo contando que el preproceso ya ha finalizado será de 0.93 segundos, es decir, obtenemos la solución final en 11.2 segundos aproximadamente.

En los resultados de la Tabla 3.3 observamos como el nivel de agua en los tres tanques varía a lo largo de la simulación, teniendo en algunos casos variaciones notables respecto a los niveles iniciales. Por ejemplo T3, que comienza con 26 metros de agua y termina con 17. Las grandes variaciones en el nivel de los tanques pueden suponer un problema para las futuras simulaciones puesto que si tenemos al comienzo de la simulación un nivel bajo en nuestros tanques podríamos encontrarnos con la no factibilidad del problema. Cabe destacar que para el cálculo de esta solución debimos modificar el valor de una constante en una de las restricciones sobre los tanques de nuestro modelo alternativo, en

concreto la tercera. Aumentamos el valor de la constante  $L$  hasta 12 ya que para valores menores de  $L$  no existe solución factible para nuestra red. Esto se podría controlar añadiendo válvulas a la red que impidan vaciar los tanques bajo determinadas condiciones, pero nuestro objetivo no es modificar redes para mejorar su rendimiento, sino optimizar las existentes.

Observando el consumo de las bombas, la primera bomba gasta menos energía que la segunda, estando activa en 16 de los 24 periodos, mientras que la segunda bomba con un consumo mucho más elevado, se encuentra conectada 12 horas al día. Como era de esperar las horas donde el consumo será mas elevado serán aquellas donde inevitablemente tengan que funcionar ambas bombas para satisfacer la demanda.

El costo total será de 943.38 euros. Ahora como en el ejemplo 1 ejecutaremos el archivo de salida donde en lugar de realizarse la simulación hora a hora se realizan las 24 horas en conjunto para poder verificar con EPANET la factibilidad de nuestra solución y comparar los valores del nivel de los tanques y el consumo de la bombas.

**Tabla 3.4.** Comprobación Ejemplo 2

Periodos	Bombas	Tanques			Consumo			Precio	Gastos
		T1	T2	T3	B1	B2	Total		
1	[1 1]	15.20	23.54	30.52	62.21	309.85	372.06	0.19	70.15
2	[0 0]	11.25	22.59	27.24	0.00	0.00	0.00	0.19	0.00
3	[1 1]	13.23	22.52	28.70	62.49	309.48	371.96	0.19	69.17
4	[1 1]	15.07	22.50	30.14	62.37	309.70	372.07	0.18	66.62
5	[1 1]	17.44	22.83	31.92	61.34	310.37	371.71	0.18	67.18
6	[1 1]	19.24	23.09	33.35	61.64	310.03	371.67	0.19	69.32
7	[1 0]	17.73	23.10	31.51	62.56	0.00	62.56	0.18	11.33
8	[1 1]	19.66	23.33	33.28	61.62	310.45	372.07	0.20	73.49
9	[1 0]	18.05	23.28	31.47	62.63	0.00	62.63	0.23	14.44
10	[1 1]	19.41	23.43	32.79	61.96	309.68	371.64	0.22	81.51
11	[0 0]	15.76	22.77	29.73	0.00	0.00	0.00	0.26	0.00
12	[0 0]	11.97	21.99	26.52	0.00	0.00	0.00	0.25	0.00
13	[0 0]	10.10	21.58	24.90	0.00	0.00	0.00	0.25	0.00
14	[0 0]	10.10	21.18	23.36	0.00	0.00	0.00	0.24	0.00
15	[1 1]	10.10	20.37	20.30	62.50	308.09	370.59	0.19	70.41
16	[1 1]	11.71	20.44	21.90	62.24	308.49	370.73	0.19	70.44
17	[1 1]	13.42	20.61	23.56	62.04	308.88	370.91	0.20	74.18
18	[1 1]	15.16	20.85	25.31	61.83	309.20	371.03	0.21	77.91
19	[1 0]	16.89	21.14	27.06	62.78	0.0	62.78	0.27	16.95
20	[0 0]	14.95	21.06	25.19	0.00	0.00	0.00	0.30	0.00
21	[0 0]	11.78	20.53	22.59	0.00	0.00	0.00	0.31	0.00
22	[1 0]	10.10	20.20	21.20	63.30	0.00	63.3	0.29	18.35
23	[1 1]	10.10	19.87	19.87	62.30	308.77	371.07	0.23	85.35
24	[0 0]	10.10	19.41	17.95	0.00	0.00	0.00	0.22	0.00

Tras introducir nuestra solución óptima en EPANET, el software nos verifica su factibilidad. Observando la Tabla 3.5 y en concreto la parte central de esta (nivel de los tanques y consumo de bombas), podemos ver como las variaciones en el consumo de las bombas son realmente leves, obteniendo un gasto total real de 936.8 euros, un 0.70% menor que el obtenido con el modelo lineal. Por otra parte, en los niveles de los tanques existen diferencias ligeramente mayores para algunos periodos pero estas desviaciones eran algo esperado. Centrándonos ahora en las últimas filas de ambas tablas, el resultado que más se aleja del valor previsto es el de T2, con una diferencia de 1.1 metros aproximadamente, por lo tanto nuestra solución óptima aportada mejora en condiciones reales puesto que el gasto es menor y todos los tanques terminan con un nivel igual o ligeramente superior al esperado.

Por último, trataremos de simular la red utilizada en [?] en el apartado de resultados computacionales. Antes de abordar este ejemplo, quiero destacar que el diseño de esta red se hace en base a los datos aportados en [?] pero debemos tener en cuenta que hay algunos datos que omiten, como la altura geográfica del tanque y las curvas (altura-caudal) que definen la altura máxima que puede vencer una bomba, entre otros. Por ello y por determinadas consideraciones que toman en [?] no será factible comparar nuestros resultados con los del artículo. A pesar de ello, presentaremos el problema y aportaremos una solución y comprobaremos la factibilidad y velocidad de nuestros resultados. La Figura 3.9 presenta la forma de nuestra tercera red de ejemplo.

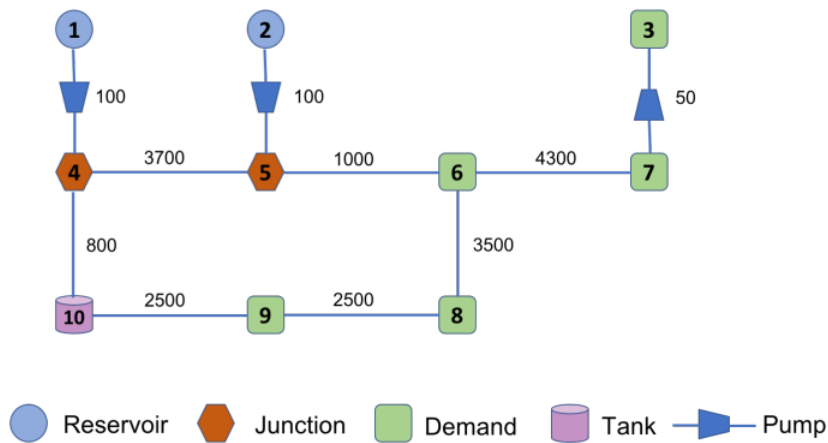


Figura 3.9. Ejemplo 3: red del artículo

Se observa como esta red consta de 2 embalses (nodos 1 y 2), dos nodos de paso, cinco nodos de demanda, un tanque, varias tuberías y tres bombas de agua. Al lado de cada tubería tenemos su longitud, mientras que al lado de cada bomba tenemos la presión que esta añade a la red, ambas características vienen dadas en metros.

Nuestro objetivo es minimizar el gasto de las bombas cubriendo la demanda establecida. Cabe destacar que al contrario de los otros dos ejemplos anteriores la simulación en esta ocasión es a 12 horas. En este ejemplo en concreto, cada nodo de demanda presenta un patrón de demanda distinto.

Ahora ejecutaremos nuestro programa con esta red y analizaremos la solución obtenida. Recordemos que en este ejemplo contamos con una simulación a 12 horas y con tres bombas de agua, por lo tanto existen un total de  $2^3 * 12 = 96$  posibilidades. Como desconocemos la altura geográfica del tanque la tomaremos en 100 metros.



```
[PATTERNS]
;ID          Multipliers
;Patron de demanda del nodo 3
patt3        170          175          158          120          110          80
patt3        65           85           100          150          165          170
;Patron de demanda del nodo 6
patt6        220          222.5        205          170          150          125
patt6        110          130          150          165          190          220
;Patron de demanda del nodo 7
patt7        200          205          180          150          130          115
patt7        100          110          120          150          180          200
;Patron de demanda del nodo 8
patt8        150          150          135          100          85           75
patt8        80           95           100          120          135          150
;Patron de demanda del nodo 9
patt9        140          130          120          100          80           60
patt9        70           90           90           110          125          140
```

Figura 3.10. Ejemplo 3: Patrones de demanda

Tabla 3.5. Resultados Ejemplo 3

Periodos	Bombas	Tanques	Consumo				Precio	Gastos
			T1	B1	B2	B3		
1	[1 1 0]	20.25	190.89	241.07	0.00	431.96	0.19	81.44
2	[1 1 0]	20.49	190.91	241.08	0.00	432.00	0.19	81.44
3	[1 1 0]	20.89	189.80	240.46	0.00	430.26	0.19	80.01
4	[1 1 0]	21.57	187.69	239.28	0.00	426.97	0.18	76.45
5	[1 1 0]	22.41	186.51	238.63	0.00	425.14	0.18	76.83
6	[1 1 0]	23.44	185.11	237.86	0.00	422.97	0.19	78.88
7	[1 1 0]	24.51	184.73	237.65	0.00	422.38	0.18	76.48
8	[0 1 0]	24.63	0.00	247.09	0.00	247.09	0.20	48.80
9	[0 0 0]	23.49	0.00	0.00	0.00	0.00	0.23	0.00
10	[1 0 0]	23.00	206.46	0.00	0.00	206.46	0.22	45.28
11	[0 0 0]	21.38	0.00	0.00	0.00	0.00	0.26	0.00
12	[0 0 0]	19.58	0.00	0.00	0.00	0.00	0.25	0.00

En la Tabla 3.5, podemos ver como la bomba con mayor consumo es B2, estando activa en todos los periodos salvo en los cuatro últimos periodos donde la tarifa de la luz es más elevada, seguida de B1 que está en funcionamiento durante ocho periodos. La última bomba no se abre durante toda la simulación, esto se debe a que paralela a la bomba B3 hay una tubería bidireccional de tal forma que si llega el agua al nodo 7 con el caudal y presión suficientes como para elevarse hasta el nodo 3 y satisfacer su demanda, no es necesario abrir la bomba, puesto que la activación de esta tercera bomba sería completamente inútil en este caso (tener en cuenta que si consideramos una bomba como cerrada el agua no puede pasar a través de ella).

El costo total será de 645.63 euros. El preproceso se realiza en 1.87 segundos y la resolución del problema una vez finalizado el preproceso tarda 0.05 segundos, es decir, el programa en conjunto encuentra una solución en 1.93 segundos.

Ahora como en los ejemplos anteriores condensaremos nuestra solución en un fichero de resultados para verificar si la solución aportada por el modelo

alternativo es factible para P2 y observaremos las posibles variaciones en el nivel de los tanques y en el consumo de las bombas con respecto a nuestra solución.

**Tabla 3.6.** Comprobación Ejemplo 3

Periodos	Bombas	Tanques	Consumo				Precio	Gastos
			T1	B1	B2	B3		
1	[1 1 0]	20.25	192.53	241.94	0.00	434.48	0.19	81.92
2	[1 1 0]	20.48	190.89	241.07	0.00	431.96	0.19	81.44
3	[1 1 0]	20.84	187.40	238.61	0.00	426.01	0.19	79.22
4	[1 1 0]	21.47	181.28	234.19	0.00	415.47	0.18	74.39
5	[1 1 0]	22.21	174.84	229.75	0.00	404.60	0.18	73.12
6	[1 1 0]	23.08	167.00	224.30	0.00	391.30	0.19	72.98
7	[1 1 0]	23.93	158.19	218.80	0.00	376.99	0.18	68.26
8	[0 1 0]	23.93	0.00	221.22	0.00	221.22	0.20	43.69
9	[0 0 0]	22.78	0.00	0.00	0.00	0.00	0.23	0.00
10	[1 1 0]	22.17	178.51	0.00	0.00	178.51	0.22	39.15
11	[0 0 0]	20.55	0.00	0.00	0.00	0.00	0.26	0.00
12	[0 0 0]	18.76	0.00	0.00	0.00	0.00	0.25	0.00

En este caso EPANET aprueba la factibilidad de nuestra solución y además vemos en Tabla 3.6 que las variaciones de los niveles de los tanques con respecto a las obtenidas en Tabla 3.5 son leves. Sin embargo podemos observar como estas leves variaciones en el nivel de los tanques afectan ligeramente al consumo de las bombas, alcanzando un gasto total tras las doce horas de 614.17 euros, obteniendo por lo tanto una solución real un poco mejor que la del modelo alternativo (que como hemos dicho no es del todo exacto).

Hemos ejecutado el programa con 3 redes distintas. Obteniendo en los tres casos buenos resultados con un tiempo de procesamiento corto, pero ha habido un problema que no hemos podido solucionar. Sobre el papel nuestro programa debería ser capaz de optimizar cualquier red de distribución de agua independientemente de su tamaño. Pero realmente esto no es así. Al ejecutar una red donde todas las posibilidades de nuestro preproceso superen el número 99 obtenemos un error. Este error podría haber sido solventado implementando nuestro código C, pero debido a la gran cantidad de tiempo dedicado a este trabajo, decidimos no era factible volver a comenzar todo el proceso.

## Conclusiones

El problema de la optimización de las redes de distribución de agua es un problema real cuyo correcto desarrollo e implementación puede ahorrar a entidades públicas o privadas de todo el mundo grandes cantidades de dinero. Nuestro objetivo inicial era mejorar la implementación en PYTHON del problema (P2) desarrollado por el equipo de mi tutor externo Rafael Nebot Medina (ingeniero del ITC). Después de unos cuantos meses tratando de ejecutar este sofisticado programa, no logramos obtener buenos resultados. Las principales razones fueron el excesivo tiempo computacional en resolver el modelo sobre el problema convexo y que no se garantizaba que la solución de este modelo fuera factible. Ante esta situación decidimos crear nuestro propio programa independiente, implementando un modelo alternativo lineal, que simplifica de manera considerable el problema abordado.

Para implementar nuestro modelo alternativo en PYTHON hicimos uso de la librería “epanettools”, que nos permitía simular el funcionamiento de las redes además de modificar distintos parámetros de entrada como pueden ser los controles sobre las distintas bombas o los patrones de demanda. Para el desarrollo de este modelo, EPANET fue fundamental ya que modificábamos el fichero de entrada manejando los controles de las bombas y los patrones de demanda para probar la factibilidad de nuestras posibles soluciones. Tras conocer que soluciones eran factibles, obteníamos toda la información relevante de estas por medio de “epanettools” y descartábamos las no factibles, es decir, aquellas soluciones que no permitían satisfacer la demanda o bien que generaban roturas. A esta parte es lo denominamos como preproceso.

Una vez finalizado el preproceso, aplicamos el modelo alternativo lineal y obtenemos una solución óptima. Debido a que nuestro modelo es una relajación del problema real, no podemos asegurar que nuestra solución sea factible. Para verificar su factibilidad escribimos un fichero de salida que contiene todos los controles de las bombas que constituyen nuestra solución óptima. Por último

introducimos este fichero en EPANET y el *software* nos dirá si nuestra solución (ahora sin considerar ninguna aproximación) es o no factible.

Cabe destacar que EPANET es un código fuente escrito en C y “epanet-tools” es un API de PYTHON que tiene ciertas limitaciones. En nuestro caso, descubrimos que no permite más de 100 iteraciones, lo que influye en el correcto funcionamiento de nuestro programa al enfrentar grandes redes de distribución. A pesar de esto, hemos conseguido soluciones factibles para redes considerablemente complejas en unos pocos segundos por lo que bajo nuestro punto de vista el desarrollo de esta memoria ha aportado valor al proyecto.

Para un futuro, se podría implementar el programa en C para tener así la capacidad de ejecutar redes más complejas. También, otras posibles líneas de continuación de este trabajo podrían consistir en implementar el modelo alternativo 2, que considera más posibilidades si cabe que nuestro modelo alternativo y aporta por tanto una solución más precisa. Incluso podríamos comparar los resultados obtenidos por ambos modelos.

# A

---

## Código en PYTHON

A continuación mostramos el código realizado en PYTHON. Se ha dividido en tres partes.

### A.1. Funciones auxiliares

```
1 # Este código hace referencia al módulo clase2 que más tarde utilizaremos
2
3 import numpy as np
4
5 def pump_psb(n):
6
7     dos_n = 2**n
8     bomb=np.zeros((dos_n,n), dtype=int)
9
10    for i in range(dos_n) :
11        for j in range(n) :
12            if( 2**(n-j-1) & i > 0) :
13                bomb[i,j]= 1
14
15    return bomb
16
17 #Esta función nos permite generar una matriz donde cada una de las filas de
18 #la matriz representa a una de las distribuciones de bombas posibles.
19 #El parámetro n se corresponde con el número de bombas que tendremos en
20 #nuestra red, y en función de este variará el tamaño de la matriz.
```

Listing A.1. Funciones auxiliares.

### A.2. Preproceso

```
1 from typing import get_args
```

```

2 from epanettools.epanettools import EPANetSimulation, Node, Link, Network,
  Nodes, Links, Patterns, Pattern, Controls, Control
3 from sko.GA import GA
4 import math
5 import numpy as np
6 import random
7 from ortools.linear_solver import pywraplp
8 from clase2 import *
9 import tempfile, os
10 import traceback
11 from time import time
12
13
14 inicio=time()
15
16 input_file="Net3.inp"
17 result_file="Resultados_programa.txt"
18 es = EPANetSimulation(input_file)
19
20 M=es.network.nodes
21 L=es.network.links
22 T=es.network.patterns
23 n_nodos=len(list(M))
24 n_links=len(list(L))
25 n_patts=len(list(T))
26 M_set=range(1,n_nodos+1)
27
28 es.run()
29
30 time_sec=es.ENgettimeparam(0)[1]
31 time_h=int(time_sec/3600)
32 times=range(time_h)
33
34 Min_pressure=-3
35 #Price of electricity in the different time slots
36 tariff=np.array([0.18854,0.18853, 0.18596 ,0.17906, 0.18072, 0.1865,
  0.18107, 0.19751, 0.23053, 0.21932, 0.25586,0.24912, 0.24729, 0.24192,
  0.19297,0.1948,0.20131 , 0.2097 , 0.27385 ,0.30105, 0.30634,
  0.28671,0.23121 ,0.21982])
37 pump=[]
38 tanks=[]
39 id_pump=[]
40
41 for m in M_set:
42     if M[m].node_type==2:
43         tanks.append(m)
44
45 for i in range(1,n_links+1):
46     if L[i].link_type==2:
47         pump.append(i)
48         id_pump.append(L[i].id)
49
50 number_pumps=len(pump)
51
52 n_tanks = len(tanks)

```

```

53
54 Tank_set = range(n_tanks)
55
56 npump=range(number_pumps)
57
58 bomb=pump_psb(number_pumps)
59
60 # We generate two arrays where we are going to save the minimum level, max
    level and initial level of each tank
61 max_level = np.zeros(n_tanks)
62 min_level = np.zeros(n_tanks)
63 ini_level = np.zeros(n_tanks)
64 for i in Tank_set :
65     max_level[i] = M[tanks[i]].results[Node.value_type['EN_MAXLEVEL']][0]
66     min_level[i] = M[tanks[i]].results[Node.value_type['EN_MINLEVEL']][0]
67     ini_level[i] = M[tanks[i]].results[Node.value_type['EN_TANKLEVEL']][0]
68
69 print('The different possibilities for the distribution of our pumps are :
    \n',bomb)
70
71 psb=len(bomb)
72
73 pump_posibilities=range( psb)
74
75
76 patterns=np.zeros((n_patts,time_h))
77
78 for n in range(n_patts):
79
80     for t in times:
81
82         patterns[n][t]=es.network.patterns[n+1][t+1]
83
84
85 #Now , we start the preprocess where we can save on different variables
    significant values of our network.
86
87 factible=np.ones((time_h,psb),dtype=int)
88
89 consumo=np.zeros((time_h,psb,number_pumps),dtype=float)
90
91 tank_lvl_changes=np.zeros((time_h,psb,len(tanks)))
92
93 f=os.path.join("/Users/Usuario/Documents/Visual Studio 2022/Prueba_python/
    PythonApplication2" , "h.inp")
94
95 es.ENsettimeparam(0, 3600)
96 # We change the value of the pattern of each node and the controls of the
    pumps and then we save the file and run the simulation with our new
    file
97
98 for t in times:
99
100     for n in range(n_patts):
101

```

```

102     es.ENsetpatternvalue(n+1,1,patterns[n][t])
103
104
105     for i in range(psb):
106
107         for c in npump:
108
109             if bomb[i][c]==0:
110
111                 es.ENsetcontrol(c+1,2,pump[c],0,0,0)
112
113             elif bomb[i][c]==1:
114
115                 es.ENsetcontrol(c+1,2,pump[c],1,0,0)
116
117
118         es.ENsaveinfile(f)
119         es.ENclose()
120         es = EPANetSimulation(f)
121         H=es.network.links
122         M2=es.network.nodes
123
124         try:
125             es.run()
126
127
128         except:
129             traceback.print_exc()
130             factible[t][i]=0
131             print('Excepcion')
132             es.ENclose()
133             es = EPANetSimulation(f)
134             es.run()
135             continue
136         a=0
137
138         for m in M_set:
139
140             if M2[m].results[Node.value_type['EN_HEAD']][1]<Min_pressure:
141
142                 a=1
143
144             if a==1:
145                 print('Segun EPANET, esta red con la distribucion',bomb[i], '
146                 es una red no factible .')
147
148                 factible[t][i]=0
149
150             else:
151
152                 for j in npump:
153
154                     consumo[t][i][j]=H[pump[j]].results[Link.value_type['

```



```

155         for b in Tank_set:
156
157             tank_lvl_changes[t][i][b]=(M2[tanks[b]].results[Node.
value_type['EN_HEAD']][1]-M2[tanks[b]].results[Node.value_type['EN_HEAD
']] [0])
158
159
160 #At the matriz consumo we have the energy spent by each pump each our and
we want to know how much energy have been spent using all of the pumps.
161 #For that reason we define consumo_real
162 consumo_real=np.zeros((time_h,psb))
163
164 h=np.zeros(number_pumps)
165
166 for t in times:
167
168     for i in range(psb):
169
170         for j in npump:
171
172             h[j]=consumo[t][i][j]
173
174             consumo_real[t][i]=sum(h)
175             h=np.zeros(number_pumps)
176
177 print('Los cambios de nivel de nuestros tanques seran: \n' ,
        tank_lvl_changes)
178 #Knowing the preece of the electricity each hour and the energy the pumps
spent we can calculate how much money the network is going to spend for
the differents distributions.
179
180 gasto=np.zeros((time_h,psb))
181
182
183 for t in times:
184
185     for i in range(psb):
186
187         gasto[t][i]=tariff[t]*consumo_real[t][i]
188
189
190 print('El consumo de cada bomba es : \n' , consumo)
191
192 tiempo = time() - inicio
193
194 print(f"El tiempo transcurrido en el preproceso en segundos es de : {
        tiempo}")

```

Listing A.2. Preproceso.

### A.3. Modelo Alternativo

```

1 #I define the variables x y l
2 x = { (i,t) : solver.BoolVar('X[%i, %i]' % (i,t)) for i in
      pump_possibilities for t in times }
3 l = { (j,t) : solver.NumVar(min_level[j], max_level[j], 'l[%i,%i]' % (j,t))
      for j in Tank_set for t in times }
4
5
6 #Objetive funtion
7 solver.Minimize(solver.Sum(gasto[t][i] * x[i,t] for i in pump_possibilities
      for t in times))
8
9 #Constrains
10 [ solver.Add( l[j,0] == ini_level[j]
11             + solver.Sum(tank_lvl_changes[0][i][j]*x[i,0] for i in
12             pump_possibilities )
13             ) for j in Tank_set]
14 [ solver.Add( l[j,t] == l[j,t-1]
15             + solver.Sum(tank_lvl_changes[t][i][j]*x[i,t] for i in
16             pump_possibilities )
17             ) for j in Tank_set for t in range(1,time_h)]
18 [solver.Add(l[j,time_h-1]>=ini_level[j]-EPS_L) for j in Tank_set]
19 [solver.Add(l[j,time_h-1]<=ini_level[j]+EPS_L) for j in Tank_set]
20
21
22 EPS_L=1.2 #At the second example EPS_L=12
23
24 for t in times:
25
26     for i in pump_possibilities:
27
28         if factible[t][i]==0:
29
30
31             [solver.Add(x[i,t]==0) ]
32
33
34 [ solver.Add(solver.Sum(x[i,t] for i in pump_possibilities ) == 1) for t in
      times]
35
36 solver.Solve()
37
38 print('Costo total = ', solver.Objective().Value() ," in ", solver.WallTime
      ()/1000, "seconds")
39
40 status = solver.Solve()
41 EPS=0.00001
42 if status == pywraplp.Solver.OPTIMAL:
43
44     for j in Tank_set:
45
46         print(f"l[{j},{time_h-1}]={l[j,time_h-1].solution_value()}")
47
48     #Writing the output file

```

```

49     b=0
50     a=0
51     with open(input_file, 'r+') as p:
52         lineas=p.readlines()
53
54         n_lineas=len(lineas)
55         for i in range(n_lineas):
56             linea_actual=lineas[i].split()
57
58             if len(linea_actual)>0 and linea_actual[0]=='[CONTROLS]':
59                 a=i
60
61             if len(linea_actual)>0 and linea_actual[0]=='[RULES]':
62                 b=i
63                 break
64
65
66     p = tempfile.TemporaryFile( dir= '/Users/Usuario/Documents/Visual
Studio 2022/Prueba_python/PythonApplication2')
67     h=open('out.inp', 'w+')
68
69
70     for i in range(a+1):
71         h.write(lineas[i])
72     #Adding optimals controls to the output file
73     h.write("\n")
74     for t in times:
75
76         for i in range(psb):
77
78             if x[i,t].solution_value()==1 :
79
80                 for c in npump:
81
82                     h.write('LINK '+str(id_pump[c])+' '+str(int(bomb[i
][c]))+' ' + 'AT TIME '+str(t)+"\n")
83
84     h.write("\n")
85     #Rewriting the information about our net in the output file
86
87     for i in range(b,n_lineas):
88         h.write(lineas[i])
89     h.close()
90
91     for j in times:
92
93         for i in pump_posibilities:
94
95             if x[i,j].solution_value() > EPS :
96                 print('x[%d,%d]=%f'%( i,j,x[i,j].solution_value() ))
97         print('\n')
98 else:
99     print('The problem doesn t have optimal solution')
100
101 print(solver.WallTime()/1000+tiempo)

```

**Listing A.3.** Modelo alternativo

---

## Bibliografía

- [1] Agencia de Protección Ambiental de Estados Unidos. Manual del usuario de EPANET 2.2. <https://www.epa.gov/water-research/epanet>.
- [2] A. Bondy, U. S. R. Murty, (2008). “Graph Theory”. Springer London
- [3] C. D’Ambrosio, A. Lodi, S. Wiese, and C. Bragalli, “Mathematical programming techniques in water network optimization,” *European J. of Operational Research*, vol. 243, no. 3, pp. 774 – 788, Jun. 2015.
- [4] J. J. Salazar González, (2001). Programación matemática. Ediciones Díaz de Santos.
- [5] M. K. Singh, V. Kekatos, (2020). “Optimal Scheduling of Water Distribution Systems”. *IEEE Transactions on Control of Network Systems*, Vol. 7.



# Optimal Scheduling of Water Distribution Systems

 **Sección de Matemáticas**  
Universidad de La Laguna

Raúl Siverio Quintero  
Facultad de Ciencias · Sección de Matemáticas  
Universidad de La Laguna  
alu0101203676@ull.edu.es

## Abstract

This paper presents the modeling of the optimization problem of water distribution systems, where customer demand must be satisfied trying to minimize operating costs. Based on the article “Optimal Scheduling of Water Distribution Systems”, we relax its non-linear model, generating an alternative model that simplifies the problem addressed. We implemented the alternative model in Python and tested its efficiency with different water distribution networks.

## 1. Introduction

Water distribution networks are a vital part of any water supply system around the world. These networks are a set of pipes, valves and other elements designed to transport water from a source of supply, such as a reservoir or a water treatment plant, to the homes and businesses. Water distribution networks are critical infrastructure for the well-being of people and the economy of cities, so their optimization is crucial.

In this work, we will present the problem of optimizing water distribution systems. At first we will consider the model of this problem given in the article already mentioned. Based on this nonlinear model, we will propose alternatives linear models to solve this problem. We'll implement one of these models in Python and check its efficiency. We will call this new model the alternative model.

With the alternative model implemented in Python and with the help of the Epanet software, we will test different water distribution networks in it.

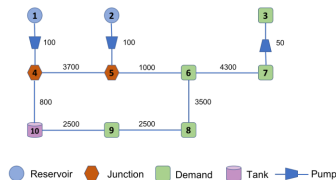


Figure 1: Example of a water distribution network

## 2. Approach and modeling of the problem

In our alternative model, we are going to consider all the possibilities for the pumps of the water distribution networks and the the different periods in which the simulation must be executed. So, since we will consider fixed speed pumps, the state of a pump will be open or closed. Assuming we have  $T$  periods and  $k$  pumps, we will have  $2^k \cdot T$  different possibilities.

Using the epanet software we will verify the feasibility of these possibilities period by period. When we have ruled out the infeasible possibilities and obtained the necessary information from the feasible ones, we can apply our model:

$$\min \sum_{t=1}^T \sum_{b \in PB} \pi_t C_{b,d(t)} x_b^t \quad (1)$$

subject to:

$$\sum_{b \in PB} x_b^t = 1 \quad t = 1, \dots, T \quad (2)$$

$$l_i^t = l_i^{t-1} + \Delta_{b,d(t)}^i x_b^t \quad t = 1, \dots, T, i = 1, \dots, n \quad (3)$$

$$l_i \leq l_i^t \leq \bar{l}_i \quad t = 1, \dots, T, i = 1, \dots, n \quad (4)$$

$$|l_i^T - l_i^0| \leq L \quad i = 1, \dots, n \quad (5)$$

$$x_b^t \in \{0, 1\} \quad t = 1, \dots, T, b \in PB \quad (6)$$

Where these restrictions verify that in each period we take only one of the possibilities aforementioned and also that a series of physical conditions on the tanks are met.

## 3. Computational results

We have tested the efficiency of our program with three different water distribution networks. First of all, we must point out that not all networks are the same size and complexity. Now, we're going to present our results on the following Table:

Table 1: Results

Network	Preprocessing Time	Total Time	Cost	Real Cost
1	0.97	0.99	7.45	7.51
2	10.18	11.2	943.38	936.80
3	1.87	1.93	645.63	614.17

The unit of time is seconds and the cost is in euros. The real cost refers to the cost provided by epanet when the optimal solution of our alternative model is introduced, which as we can see is approximate.

## References

- [1] Agencia de Protección Ambiental de Estados Unidos. Manual del usuario de EPANET 2.2. <https://www.epa.gov/water-research/epanet>.
- [2] A. Bondy, U. S. R. Murty, (2008). “Graph Theory”. Springer London
- [3] C. D'Ambrosio, A. Lodi, S. Wiese, and C. Bragalli, “Mathematical programming techniques in water network optimization,” European J. of Operational Research, vol. 243, no. 3, pp. 774 – 788, Jun. 2015.
- [4] J. J. Salazar González, (2001). Programación matemática. Ediciones Díaz de Santos.
- [5] M. K. Singh, V. Kekatos, (2020). “Optimal Scheduling of Water Distribution Systems”. IEEE Transactions on Control of Network Systems, Vol. 7.