Trabajo de Fin de Máster
Máster en Ingeniería Electrónica, Robótica y Automática

Optimización dinámica de redes de distribución de agua

Autor: Enrique Antonio Rodríguez González
Tutor: Daniel Limón Marruedo

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Trabajo de Fin de Máster
Máster en Ingeniería Electrónica, Robótica y Automática

Optimización dinámica de redes de distribución de agua

Autor:

Enrique Antonio Rodríguez González

Tutor:

Daniel Limón Marruedo

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Trabajo de Fin de Máster: Optimización dinámica de redes de distribución de agua

Autor: Enrique Antonio Rodríguez González

Tutor: Daniel Limón Marruedo

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2021

El Secretario del Tribunal

Agradecimientos

Me gustaría agradecer a mi familia y amigos por su apoyo durante este año y por conseguir que todo haya sido más llevadero incluso en estos tiempos difíciles.

Quiero agradecer también a mi tutor Daniel Limón Marruedo su ayuda durante la realización de este Trabajo de Fin de Máster, por el gran número de oportunidades que me ha otorgado en estos últimos años y por su interés genuino por mi futuro en general.

Enrique Antonio Rodríguez González

Escuela Técnica Superior de Ingeniería

Sevilla, 2021

Resumen

El objetivo de este Trabajo de Fin de Máster ha sido el de implementar una red de aguas para su simulación en Matlab, con el fin de desarrollar una herramienta que permita, partiendo de un modelo en EPANET de la red, proporcione un entorno en el que se pueda simular y controlar la red en todo momento.

El trabajo estará dividido en dos partes claramente diferenciadas: en primer lugar, se ha trabajado en la implementación de la red en cuestión en EPANET, la cual se utilizará, mediante la Toolbox de EPANET en Matlab, para simular la red en cada momento. A partir de esto, se implementará posteriormente un algoritmo RTO que calculará el punto de operación que optimiza una función de costes definida para la red.

Por otro lado, en la segunda parte del trabajo, se llevará a cabo la implementación de la misma red a través de la herramienta CasADi, con el objetivo de poder trabajar con una herramienta más potente para la implementación de la optimización del punto de operación.

A su vez, se implementará también un SSTO que permita adaptar la trayectoria óptima calculada mediante el RTO a un modelo lineal del sistema, con el objetivo de poder implementar un MPC que permita, a su vez, calcular la trayectoria óptima a seguir para llevar la red hasta su punto de funcionamiento óptimo.

Finalmente, se implementará un control de bajo nivel (mediante controladores PI) con el cual, partiendo de los valores de los caudales que proporciona como referencia el MPC, permitan determinar cuáles deberán ser los valores de apertura de las válvulas y de la bomba de la red.

Abstract

The main goal of this dissertation was to implement a water distribution network in Matlab, in a way that makes it possible to simulate and control the water network in different scenarios by using its EPANET model inside the Matlab environment.

Thus, the dissertation will be divided in two clearly differentiated parts: the first part will be focused on implementing a model of the water network in EPANET and using this model, along with the EPANET Toolbox available for Matlab, to simulate the network at any point in time in Matlab. With this, it's possible now to implement a Real Time Optimization (RTO) algorithm that determines what the optimal operating point is for the network, given a cost function.

On the other hand, the second part of the dissertation will be focused on implementing said network using the CasADi algorithmic differentiation tool, which proves to be a more powerful approach to the optimization problem.

At the same time, and with the objective of adapting the optimal operating point provided by the RTO to a lineal steady-state model of the water network, a steady-state target optimization (SSTO) will be implemented together with a Model Predictive Control (MPC) that will provide the optimal trajectory that the system would need to follow in order to reach the optimal operating point from any other operating point.

Finally, a low-level controller (in this case, a PI controller) will be implemented in order to determine what the manipulated variables should be in order to follow the trajectory provided by the MPC.

Índice

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

Las redes de distribución de agua son un claro ejemplo de sistemas que pueden verse muy beneficiados por una optimización de sus puntos de funcionamiento, pero que a su vez presentan una gran dificultad en la implementación de la misma, debido principalmente a la gran complejidad del comportamiento del sistema, que es no lineal, y al gran número de restricciones (tanto lineales como no lineales) que existen en el problema.

Con el objetivo de realizar una descripción detallada de cuál es exactamente el problema a optimizar, la primera sección de este Trabajo de Fin de Máster va a estar destinada a exponer cómo se ha aproximado el problema de modelar la red de aguas, tanto de manera genérica como para el caso de estudio, además de realizar una descripción de cuáles van a ser las restricciones bajo las cuales se ha de trabajar en una red de aguas, así como cuáles serán los objetivos económicos que tendrán mayor importancia a la hora de optimizar el funcionamiento de la red.

Con esto, se pasa ahora a hacer una introducción a las redes de agua y a detallar cuál es su modelo dinámico, para posteriormente describir cuál ha sido la red caso de estudio y los distintos elementos que la componen.

1.1. Introducción a las redes de aguas y modelo dinámico

Una red de distribución de agua se define como aquel conjunto de elementos interconectados que comunican las instalaciones de una empresa de abastecimiento de agua con sus consumidores, para satisfacer una determinada demanda de agua por parte de estos últimos.

Estas redes, por lo general, estarán compuestas por tres partes: unas instalaciones de captación de agua, una red de aducción que comunique las instalaciones de captación con las estaciones de tratamiento y almacenamiento del agua, y una red de distribución a través de la cual se pueda llevar el agua desde los depósitos de almacenamiento hasta las acometidas de los usuarios finales.

Por lo general, las redes de distribución de agua estarán compuestas por los siguientes elementos [1]:

- Reservorios o embalses que representan fuentes o sumideros infinitos de agua, y que tendrán una presión que se considerará constante.
- Depósitos o tanques en los cuales se almacene agua y cuyo nivel variará con el tiempo en función del caudal que entre o salga de los mismos a través de la siguiente ecuación:

  $$S\frac{dH}{dt} = Q,$$

  donde S representa la sección del depósito (que podrá ser constante o variar con la altura del mismo).
- Tuberías a través de las cuales circulará el agua desde un nodo a otro de la red. Estas tuberías tendrán pérdidas cuyo modelado, para tramos rectos, se realiza a través de la siguiente ecuación:

  $$\Delta h_t = r|Q|^{n-1}\,Q,$$

  donde Δh_t son las pérdidas de carga en la tubería, r es un coeficiente de rugosidad asociado a la tubería y n es un exponente que dependerá del tipo de coeficiente de fricción usado. Por lo general, los tres modelos de coeficientes de fricción más utilizados son el de Darcy-Weisbach, Hazen-Williams y Manning, cuyas expresiones son:

  - Darcy-Weisbach: $r = \frac{f8L}{D^5\pi^2 g},\ n = 2$, donde f es el coeficiente de Darcy-Weisbach.
  - Hazen-Williams: $r = \frac{10.67L}{D^{4.87}C^{1.852}},\ n = 1.852$, donde C es el coeficiente de Hazen-Williams.
  - Manning: $r = \frac{10.29n_r^2}{D^{5.33}}L,\ n = 2$, donde n_r es el coeficiente de Manning.

  Para este trabajo, el coeficiente a utilizar será el de Manning y se considerará que todos los tramos de tuberías de la red son rectos.

- Válvulas, elementos actuadores que regularán el caudal que pasará por una rama y que podrán tener diferentes funcionalidades:
    - Las válvulas reductoras o sostenedoras de presión mantendrán una consigna de presión aguas abajo o aguas arriba, respectivamente.
    - Las válvulas de rotura de carga proporcionarán un salto de presión constante entre la entrada y la salida de la válvula.
    - Las válvulas de regulación de caudal permiten controlar cuál será el caudal que pase por la misma.
    - Las válvulas de corte serán válvulas todo o nada que permitirán cerrar o abrir completamente las ramas en las que se sitúan.
    - Pr último, las válvulas de retención sólo permiten que el agua pase en un único sentido.

  Para este trabajo, las válvulas que más interés van a tener (por ser las únicas que aparecerán en la red caso de estudio) son las válvulas de regulación de caudal, las cuales de modelan a través de sus pérdidas de presión, al igual que las tuberías:

$$\Delta h_v = r|Q|Q$$

  Donde el coeficiente de rugosidad, r, sería la variable que controla la válvula para impedir o dejar que pase el agua, teniendo r un valor muy pequeño si está completamente abierta y muy grande si está completamente cerrada.

- Por último, las bombas hidráulicas permitirán fijar la consigna de caudal y el salto de presión en ramas de la red. Este caudal y presión estarán relacionados a través de la ecuación característica de la bomba, cuya expresión es:

$$\Delta h = AQ^2 + BQ\omega_r + C\omega_r^2,$$

  donde ω_r es la velocidad relativa de la bomba y los parámetros A, B y C dependerán del modelo específico de la bomba.

El comportamiento de estos elementos de la red se puede describir a través de un modelo dinámico, que estará definido por una serie de variables y ecuaciones que las relacionan. Las variables que definen el modelo dinámico de una red de aguas son:

- Los caudales Q en cada una de las ramas.
- Las presiones H de los nodos libres de la red.
- Las demandas D que puedan existir en los nodos libres.
- Las alturas H_x de los depósitos o tanques que forman parte de la red.
- Las alturas H_r de los embalses, que normalmente se considerarán constantes, ya que su variación tendrá una dinámica mucho más lenta que el resto de variables de la red.
- Las variables de actuación, u, con las que se podrá controlar la red y que serán las principales variables de decisión cuando se trate de optimizar el funcionamiento de la misma. Estas variables de actuación serán, por lo general, la velocidad relativa, ω_r, de las bombas de la red, o los coeficientes de rugosidad de las válvulas de regulación.
- Por último, los términos G de pérdidas las ramas de la red. Estos términos dependerán del tipo de rama (por ejemplo, la ecuación de una bomba o el coeficiente de rugosidad, diámetro y longitud de una tubería).

Estas variables estarán relacionadas entre sí para determinar el comportamiento dinámico de la red de aguas mediante tres tipos de ecuaciones:

- Ecuaciones de balance de masa en los tanques, que establecen la relación entre la variación en su altura

con el caudal entrante/saliente de los mismos. Estas ecuaciones, como ya se ha visto antes, tendrán la siguiente expresión:

$$\frac{dH_x}{dt} = BQ$$

Donde B es una matriz con dimensiones del número de tanques (n_t) por el número de caudales (n_r) que relaciona los caudales que entran o salen de los tanques con la variación en su altura a través de la inversa de su superficie.

- Ecuaciones de balance de masa en los nodos libres, que relacionan los caudales entrantes y salientes en cada uno de los nodos con las demandas que existen en los mismos:

$$AQ = D$$

Donde A es una matriz de incidencia con dimensiones del número de nodos libres (n_n) por el número de caudales. Los términos de la matriz serán 1, -1 ó 0 si el caudal entra, sale, o no afecta al nodo en cuestión, respectivamente.

- Ecuaciones de presiones en las mallas de la red, que relacionan los valores de presión en cada uno de los nodos de la red y las pérdidas en las ramas de la siguiente forma:

$$A^T H + G(Q,u) + A_r^T H_r + A_t^T H_x = 0,$$

donde la matriz G será la matriz de pérdidas en cada una de las ramas de la red, A_r será la matriz de incidencia de los caudales sobre los reservorios, y la matriz A_t será la matriz de incidencia sobre los depósitos de la red. Por otro lado, H, H_r y H_t serán los valores de las presiones en los nodos libres, los reservorios y los depósitos, respectivamente.

Con estas ecuaciones quedaría definido el modelo dinámico de una red de aguas, el cual se puede implementar para simular cuál será el comportamiento de la red en distintas condiciones de demandas y valores de las entradas manipulables. A continuación, se pasa a explicar cuál ha sido la red caso de estudio para este Trabajo de Fin de Máster y a definir su modelo dinámico.

1.2. Modelo dinámico de la red caso de estudio

La red de aguas con la que se ha trabajado a lo largo de este Trabajo de Fin de Máster viene descrita por el siguiente esquema:

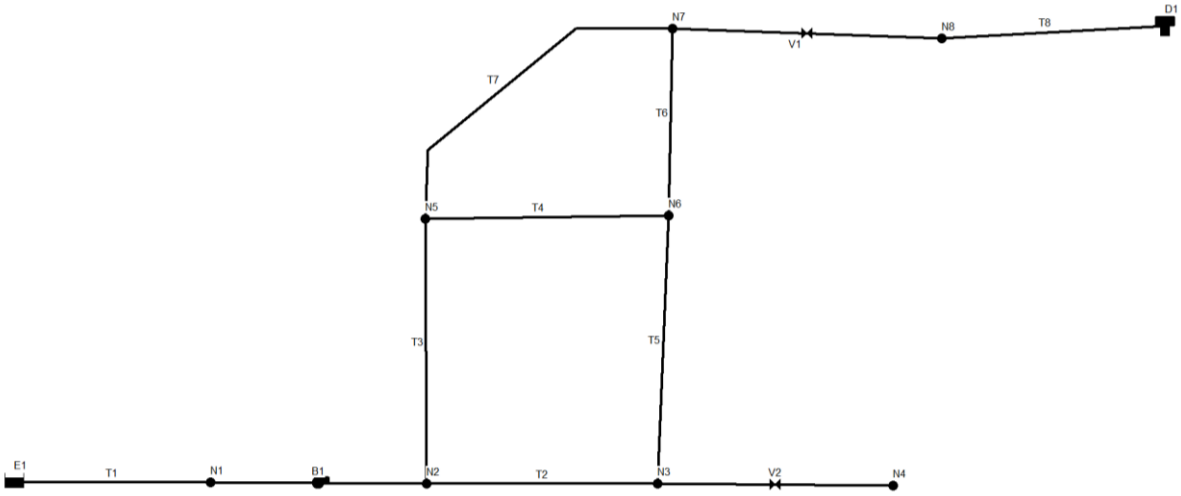

Figura 1 - Esquema de la red de aguas.

Como puede apreciarse en la figura, la red está compuesta por una serie de nodos, los cuales son:

- Un embalse (E1).
- Un depósito de agua (D1).
- Ocho nodos libres (N1-N8).

Por otro lado, las ramas que forman la red son las siguientes:

- Una serie de tuberías (T1-T8).
- Una bomba (B1), situada entre los nodos N1 y N2.
- Dos válvulas de regulación de caudal: la válvula V1, situada entre los nodos N7 y N8, y la válvula V2, situada entre los nodos N3 y N4.

La altura del embalse con la que se ha trabajado se ha considerado constante:

$$H_r = 0.5\ m$$

Por otro lado, las cotas de los nodos libres, así como la del tanque, han sido las siguientes:

$$Z_1 = 0.5\ m, \quad Z_2 = 3\ m, \quad Z_3 = 2\ m, \quad Z_4 = 2\ m, \quad Z_5 = 0.5\ m, \quad Z_6 = 4\ m,$$
$$Z_7 = 17\ m, \quad Z_8 = 17\ m, \quad Z_t = 38\ m$$

Por último, para el cálculo de los términos de pérdidas, en primer lugar, se dispone de la curva que relaciona la presión que proporciona la bomba con su caudal:

$$H_2 - H_1 = -55\omega_r^2 + 5.8 \cdot 10^{-4} Q^2$$

Por su parte, para el cálculo de los coeficientes de rugosidad de las tuberías, se dispone de sus diámetros, sus longitudes y sus coeficientes de Manning, que serán los mismos e iguales a 0.015. En base a estos datos se

pueden calcular sus coeficientes de rugosidad mediante la expresión del coeficiente de rugosidad a partir del coeficiente de Manning que se vio antes:

$$r_n = \frac{10.29 \cdot n_r^2}{D^{5.33}} \, L \cdot 10^{-6},$$

con L y D en metros y trabajando con caudales en lps.

Teniendo en cuenta que los diámetros y longitudes para cada una de las tuberías son los siguientes:

$$L_1 = 1\ m, \quad L_2 = 500\ m, \quad L_3 = 500\ m, \quad L_4 = 500\ m,$$
$$L_5 = 500\ m, \quad L_6 = 500\ m, \quad L_7 = 750\ m, \quad L_8 = 500\ m$$

$$D_1 = 300\ mm^2, \quad D_2 = 250\ mm^2, \quad D_3 = 250\ mm^2, \quad D_4 = 150\ mm^2,$$
$$D_5 = 200\ mm^2, \quad D_6 = 200\ mm^2, \quad D_7 = 150\ mm^2, \quad D_8 = 300\ mm^2$$

Los coeficientes de rugosidad resultantes son:

$$r_1 = 1.418 \cdot 10^{-6}, \quad r_2 = 0.001873, \quad r_3 = 0.001873, \quad r_4 = 0.0285,$$
$$r_5 = 0.00615, \quad r_6 = 0.00615, \quad r_7 = 0.04277, \quad r_8 = 7.088 \cdot 10^{-4}$$

Y teniendo en cuenta que los términos del vector G asociados a las tuberías tienen como expresión $G(Q) = r_n|Q_n|Q_n$, el vector $G(Q,u)$ de la red será entonces:

$$G(Q,u) = \begin{bmatrix} 1.418 \cdot 10^{-6} Q_1|Q_1| \\ 0.001873 Q_2|Q_2| \\ 0.001873 Q_3|Q_3| \\ 0.0285 Q_4|Q_4| \\ 0.00615 Q_5|Q_5| \\ 0.00615 Q_6|Q_6| \\ 0.04277 Q_7|Q_7| \\ 7.088 \cdot 10^{-4} Q_8|Q_8| \\ -55\omega_r^2 + 5.8 \cdot 10^{-4} Q_9^2 \\ u_1 Q_{10}|Q_{10}| \\ u_2 Q_{11}|Q_{11}| \end{bmatrix}$$

En cuanto al depósito de la red, teniendo en cuenta que el tanque tiene una superficie circular de diámetro $D = 20\ m$, que los caudales se tienen en lps, y que se trabaja con horas como unidad de tiempo, la matriz B será:

$$B = \left[0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ \frac{3.6}{S} \ 0 \ 0 \ 0\right] = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0.01146 \ 0 \ 0 \ 0]$$

Por otro lado, las demandas que se tienen para la red se han considerado de forma que el caudal aportado por la bomba se encuentre en torno a 80 lps, siendo algunas de ellas constantes en el tiempo, mientras que otras varían periódicamente, de forma que el vector de demandas D ha sido el siguiente:

$$D = \begin{bmatrix} 0 \\ 10 - 3\,\mathrm{sen}\left(\frac{\pi}{6}\right) \\ 5 \\ 40 + 2sen\left(\frac{\pi}{12}\right) \\ 5 \\ 20 + sen\left(\frac{\pi}{6}\right) \\ 0 \\ 10 \end{bmatrix}$$

Por último, las matrices de incidencia sobre los nodos libres, el embalse y el depósito (A, A_r y A_t, respectivamente) son:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix},$$

$$A_r = [-1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0], \qquad A_t = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1]$$

Con esto queda descrito el modelo dinámico de la red que se ha utilizado como caso de estudio. Ahora se pasa a realizar una descripción de las restricciones en el funcionamiento de la red que se deberán de tener en cuenta a la hora de plantear el problema de optimización del punto de operación.

1.3. Restricciones presentes en una red de distribución de agua y costes a optimizar

El control de una red de distribución de agua está sujeto a un gran número de restricciones que tienen como objetivo garantizar una operación que no sólo sea segura y satisfaga las necesidades de los clientes (tanto en cantidad como en calidad del agua), sino que, además, permite realizarlo de una forma respetuosa con el medio ambiente y minimizando posibles deterioros en la red.

De esta forma, en la operación de la red se deberán de cumplir una serie de restricciones, las cuales podrán comprender restricciones aplicadas a [1]:

- Presiones máximas en nodos de la red: $\boldsymbol{0 < H < H_{max}}$
- Niveles de mínimos y máximos en los tanques de la red: $\boldsymbol{H_x^{min} < H_x < H_x^{max}}$
- Límites de operación de los actuadores (bombas) de la red: $\boldsymbol{u_{min} < u < u_{max}}$
- Volúmenes mínimos en tanques que garanticen el suministro durante un tiempo determinado:

$$\boldsymbol{\sum V_x > \sum D \cdot T_{Sum}}$$

- Variaciones suaves en la actuación: $\boldsymbol{\Delta u < \Delta u_{max}}$

El cumplimiento de estas restricciones podrá ser necesario para garantizar un funcionamiento seguro y eficiente de la red. En el caso de la red que se ha utilizado como caso de estudio, las restricciones que se han fijado para

el problema de optimización han sido las siguientes:

- Presiones máximas en los nodos: $\mathbf{0 < H < 100\ mca}$
- Niveles mínimo y máximo en el tanque: $\mathbf{40 < H_x < 60\ m}$
- Límites de operación en la bomba: $\mathbf{0.3 < \omega_r < 1.5}$
- Límites en la variación de la consigna de la bomba: $\mathbf{\Delta\omega_r < 0.1}$

El objetivo de la labor de optimización de la red de aguas será el de minimizar los posibles costes asociados a la operación de la misma bajo estas restricciones. Estos costes tendrán principalmente una componente económica asociada al coste eléctrico del bombeo de agua para garantizar que se cumplen las demandas marcadas, pero también podrán comprender otros costes asociados a factores de seguridad, cuya función es mitigar la operación de la planta en condiciones que, aunque cumplan las restricciones planteadas, puedan no ser beneficiosas en períodos largos de tiempo.

Así, posibles costes que pueden asociarse a la operación de una red de aguas pueden ser:

- Coste económico asociado al bombeo: $\sum_{i=1}^{N_b} \boldsymbol{c_e(k)} \sum_{j=1}^{M_i} \frac{q_{ij}\Delta h_i}{\eta_{ij}(q_{ij},u_{ij},\omega_{ij})}$,

  donde $c_e(k)$ se corresponde con el coste unitario de la energía en un instante determinado, q_{ij} y Δh_i serían el caudal de la bomba ij y la variación de presión que proporciona el grupo de bombeo i, respectivamente, u_{ij} es la variable que indica si una bomba de un grupo de bombeo está conectada, ω_{ij} su velocidad relativa y η_{ij} su rendimiento.

- Costes asociados a variaciones en caudales asociados a las válvulas de regulación: $\sum_{i=1}^{N_v} |\Delta \boldsymbol{Q_i}|$

- Costes asociados a niveles de seguridad en los tanques:

$$\sum_{i=1}^{N_x} \left(\boldsymbol{H_{xi}^{min}} - \mathbf{min}\left(\boldsymbol{H_{xi}}, \boldsymbol{H_{xsi}^{min}}\right)\right)^2, \qquad \sum_{i=1}^{N_x} \left(\boldsymbol{H_{xi}^{max}} - \mathbf{max}\left(\boldsymbol{H_{xi}}, \boldsymbol{H_{xsi}^{max}}\right)\right)^2$$

  donde sólo se penalizaría, de forma cuadrática, si el nivel está por debajo del nivel de seguridad inferior o por encima del nivel de seguridad superior, siempre dentro del rango comprendido entre los mínimos y máximos absolutos de los tanques impuestos por las restricciones.

- Costes asociados a presiones en los nodos con demandas: $\sum_{i=1}^{N_n} \boldsymbol{H_i}$

De esta forma, los costes que se han establecido en un principio para la red han sido los siguientes:

- Coste asociado al bombeo: se ha considerado un rendimiento de la bomba constante, $\eta = 0.8$, y un coste económico c_e, de forma que:

$$\boldsymbol{J_b = c_e \frac{Q_9(H_2 - H_1)}{0.8}}$$

  Este coste se adaptará al optimizador que se esté utilizando. Para los optimizadores de Matlab cuando se trabaja con el simulador de EPANET, el coste se sustituirá por el siguiente:

$$\boldsymbol{J_b = 10 * c_e * \omega_r}$$

- Coste asociado a niveles de seguridad en los tanques: se han considerado para el tanque un nivel mínimo de seguridad de 45 m y un nivel máximo de seguridad de 55 m. De esta forma, el coste será:

$$J_{Hxns} = J_{Hxmin} + J_{Hxmax} = (45 - \min(H_x, 45))^2 + \left(55 - max(H_x, 55)\right)^2$$

Estos costes deberán normalizarse y ser ponderados en función de la importancia que cada uno de ellos tenga para la red en cuestión. Este proceso se detallará a fondo durante la implementación de cada uno de los optimizadores propuestos para este Trabajo de Fin de Máster.

2 JERARQUÍA DE CONTROL DE LA RED DE AGUAS

Como bien se ha comentado anteriormente, el objetivo final de este Trabajo de Fin de Máster es el de, aprovechando las distintas implementaciones de simuladores de redes de agua que se han realizado a lo largo del desarrollo del mismo, poder implementar distintos algoritmos que permitan calcular cuál deberá ser el punto de operación óptimo de una red de aguas que garantice que se cumplen una serie de restricciones y que se minimiza una función asociadas a los distintos costes que existen en la operación de la red.

Además, con el objetivo de poder llevar a cabo la operación eficiente de la red de aguas en base a los puntos de operación óptimos que se lleven a cabo en cada momento, se ha hecho uso de un control jerárquico dividido en varios niveles de control.

En el nivel más alto de la jerarquía de control, mediante un RTO (Real Time Optimization) se obtiene cuál será el punto de operación óptimo en base a los costes que se definieron en la sección 1 de este trabajo, a la vez que se trabaja bajo las restricciones definidas en el mismo.

La formulación del problema que intentará optimizar el RTO es simple: existe una función de coste, $f(x)$, la cual estará compuesta por los distintos costes asociados a la operación de la red de aguas, que se pretende optimizar bajo una serie de restricciones, las cuales se dividirá en dos tipos, restricciones de igualdad ($h(x)$) y restricciones de desigualdad ($g(x)$). Además, las variables de decisión x, que en el caso de la red serán la altura inicial del depósito y las consignas de las entradas en cada instante, se encontrarán dentro de un intervalo definido por un límite inferior y un límite inferior. Así, la formulación del problema será la siguiente [2]:

$$\begin{aligned} & \min_x f(x) \\ s.a. \quad & h(x) = 0, \\ & g(x) \leq 0, \\ & x_{min} \leq x \leq x_{max} \end{aligned}$$

El RTO trabaja optimizando un modelo no lineal de la red de aguas en una escala de tiempo relativamente grande (del orden de días o semanas). De este modo, la salida del RTO (es decir, la trayectoria óptima que seguirá el punto de operación de la red en un intervalo grande de tiempo) no se corresponderá exactamente con la trayectoria óptima del modelo lineal de la planta que se utilizará en el MPC implementado para llevar la red a dicha trayectoria óptima, tanto en términos de las entradas de la red, como de las salidas que éstas producen.

Estas discrepancias entre el modelo real de la red y el modelo lineal que utiliza el MPC pueden llevar a la aparición de offsets en régimen permanente, lo cual significaría una operación no óptima de la planta, o incluso hacer que la planta intente trabajar en una región no factible. Por ello, será necesario implementar un paso intermedio entre el RTO y el MPC que convierta las consignas obtenidas mediante el RTO en consignas factibles para el modelo lineal dinámico del MPC.

Este paso intermedio se incorporará a través de un SSTO (steady-state target optimization), cuya implementación, cuando se utiliza un modelo dinámico lineal de la red, se puede plantear como un problema de programación cuadrática (QP) que intenta minimizar la diferencia entre la trayectoria del RTO y la que tendría el modelo lineal de la red. La descripción del problema se detallará más adelante cuando se trate la implementación del SSTO y del MPC para la red que se ha utilizado como caso de estudio.

Por otro lado, a partir de las consignas adaptadas al modelo dinámico lineal de la red que se proporciona el SSTO, el MPC obtendrá cuál deberá ser la trayectoria óptima desde el punto de operación actual de la planta hasta llegar a dicha trayectoria óptima. Esto se plantea de igual manera como un QP con una estructura muy similar a la del SSTO cuando éste optimiza trayectorias (como es el caso), pero utilizando ahora la trayectoria que proporciona el SSTO y con algunas diferencias en la formulación. El MPC proporcionará las consignas de caudal que la red deberá alcanzar en cada momento para garantizar que se llega a la trayectoria marcada por el SSTO de forma óptima.

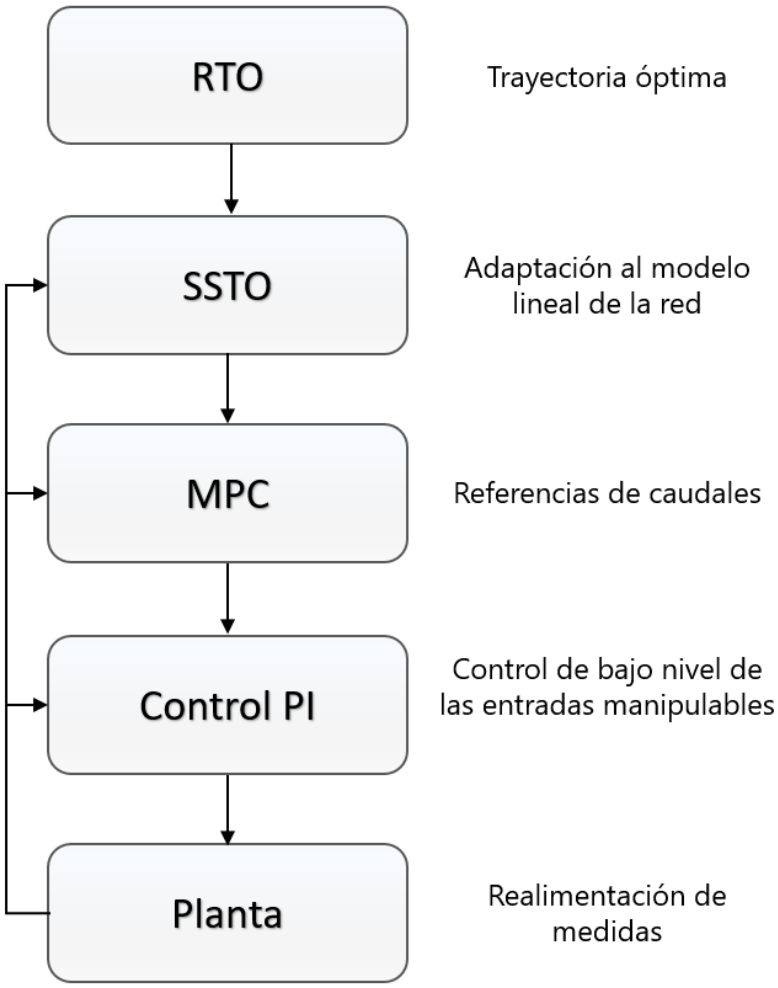

Figura 2 – Esquema de la jerarquía de control.

Por último, se han diseñado controladores de bajo nivel, más específicamente controladores PI, que llevarán a cabo el control de los caudales o alturas asociados a cada una de las entradas manipulables (en este caso, la bomba y las dos válvulas de regulación). El diseño de los parámetros de cada uno de los controladores PI se detallará más adelante cuando se hable de su implementación.

Una vez detallado el modelo dinámico de la red de aguas, así como las diferentes restricciones y costes que van a existir durante su operación y de la jerarquía de control que se ha seguido, se pasa ahora a la descripción de EPANET, el software de simulación de redes de agua que se ha empleado para la primera parte del trabajo.

3 EPANET

EPANET es una aplicación software desarrollada por la Agencia de Protección Ambiental de Estados Unidos (EPA) en 1993 como una herramienta de análisis del comportamiento del agua en distintas aplicaciones de redes de distribución de agua. El software permite implementar, mediante una interfaz gráfica, una red de distribución de agua desde cero utilizando como base los parámetros que se han descrito en la sección anterior, y además realizar un análisis de cuál será el comportamiento de la red ante diferentes posibles situaciones de demanda por parte de los consumidores y del efecto que cada una de las variables de actuación tendrán en la misma.

En este Trabajo de Fin de Máster se ha trabajado partiendo de una red diseñada en EPANET cuya simulación, mediante una herramienta desarrollada por el Centro de Investigación para Sistemas Inteligentes y Redes de la Universidad de Chipre, conocida como "EPANET-MATLAB Toolkit", se ha llevado a cabo dentro de la interfaz de Matlab, con el objetivo de combinar la facilidad y precisión que EPANET proporciona a la hora de llevar a cabo la implementación y simulación de redes de agua con las diferentes herramientas relacionadas con la optimización de sistemas y de recopilación y tratamiento de datos que Matlab proporciona.

De esta forma, se pasa ahora a dar una primera vista de cuál será el funcionamiento de la interfaz gráfica de EPANET y de cómo es el procedimiento de implementación de una red de aguas, para luego pasar a describir el Toolbox de EPANET en Matlab con el que se ha trabajado, y la implementación de la red que se ha utilizado como caso de estudio en la misma.

3.1. Interfaz gráfica de EPANET

El funcionamiento de la interfaz gráfica de EPANET es sencillo: para definir la red sobre la que se va a trabajar, se da la opción de introducir en la misma una serie de nodos que, como ya se ha visto en el apartado anterior, podrán ser nodos libres, embalses o depósitos. Para cada uno de estos tipos de nodos se tienen una serie de parámetros configurables, como pueden ser la cota de cada nodo libre, embalse o depósito, la altura del embalse o las alturas y diámetros de los depósitos.

La demanda en cada uno de los nodos libres, definidas a partir de una "demanda base" que podrá modificarse a partir de un patrón de demanda que multiplicará la demanda por un factor determinado en cada instante de tiempo.

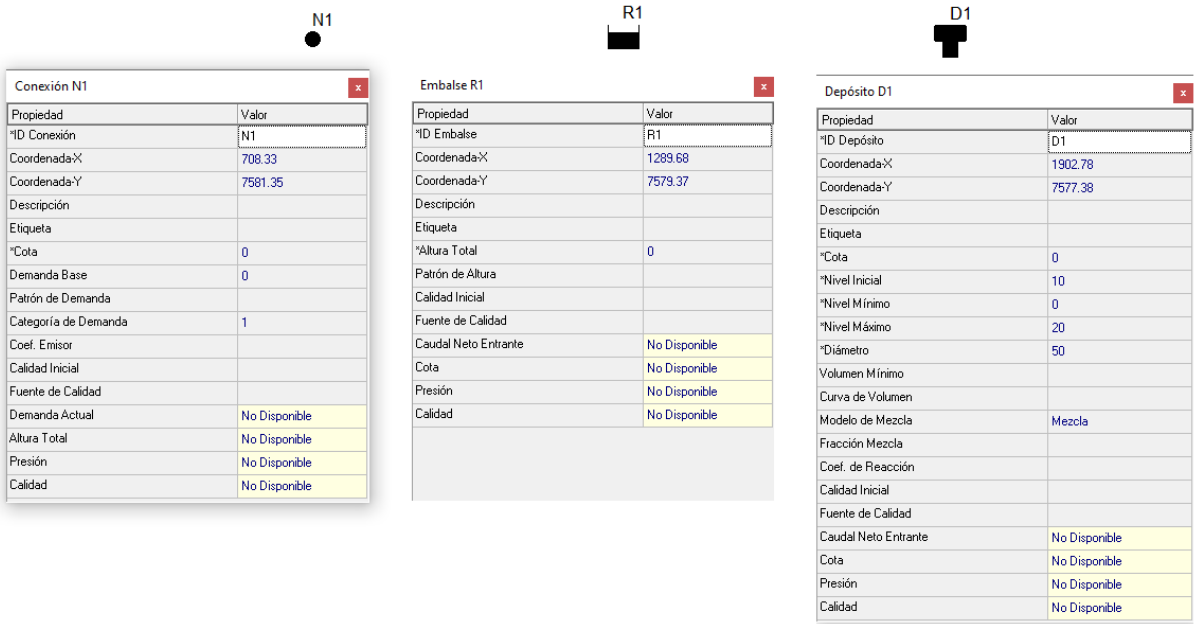

Figura 3 - Tipos de nodos y sus parámetros en EPANET.

Por otro lado, una vez definidos dos o más nodos, se dispone una serie de elementos de unión para estos nodos, como pueden ser tuberías, bombas o válvulas de distintos tipos, cada uno de los cuales con distintos parámetros configurables. Estos parámetros incluyen el diámetro, longitud y coeficiente de rugosidad (ya sea el coeficiente Darcy-Weisbach, coeficiente de Manning o coeficiente de Hazen-Williams) de las tuberías, la curva altura-caudal de una bomba y su velocidad relativa, o el tipo de válvula (reductora, sostenedora, de ruptura de carga, limitadora de caudal, de regulación o de propósito general), así como su diámetro y su consigna, que se corresponderá con su coeficiente de pérdidas menores.

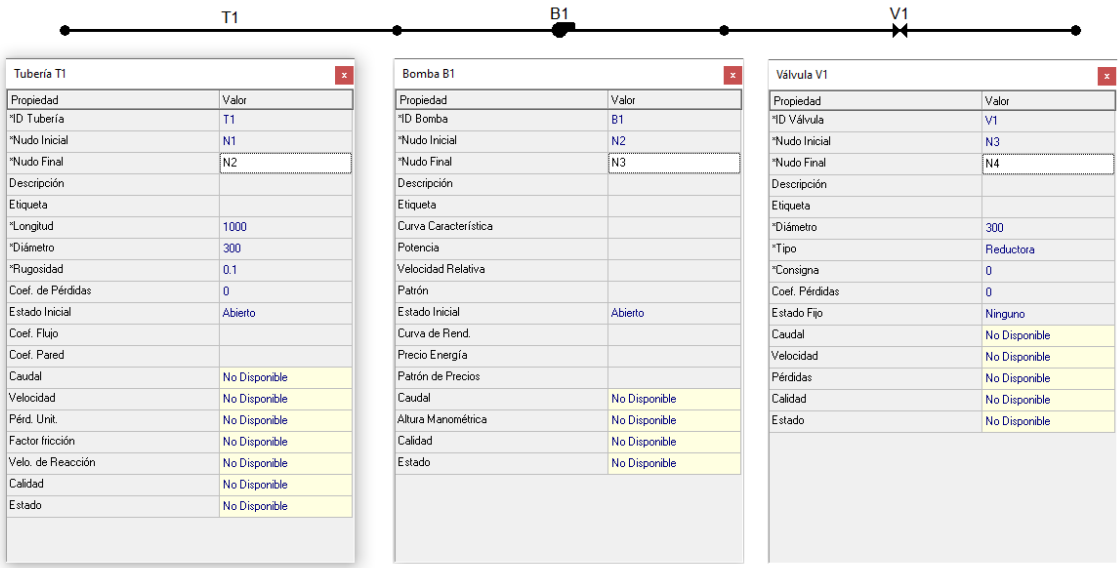

Figura 4 - Tipos de elementos de unión en EPANET.

En el caso de las curvas, por ejemplo, asociadas a la relación altura-caudal de una bomba, existe una sección dedicada a curvas donde se podrán definir las mismas a partir de una serie de puntos:

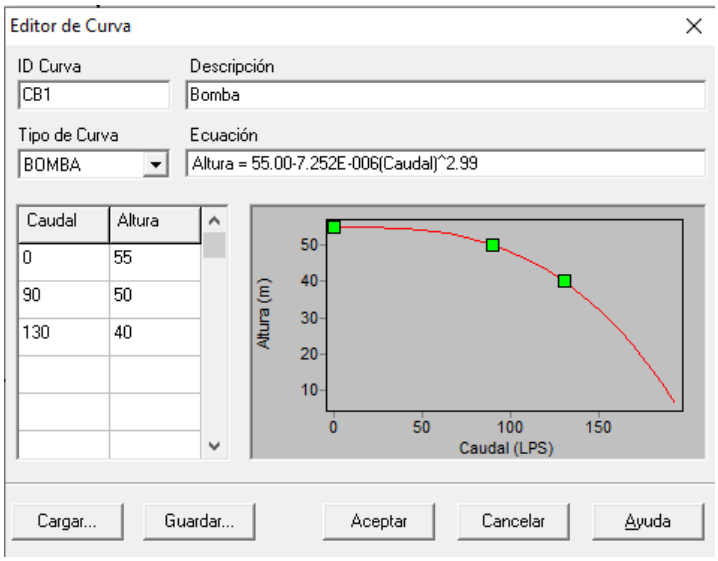

Figura 5 - Definición de curvas en EPANET.

En cuanto a la simulación de redes, EPANET permite configurar determinados parámetros como la duración de la simulación, el intervalo de cálculo hidráulico o la precisión con la que trabajar. En base a estos parámetros, a las demandas que se asocien a cada uno de los nodos de la red, y a los valores de las variables de actuación, se

podrá simular la red y obtener (en el caso de que el problema tenga solución) valores de variables asociadas a cada uno de los nodos o elementos de unión, como pueden ser las alturas en los nodos o los caudales que circulan por las tuberías, en cada instante del intervalo de simulación.

Por ejemplo, simulando la red durante 196 horas con unas demandas constantes:

$$D = \begin{bmatrix} 0 \\ 10 \\ 5 \\ 40 \\ 5 \\ 20 \\ 0 \\ 0 \end{bmatrix}$$

Y con unos valores constantes para las variables de actuación:

$$\omega_r = 1, \qquad u_1 = 0, \qquad u_2 = 0$$

La evolución de la altura del depósito a lo largo del intervalo de tiempo, así como el caudal que para por la bomba, que se obtienen de la simulación son los siguientes:

Figura 6 - Evolución la altura del tanque y caudal en la bomba en una simulación ejemplo en EPANET.

EPANET dispone además de la posibilidad de exportar cualquier red creada a un archivo '.inp' en el cual se recogen cada uno de los elementos que forman la red con sus respectivos parámetros de configuración. De hecho, partiendo de este archivo ha sido como se ha decidido implementar la red en Matlab mediante la toolbox mencionada antes, ya que, aunque la propia Toolbox permite implementar una red a través de una serie de comandos, la interfaz gráfica de EPANET permite hacerlo de una forma mucho más intuitiva y rápida.

Con esto, se puede pasar ahora a la descripción del funcionamiento de la herramienta de EPANET en Matlab, así como a detallar el procedimiento que se ha seguido para la implementación de la red y su simulación.

3.2. EPANET-Matlab Toolkit

Como ya se ha mencionado antes, la toolbox de EPANET-Matlab [3] permite trabajar con una red diseñada en EPANET y utilizar todas muchas de las herramientas de las que el software dispone para la simulación de redes de agua.

La última versión de la herramienta puede ser descargada desde su página en GitHub. El enlace a esta página, en el momento en el que este trabajo es escrito, es el siguiente:

https://github.com/OpenWaterAnalytics/EPANET-Matlab-Toolkit

El toolkit implementa prácticamente todas las funcionalidades relacionadas con la simulación de la red de las que se dispone en EPANET dentro de la interfaz de Matlab, además de permitir modificar una determinada red de EPANET guardada en un archivo '.inp' para añadir o quitar elementos de una red o cambiar parámetros de los mismos o de la simulación en sí. A grandes rasgos, los comandos de los que se dispone en el toolkit se podrían dividir en las siguientes categorías en base a su funcionalidad:

- Los comandos 'get' permiten obtener información de la red, ya sea información con respecto a los distintos elementos que la forman o los parámetros de cada uno, parámetros de la simulación, parámetros de la configuración de la red, como, por ejemplo, las unidades con las que se trabaja, o medidas de la red tras la simulación, entre muchas otras funcionalidades. Algunos ejemplos de comandos tipo 'get' podrían ser:
    - getLinkDiameter devuelve el diámetro de cada una de las tuberías que componen la red.
    - getNodeBaseDemands devuelve las demandas base de cada uno de los nodos.
    - getUnits devuelve todas las unidades con las que está trabajando la red, como pueden ser los caudales, alturas, presiones o tiempo.
- Los comandos 'set' permiten cambiar parámetros de la red o de cada uno de sus elementos, de manera que se puedan hacer modificaciones permanentes en la red (por ejemplo, cambiando el diámetro de una tubería) o temporales (cambiando, por ejemplo, la demanda de un nodo en un determinado instante de tiempo). Posibles comandos tipo 'set' son:
    - setNodeElevations permite cambiar la cota de un determinado nodo.
    - setNodeTankInitialLevel permite cambiar el nivel inicial de un depósito.
    - setNodeBaseDemands permite cambiar las demandas en uno o más nodos en cada instante.
- Los comandos 'add' y 'delete', que permiten añadir y quitar elementos, curvas o patrones de demandas asociados a la red:
    - Con deleteLink, por ejemplo, se podría eliminar una determinada rama de la red.
    - Mediante el comando addNodeTank se podría añadir un nuevo depósito en la red.
- Otros comandos que implementan distintas funcionalidades, como por ejemplo aquello relacionados con la ejecución de las simulaciones en sí o inicialización de los valores de la red, entre otras funcionalidades:
    - El comando plot, por ejemplo, dibuja en un plot de Matlab cómo se vería la red en EPANET.
    - El comando initializeHydraulicAnalysis carga los valores de los parámetros iniciales de la red.

- El comando runHydraulicAnalysis simula la red durante el tiempo de integración que se haya configurado y actualiza el estado de la red al final del intervalo de integración.

La forma en la que se trabaja con redes de EPANET en la toolbox es a través de un tipo de Matlab Class denominada 'epanet', en la que se recogen una serie de elementos que definen la red, que son los siguientes:

- Las propiedades del modelo de la red.
- Propiedades estáticas.
- Funciones que el usuario puede utilizar directamente en Matlab.
- Funciones internas que la toolbox utiliza para hacer llamadas a EPANET.

Para poder crear un objeto con clase 'epanet', se asigna una variable de Matlab a un archivo '.inp' de una red creada en EPANET a partir de un comando del tipo:

```
r = epanet('Red_Epanet.inp')
```

De esta forma, se generará un objeto 'r' que poseerá todas las propiedades de la red en el archivo 'Red_Epanet.inp' y que se podrá llamar, bien para obtener valores de los parámetros de los distintos elementos que componen la red, como también para simular la evolución de la red durante un determinado intervalo de tiempo y obtener los valores de las variables internas de la red en cada instante.

Para la simulación de la red se ha desarrollado una script de Matlab, 'Simul_EPANET.m' que hace uso de la toolbox y de las herramientas que se acaban de comentar para obtener el estado en el que la red se encuentra en cualquier momento en base a las variables de actuación y demandas que se definen para la misma.

Este script recibe como entradas la red 'r', que será el objeto de clase 'epanet' asociado al archivo '.inp' de la red en EPANET, el tiempo 'tsim' de simulación de la red que, por lo general, será 1 hora, ya que éste es el intervalo que se ha escogido para trabajar en el RTO, la altura inicial del depósito, 'Hxini', el vector de demandas 'D' existentes en el instante en el que se quiere simular, y el vector de entradas, 'uk', que será:

$$u_k = [\omega_r \quad u_2 \quad u_1],$$

donde ω_r es la velocidad relativa de la bomba, u_2 la consigna de la válvula V2, y u_1 la consigna de la válvula V1 (las cuales se dan como términos del coeficiente de rugosidad de cada una de ellas).

En primer lugar, el script fija el intervalo de informe a 5 minutos y la duración de la simulación al valor 'tsim' que se pasa al script, y se inicializa el modelo para llevar a cabo un análisis hidráulico.

```
r.setTimeSimulationDuration(tsim);
r.setTimeReportingStep(300);
r.openHydraulicAnalysis;
r.initializeHydraulicAnalysis;
```

A continuación, se fija el nivel inicial del depósito al nivel 'Hxini', así como los valores de las entradas manipulables y las demandas que se pasan al script para el instante determinado en el que se ejecuta.

```
r.setNodeTankInitialLevel(Hxini'-38);
if not(isempty(uk))
    r.setLinkSettings(9,uk(1));
    r.setLinkSettings(10,(300^4/8.2627e4)*uk(2));
    r.setLinkSettings(11,(300^4/8.2627e4)*uk(3));
end
r.setNodeBaseDemands(D');
```

Donde los valores de las consignas de las válvulas se adaptan para proporcionar las consignas en términos de coeficiente de pérdidas menores que espera EPANET. La correspondencia entre el coeficiente de rugosidad y el de pérdidas menores se obtiene a partir del siguiente desarrollo:

Partiendo del modelo de pérdidas menores:

$$\Delta h = \frac{K}{2g} v^2$$

Donde K es el coeficiente de pérdidas menores y v la velocidad del agua que circula por la válvula en m/s.

Por otro lado, el coeficiente de rugosidad de la válvula será:

$$\Delta h = r q^2$$

Teniendo en cuenta que $q = vS = \frac{\pi}{4} v D^2$, y usando q en lps y D en mm^2, se llega a la siguiente expresión:

$$\Delta h = \frac{K}{2g} \frac{16}{\pi^2} \frac{10^6}{D^4} q^2 = \frac{8.2627 \cdot 10^4 K}{D^4} q^2$$

De manera que la relación entre el coeficiente de rugosidad y el de pérdidas menores será:

$$K = \frac{D^4 r}{8.2627 \cdot 10^4}$$

Una vez definidas las entradas y demandas de la red, se simula la red durante el tiempo de simulación y se van guardando los valores de las alturas y los caudales de cada uno de los nodos y tuberías de la red, y una vez alcanzado el instante final, se cierra el análisis hidráulico y se devuelven estos valores como vectores de salida 'H' y 'Q':

```
tstep=300;H=[];Q=[];
while (tstep>0)
    H=[H; r.getNodeHydaulicHead];
    Q=[Q; r.getLinkFlows];
    tstep=r.nextHydraulicAnalysisStep;
end
r.closeHydraulicAnalysis;
```

Este script será la base utilizada para simular la red durante la ejecución de los algoritmos utilizados por el RTO, así como para comprobar la evolución de los caudales y alturas de la red cuando se implementen los controladores de bajo nivel, y el seguimiento de las referencias marcadas por el MPC.

Se pasa ahora a la descripción del proceso de implementación del RTO haciendo uso del simulador de EPANET y de los scripts de los que éste hace uso para el cálculo de costes y comprobación de cumplimiento de restricciones.

4 IMPLEMENTACIÓN DEL RTO BASADO EN EL SIMULADOR DE EPANET EN MATLAB

4.1. Algoritmos de optimización utilizados

Para la implementación del RTO se ha planteado utilizar distintos algoritmos de optimización de Matlab, con el objetivo de comparar los resultados que cada uno de ellos proporciona y comprobar cuál de ellos es el que va a ser más eficiente a la hora de resolver el problema de optimización. En este caso, se ha implementado el problema para resolverlo mediante la función 'fmincon' de Matlab, que incorpora distintos algoritmos de optimización, de entre los cuales se han usado:

Método de punto interior

Recordando lo visto en la sección anterior, donde se definía la formulación del problema de optimización como [4], [5]:

$$\begin{aligned} &\min_x f(x) \\ s.a.\ \ &h(x) = 0 \\ &g(x) \leq 0 \end{aligned}$$

Los métodos de punto interior o de barrera reformulan el problema para trabajar sin restricciones de desigualdad, haciendo que éstas pasen a ser un término de la función de costes que se hace muy grande cuando alcanza la región no factible del problema, de manera que la nueva formulación será:

$$\begin{aligned} F_\mu(x) = \min_x f(x) + \mu \sum_{i=1}^{n} \ln\big(g_i(x)\big) \\ s.a.\ \ h(x) = 0 \end{aligned}$$

De esta forma, el algoritmo de punto interior, partiendo de un valor μ_k en un instante determinado, que se corresponde con las variables de decisión x_k, itera siguiendo dos pasos:

1. Se actualiza el valor de μ_{k+1} a uno más pequeño que μ_k.

2. Se ejecuta un algoritmo que minimiza $F_{\mu_{k+1}}(\cdot)$, empezando en x_k, hasta que se encuentra un punto x_{k+1} cercano a $x^*(\mu_{k+1}) = argmin\ F_{\mu\_(k+1)}(\cdot)$, siendo este punto uno donde se cumple $g(x) \leq 0$. Hecho esto se vuelve a iterar pasando al instante $k+1$.

Y se itera hasta que se cumplen las condiciones que se pasan al algoritmo (longitud mínima del paso, tolerancia de las restricciones y tolerancia de optimalidad).

Algoritmo SQP (Sequential Quadratic Programming)

La idea detrás de los métodos SQP es aproximar el problema de optimización NLP en cada instante x_k mediante un subproblema de programación cuadrática (QP), cuya solución se utiliza para construir un un nuevo valor x_{k+1} con el que se iterará en el instante inmediatamente posterior, de tal manera que la secuencia x_k tiene a un mínimo local del NLP conforme k tiende a infinito.

La formulación del subproblema QP es tal que [6]:

$$\min_{d_x} \nabla f(x_k)^T d_x + \frac{1}{2} d_x^T H_k d_x$$
$$s.a. \quad \nabla h(x^k)^T d_x + h(x^k) = 0$$
$$\nabla g(x^k)^T d_x + g(x^k) \leq 0$$

Donde $d_x = x - x^k$ y la matriz H_x es el hessiano del lagrangiano $L(x) = f(x) + \sum_{i=1}^{n} \lambda_i \cdot g_i(x)$.

Este subproblema se soluciona utilizando un algoritmo QP y la solución de éste se utiliza para formar el valor de la siguiente iteración:

$$x_{k+1} = x_k + \alpha_k d_k,$$

Siendo α_k la longitud del paso, que se obtiene mediante una búsqueda en línea.

Nuevamente, al igual que ocurría con el algoritmo de punto interior, se itera hasta que se cumplen las condiciones que se pasan al algoritmo para determinar si el punto calculado es suficientemente bueno.

Por último, también se ha hecho uso de la función de algoritmo genético de Matlab, que proporciona un método heurístico para encontrar, al menos, un mínimo local para la función de costes de la red de aguas que cumpla las restricciones marcadas para la misma.

Algoritmo genético

El funcionamiento de los algoritmos genéticos está basado [7] en el proceso de selección natural que caracteriza la evolución biológica. El algoritmo modifica continuamente una población de soluciones (que serían distintas soluciones factibles para el problema planteado para la red de aguas que se proporcionen al algoritmo), a la vez que, en cada paso, se eligen individuos de dicha población (soluciones individuales) para producir una nueva generación, proceso el cual es conocido como 'crossover'.

Además, este tipo de algoritmos implementan un mecanismo de alteración de los resultados de cada crossover conocido como 'mutation', a través del cual se realizan pequeños cambios, imitando las mutaciones que se producen en el ADN de los seres vivos, en este caso modificando ligeramente la solución obtenida mediante las combinación de otras dos soluciones factibles, con la idea de crear más variedad en la población con la que se está trabajando y conseguir que el algoritmo avance el nuevas direcciones que puedan llevar a la solución óptima.

Esta serie de mecanismos aleatorios que los algoritmos genéticos implementan hacen que su uso sea atractivo para sistemas que son pocos conocidos o inclusos sistemas caja negra, donde se desconoce por completo la relación entre las entradas y las salidas.

4.2. Implementación del RTO en Matlab

La implementación del RTO en Matlab con el toolbox de EPANET se ha realizado a través del script 'RTO.m', que a su vez hará uso de otros scripts que simularán la evolución de la red para cada una de las consignas que el RTO genera, y que calcularán los términos de la función de costes y comprobarán si se cumplen las restricciones en todo momento.

En este script, en primer lugar, se inicializan las variables del modelo que se van a utilizar, primero mediante el script 'Parametros_Modelo.m', en el que se establecen las variables fijas del modelo, como pueden ser los valores mínimos y máximos de las entradas y del depósito, el tiempo de muestreo o la altura del embalse, y a continuación, en el propio script del RTO se definen también algunas variables que podrán variar dependiendo del punto de operación del que se parta, como la definición de las demandas en el tiempo o la altura del depósito y entradas de la solución inicial que se pase a los algoritmos de optimización.

La solución inicial de la que se ha partido ha sido un punto de equilibrio en el que, habiendo elegido unos

determinados valores de velocidad relativa de la bomba y apertura de las válvulas, se ha simulado durante un período largo de tiempo en EPANET hasta que se ha comprobado que el nivel del depósito se estabilizaba en un punto de equilibrio, que será el que se ha utilizado como nivel inicial.

Es conveniente mencionar que en esta simulación las demandas han sido constantes, correspondiéndose con la parte constante de las demandas que se van a utilizar realmente para las condiciones con las que se va a trabajar con la red. No obstante, la diferencia entre ambas no va a ser muy grande en términos del nivel del depósito al final del período en el que se optimiza en comparación con el nivel inicial.

En el caso de los optimizadores implementados en fmincon, se ha utilizado una única solución inicial, la cual es la siguiente:

$$H_{xini,1} = 48.35$$

$$u_{ini,1} = [\omega_r \quad u_2 \quad u_1] = [1 \quad 0.001 \quad 0.001],$$

donde el valor de las variables manipulables se mantiene constante durante todo el intervalo de tiempo para el que se optimiza.

En el caso del optimizador con algoritmo genético, se han aportado dos soluciones iniciales más, con el objetivo de partir de una población con más de un individuo. Estas soluciones iniciales se han obtenido mediante el mismo método que la anterior, y sus valores son:

$$H_{xini,2} = 53.7$$

$$u_{ini,2} = [1.05 \quad 0.001 \quad 0.001]$$

$$H_{xini,3} = 42.71$$

$$u_{ini,3} = [0.95 \quad 1.02e-5 \quad 1.02e-5]$$

Una vez definidos los valores de la solución inicial de la red y sus demandas en el tiempo, se reorganiza la matriz de entradas manipulables para convertir ésta en un solo vector columna con los valores de las entradas en el período N para el que se optimiza, ya que las funciones 'fmincon' y 'ga' de Matlab van a exigir que tanto la solución inicial como los límites inferiores y superiores de las variables de decisión sean un único vector.

Por otro lado, también se definen las variaciones máximas que existen para la variable asociada a la velocidad relativa de la bomba. Este parámetro será una restricción lineal sobre dicha variación que, además de garantizar una operación suave de la planta, también ayudará a la convergencia de los algoritmos de optimización, ya que variaciones más pequeñas de dicha velocidad se traducen en una probabilidad menor de que el algoritmo entre en una región no factible del problema de la que no pueda salir. Estas restricciones vienen definidas como:

$$-a \leq \Delta\omega_{r,k} \leq a \; \forall k \in N,$$

siendo N el número de instantes totales (en este caso, el número de horas) para los que se va simular y calcular la trayectoria óptima, y siendo a el valor de la variación mínima y máxima de la velocidad relativa de la bomba, y que por lo general será un valor que se encuentre entre 0.05 y 0.15, con el objetivo de que las variaciones sean pequeñas.

Una vez definidas las restricciones lineales, se definen las referencias a las funciones de cálculo de costes (en el script 'CalculosCostes.m') y de las restricciones no lineales de la red (en el script 'Restricciones.m'), que tanto 'fmincon' como 'ga' utilizarán para comprobar el coste de cada una de las soluciones que proporcionan, así como comprobar que se cumplen las restricciones no lineales. A continuación, se realizará una descripción más detallada del funcionamiento de estos scripts.

Función de cálculo de costes

Esta función recibe como argumentos las variables de decisión del problema de optimización, las demandas en el tiempo, los parámetros del modelo y el objeto tipo 'epanet' asociado a la red de aguas y devuelve un valor J correspondiente a un coste asociado a la operación de la red bajo las variables de decisión mencionadas.

Para obtener dicho el coste J, se hace uso de otro script, 'Prediccion', en el que se simulará la red durante el

período en el que se quiere optimizar para obtener los valores de las variables algebraicas y del nivel del depósito a lo largo de dicho período.

De esta forma, para cada instante se actualiza el valor de J en función de:

- J_{bk}, coste asociado a la bomba en el instante k, que se calculará mediante la expresión:

$$J_{bk} = 10 * c_e * U_t^2(1 + 3k)$$

donde c_e es el valor de un coste económico asociado al coste eléctrico de la bomba y los valores $U_t(1 + 3k)$ se corresponderán con los valores de velocidades relativas que se han elegido como variables de decisión para el problema.

La expresión de c_e en el tiempo que se ha elegido es:

$$c_e(k) = 1 + 0.3 \sin\left(\frac{2\pi(k-1)}{48}\right)$$

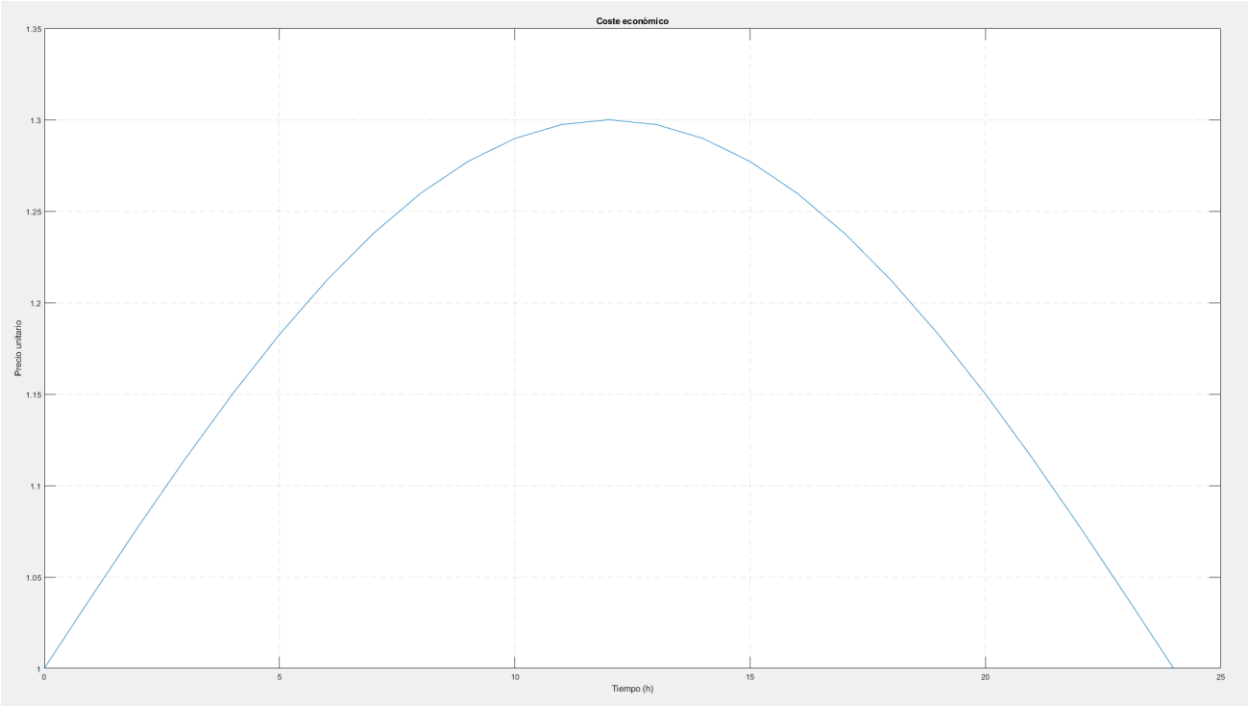

Figura 7 - Coste económico en el tiempo.

- J_{tsk1} y J_{tsk2}, costes asociados al cumplimiento de los niveles de seguridad mínimo y máximo del depósito, respectivamente, cuyas expresiones son:

$$J_{tsk1} = \frac{1}{10} * \left(H_{xsk}^{min} - \min\left(H_{xsk}^{min}, H_{xk}\right)\right)^2,$$

$$J_{tsk2} = \frac{1}{10} * \left(H_{xsk}^{max} - \max(H_{xsk}^{max}, H_{xk})\right)^2,$$

De forma que el coste total en cada instante será:

$$J_k = J_{k-1} + J_{bk} + J_{tsk1} + J_{tsk2}$$

Por último, se añade un término cuadrático adicional asociado a la variación de la altura en el depósito en el

instante $N+1$ con respecto a la altura inicial, con el objetivo de evitar que el optimizador haga cambios bruscos en la red al final del período lo cual, aunque conseguiría reducir el coste total, podría llevar la red a un punto de operación desfavorable para el siguiente período. Este coste tiene la siguiente expresión:

$$J = J_N + 10^4 * \left(H_{xini} - H_x(N+1)\right)^2$$

Donde se ha ponderado de manera que, para una diferencia de 0.01 entre el nivel inicial y el nivel final del depósito, el coste asociado sea 1.

El motivo por el cual se ha utilizado este término de coste adicional, en lugar de plantear la restricción de periodicidad como una restricción de igualdad, tiene que ver con la convergencia de los optimizadores. Se ha probado experimentalmente que los optimizadores con los que se ha trabajado eran incapaces de encontrar una solución factible al problema forzando $H_x(N+1) = H_{xini}$, mientras que el uso de este término de coste, los optimizadores tienden a hacer que el nivel final se parezca al nivel inicial, y se consigue que el problema converja con bastante mayor facilidad.

Implementación de restricciones no lineales del problema

El script de implementación de las restricciones no lineales recibe los mismos argumentos que la función de costes, excepto por el parámetro T, y genera como salida dos vectores: un primer vector en el que se encuentran los resultados de comprobar las restricciones de desigualdad, y otro en el que se encuentran los resultados de comprobar las restricciones de igualdad. En el caso de las restricciones de desigualdad, éstas deberán de estar formuladas de forma que, si se cumple la restricción, el valor generado sea negativo. Por otro lado, para las restricciones de igualdad, el valor generado deberá ser cero. El motivo por el que se requiere del uso de esta formulación es simplemente porque así esto es precisamente lo que comprobarán las funciones 'fmincon' y 'ga' para comprobar si se cumplen las restricciones.

En este caso, sólo se han utilizado restricciones de desigualdad. Si bien, como se había mencionado antes, idealmente se debería de hacer planteado la restricción de periodicidad del problema como una restricción de igualdad, al hacerlo de esta forma era muy difícil conseguir que el problema convergiese.

Así, las restricciones de desigualdad que se han impuesto al problema de optimización han sido:

- Niveles mínimo y máximo del tanque: $R_{1,k} = H_{xk} - H_x^{max},\ R_{2,k} = H_x^{min} - H_x$
- Caudal que sale del embalse positivo: $R_{3,k} = -Q_{1,k}$
- Alturas de los nodos libres mayores que la altura mínima:

$$R_{4,k} = -H_{1,k}, \quad R_{5,k} = -H_{2,k}, \quad R_{6,k} = -H_{3,k}, \quad R_{7,k} = -H_{4,k}$$
$$R_{8,k} = -H_{5,k}, \quad R_{9,k} = -H_{6,k}, \quad R_{10,k} = -H_{7,k}, \quad R_{11,k} = -H_{8,k}$$

- Alturas de los nodos libres menores que la altura máxima:

$$R_{12,k} = 70 - H_{1,k}, \quad R_{13,k} = 70 - H_{2,k}, \quad R_{14,k} = 70 - H_{3,k}, \quad R_{15,k} = 70 - H_{4,k}$$
$$R_{16,k} = 70 - H_{5,k}, \quad R_{17,k} = 70 - H_{6,k}, \quad R_{18,k} = 70 - H_{7,k}, \quad R_{19,k} = 70 - H_{8,k}$$

De esta forma, si se cumplen todas las restricciones, el script devolverá un vector con sólo valores negativos, y otro vector, correspondiente a las restricciones de igualdad, vacío.

Una vez definidas las funciones de costes y de restricciones no lineales, se forma el vector inicial, compuesto por la altura inicial del depósito y las entradas manipulables de la solución inicial.

Además del vector inicial, se definen los vectores con los límites superiores inferiores y superiores para las variables de decisión, que serán:

$$z = [T \ \ H_{xini} \ \ \omega_r\,(1) \ \ u_2\,(1) \ \ u_1\,(1) \ \ \omega_r\,(2) \ \ u_2\,(2) \ \ u_1\,(2) \ \ \ldots \ \ \omega_r\,(N) \ \ u_2\,(N) \ \ u_1\,(N)]$$
$$z_{min} = [0 \quad 40 \quad 0.2 \quad 10^{-7} \quad 10^{-7} \quad 0.2 \quad 10^{-7} \quad 10^{-7} \quad \ldots \quad 0.2 \quad 10^{-7} \quad 10^{-7}]$$

$$z_{max} = [0.1 \quad 60 \quad 1.2 \quad 10^7 \quad 10^7 \quad 1.2 \quad 10^7 \quad 10^7 \quad \dots \quad 1.2 \quad 10^7 \quad 10^7]$$

Una vez definidos los límites inferiores y superiores, los scripts con la función de costes y restricciones y las soluciones iniciales, el script ya puede llamar a las funciones de los optimizadores de Matlab con éstos y con las opciones correspondientes como argumentos. A continuación, se va a hacer una descripción de los resultados que se han obtenido para cada uno de los distintos algoritmos utilizados y una comparación de los mismos.

4.3. Resultados obtenidos por los optimizadores de fmincon y algoritmo genético

Los resultados presentados a continuación recogen tanto la evolución de los caudales y alturas de cada uno de los nodos y tuberías de la red de aguas, como la evolución de la altura del depósito y de las entradas manipulables.

En primer lugar, se van a presentar los resultados obtenidos mediante el uso del optimizador de 'fmincon', primero con el algoritmo de punto interior, y después con el algoritmo sqp. Posteriormente, se mostrarán los resultados que se han obtenido con el algoritmo genético y, finalmente, se hará una comparación entre ambos para ver cuál ha obtenido mejores resultados.

Resultados del algoritmo de punto interior

Caudales

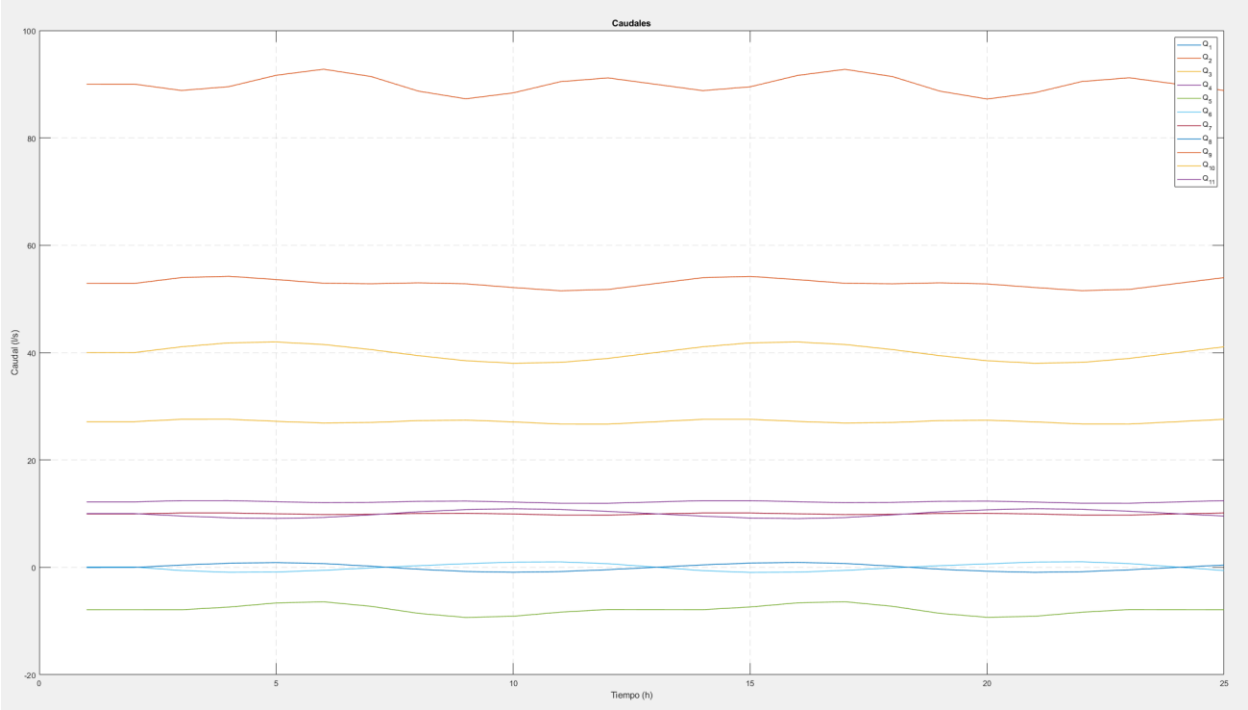

Figura 8 – Evolución de los caudales con el algoritmo de punto interior.

Alturas

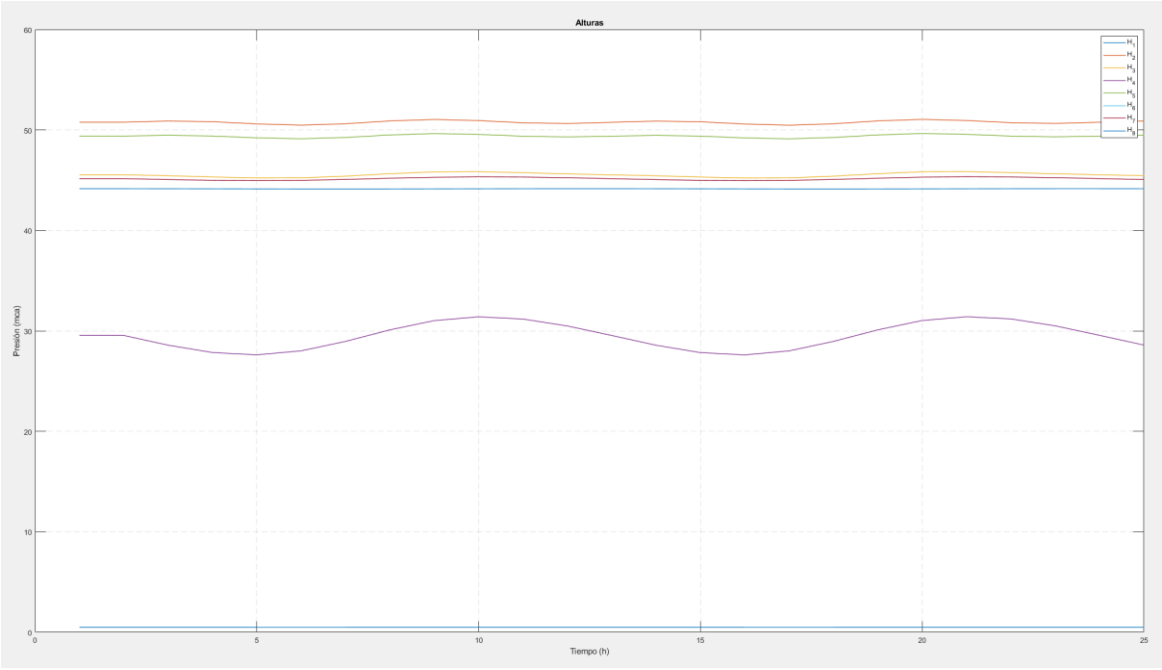

Figura 9 – Evolución de las alturas con el algoritmo de punto interior.

Nivel del depósito

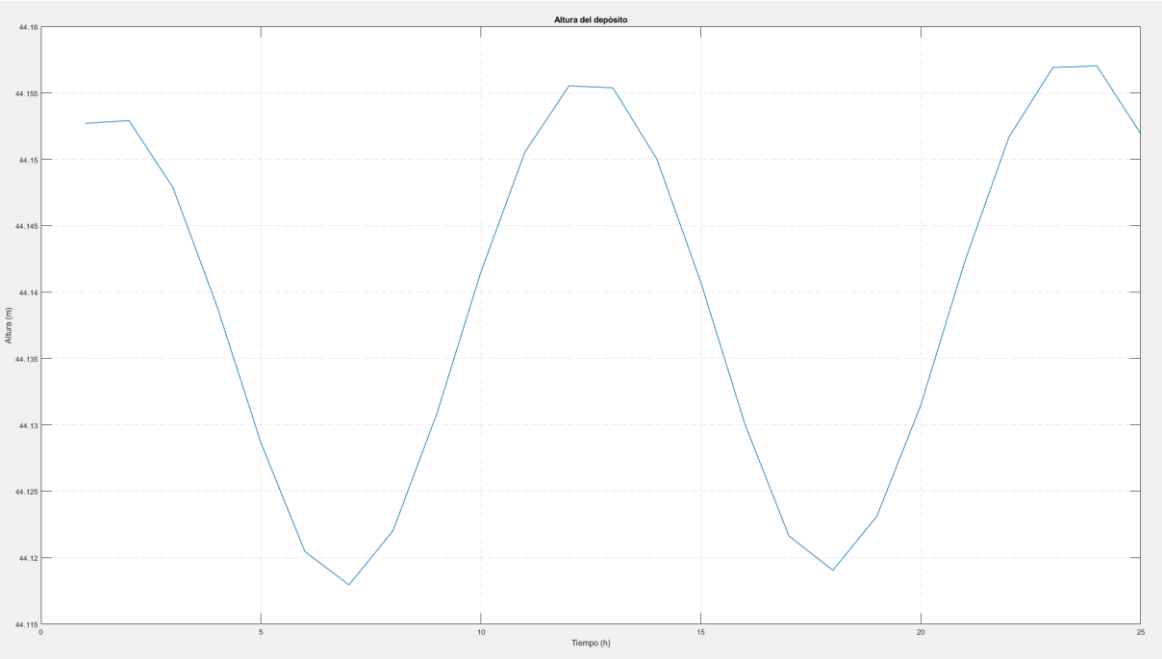

Figura 10 – Evolución de la altura del depósito con el algoritmo de punto interior.

Entradas manipulables

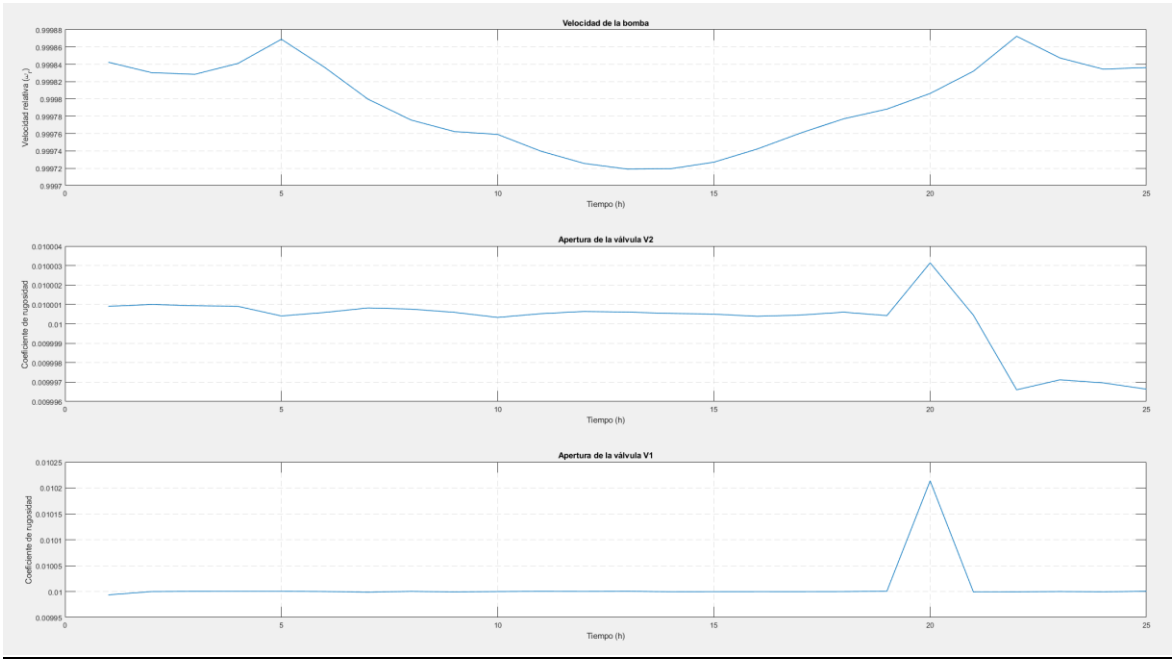

Figura 11 - Evolución de las entradas manipulables con el algoritmo de punto interior.

Como puede observarse, los resultados no son muy buenos. El algoritmo apenas hace variar la velocidad de la bomba a lo largo del período de 24 horas, y aunque parece que sí que intenta disminuir ésta acorde a las variaciones en el coste eléctrico, dichas variaciones son muy pequeñas.

En este caso, el coste calculado para el óptimo que ha encontrado el algoritmo de punto interior es $J = 251.13$.

Resultados del algoritmo sqp

Caudales

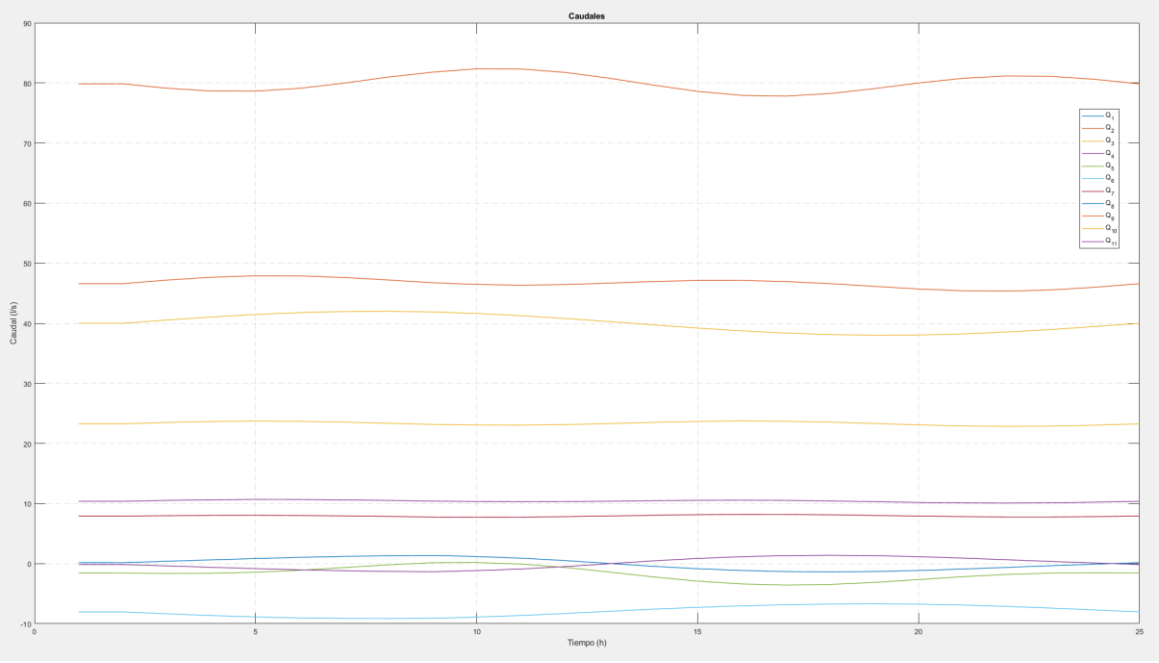

Figura 12 - Evolución de los caudales con el algoritmo sqp.

Alturas

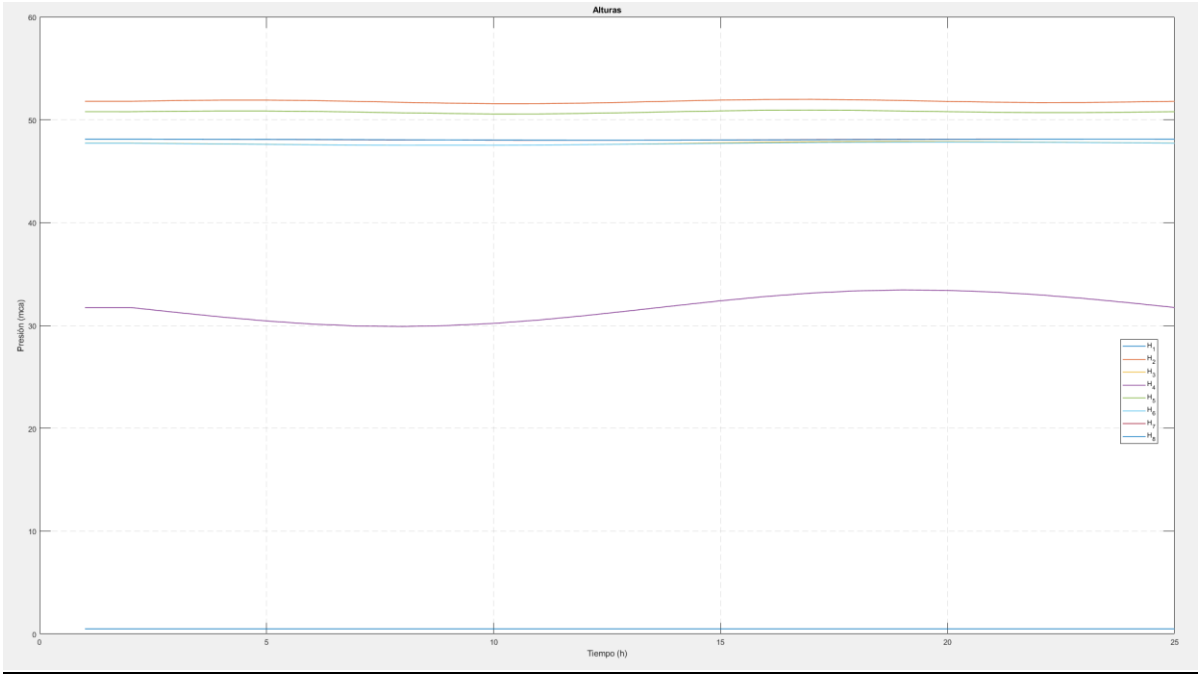

Figura 13 - Evolución de las alturas con el algoritmo sqp.

Nivel del depósito

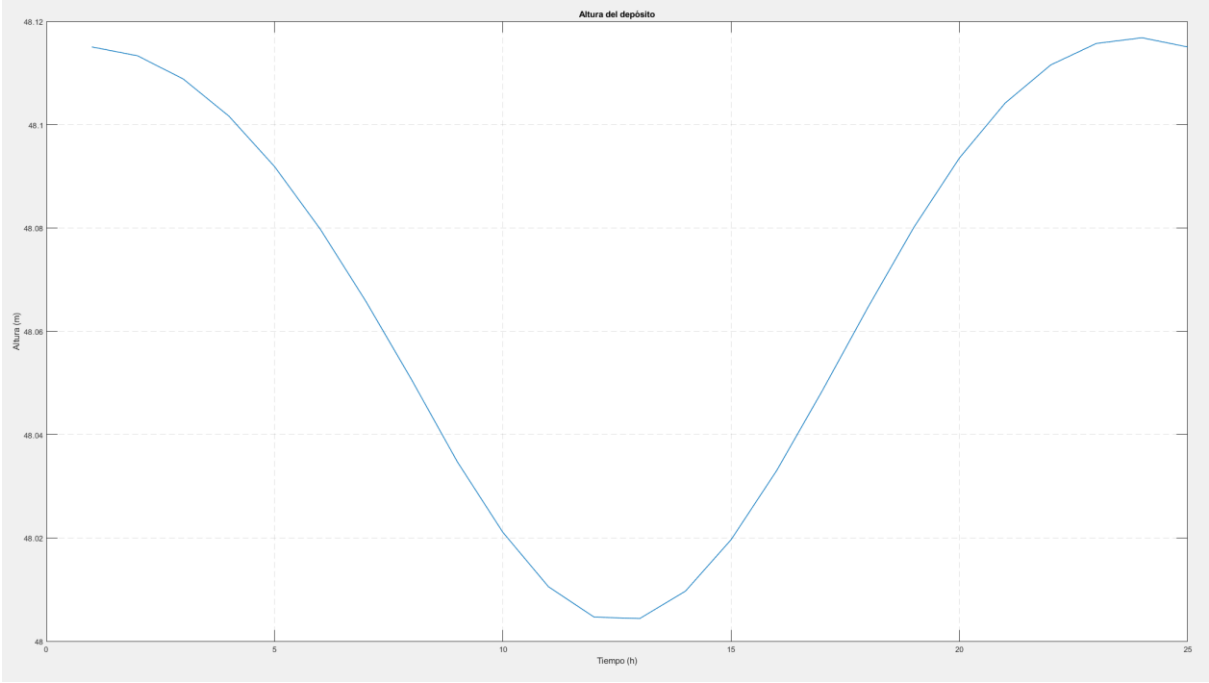

Figura 14 - Evolución de la altura del depósito con el algoritmo sqp.

Entradas manipulables

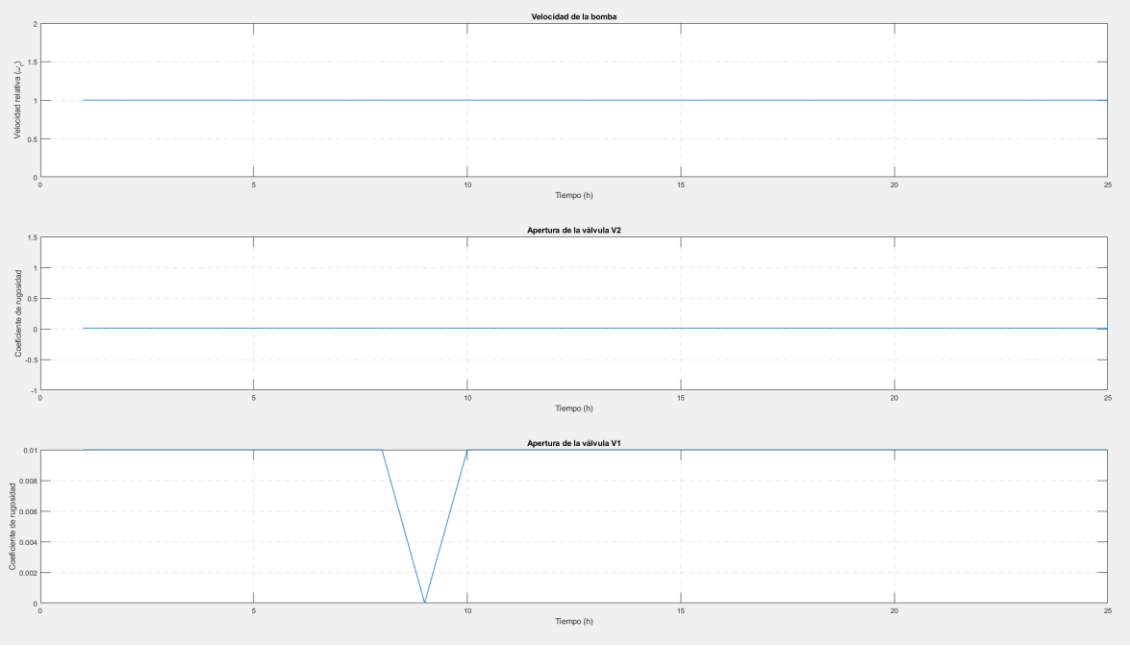

Figura 15 - Evolución de las entradas manipulables con el algoritmo sqp.

Los resultados obtenidos para el algoritmo sqp son todavía peores que los del algoritmo de punto interior. En este caso, el optimizador prácticamente no mueve el punto de operación del punto inicial, y sólo hay variaciones muy pequeñas en las variables de actuación en instantes determinados de tiempo.

El coste calculado para el punto que ha obtenido el algoritmo ha sido $J = 254.238$

Resultados del algoritmo genético

Caudales

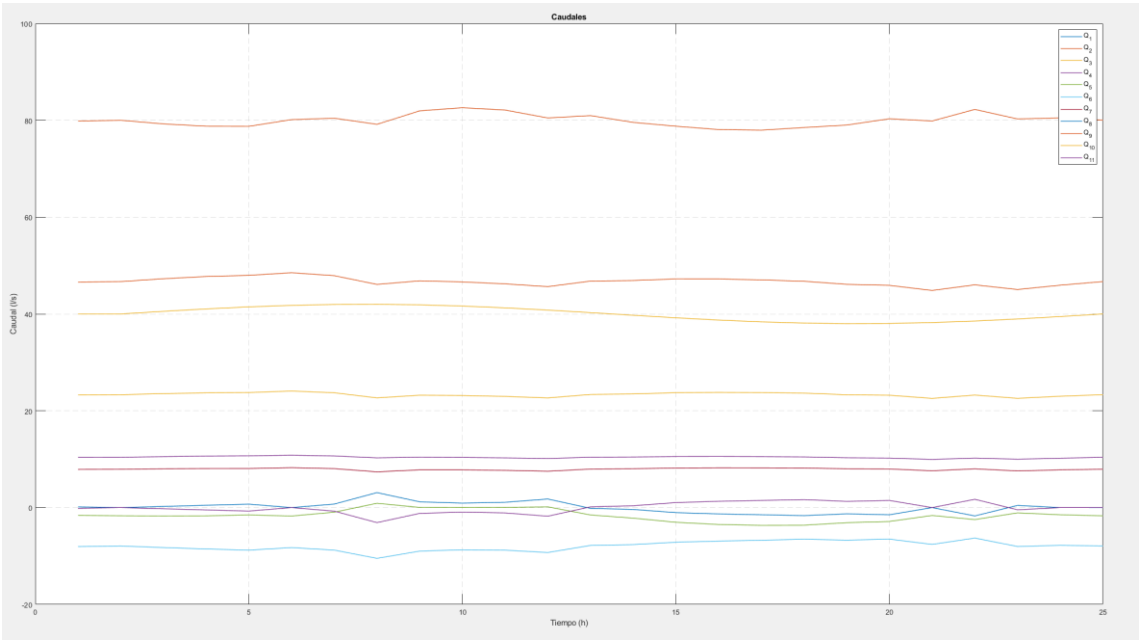

Figura 16 - Evolución de los caudales con el algoritmo genético.

Alturas

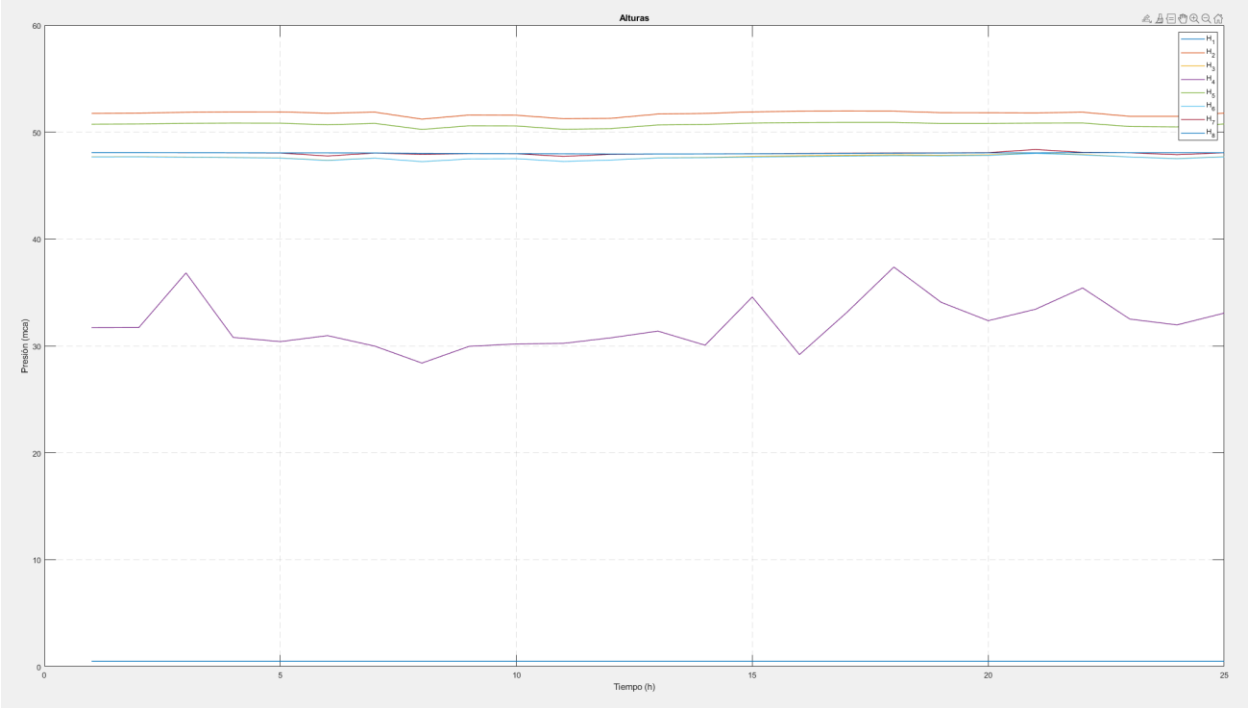

Figura 17 - Evolución de las alturas con el algoritmo genético.

Nivel del depósito

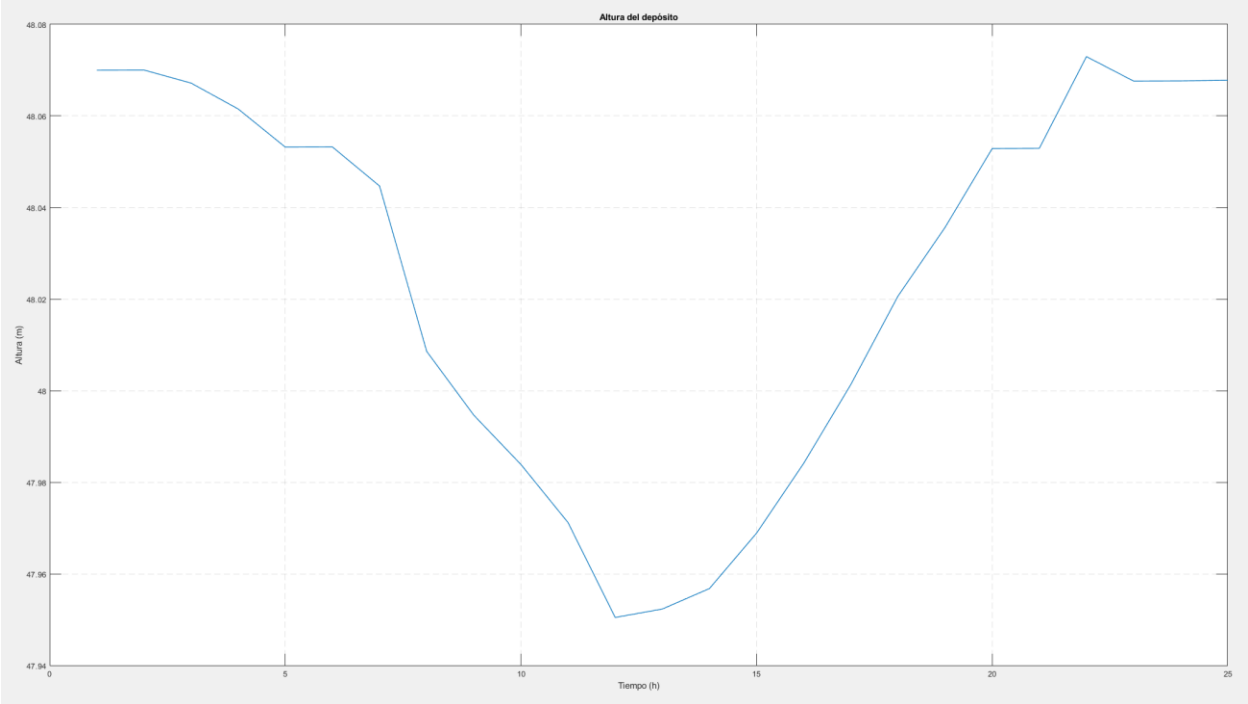

Figura 18 - Evolución de la altura del depósito con el algoritmo genético.

Entradas manipulables

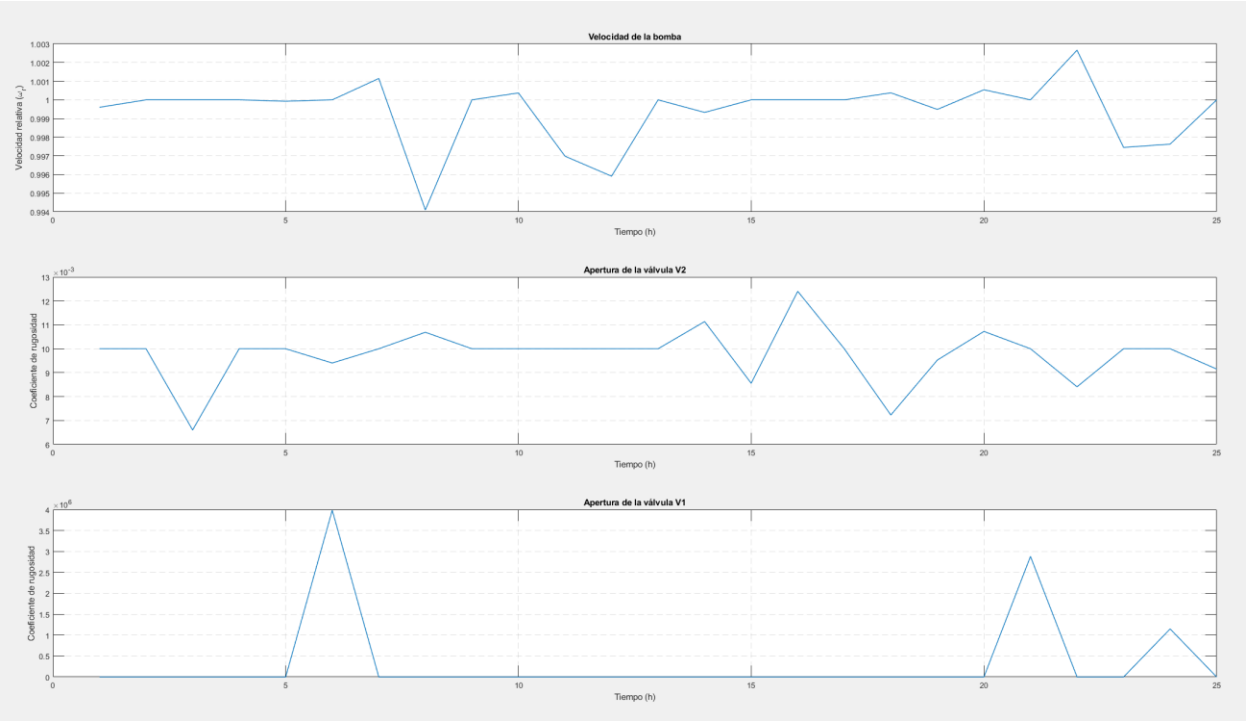

Figura 19 - Evolución de las entradas manipulables con el algoritmo genético.

En el caso del algoritmo genético, los resultados son algo mejores. En este caso sí que se puede comprobar que hay una variación en las entradas manipulables y que el nivel del depósito es el mismo al principio y al final del período. En este caso el coste final que proporciona la función 'CalculoCostes' ha sido $J = 249.155$.

En general, los resultados que se han obtenido mediante los algoritmos de optimización de Matlab con el simulador de EPANET no han sido buenos, si bien es posible que, condicionando el problema de una forma distinta y con soluciones iniciales distintas (quizá no basadas en puntos de régimen permanente) para el algoritmo genético, podría ser llegar a ser posible obtener mejores resultados.

Es conveniente mencionar se han probado un gran número de posibles combinaciones de términos de la función de coste (como, por ejemplo, dando peso a las presiones en los nodos con demandas, o cambiando la forma en la que se incorpora el coste de la bomba, utilizando su caudal y la presión que proporciona), así como distintas formas de implementar la restricción de periodicidad en el nivel del depósito. No obstante, estos cambios no han proporcionado resultados mucho mejores, y en algunos casos sólo empeoraban la capacidad de converger de los optimizadores.

Es por ello que sería razonable plantear una alternativa al uso del simulador de EPANET en Matlab para proporcionar una solución al problema de optimización. El uso de la toolbox de EPANET en Matlab proporciona muchos beneficios a la hora de simular la red, pero la optimización de la misma mediante los optimizadores disponibles en Matlab presenta grandes dificultades a la hora de obtener buenos resultados.

Para ello se plantea el uso de una herramienta distinta tanto para la implementación de la red como la del problema de optimización. La siguiente sección se dedicará a la descripción de CasADi como herramienta de implementación de problemas basados en ecuaciones algebraico-diferenciales, como es el caso de las redes de agua, y de su simulación y optimización con las distintas herramientas que incorpora.

5 OPTIMIZACIÓN MEDIANTE CASADI

Debido a la gran dificultad que se ha comprobado que los algoritmos de optimización de Matlab tienen a la hora de trabajar con el simulador de EPANET, y con las redes de agua en general, se ha planteado desde un principio el utilizar otros métodos para la optimización del problema. En particular, se ha planteado utilizar CasADi para el modelado de la red de aguas y la resolución del problema de optimización del RTO.

CasADi [8] es una herramienta de optimización y diferenciación algorítmica de código abierto cuyo objetivo es facilitar la implementación de diferentes métodos de optimización y control óptimo, incluyendo herramientas para resolución de problema de programación no lineal, problemas de programación cuadrática, e integradores de ecuaciones algebraico-diferenciales. Para este trabajo, se ha trabajado con la interfaz de CasADi en Matlab, pero ésta también está disponible en Python y Octave.

Todo esto hace a CasADi una herramienta que puede ser muy interesante para su aplicación en redes de agua, ya que el programa tendrá una visión más transparente de cómo se va a comportar la red, todo lo contrario al caso anterior en el que la interfaz de EPANET era completamente opaca para los optimizadores de Matlab.

Dicho esto, aunque CasADi propone una forma bastante rápida y sencilla de implementar una serie de integradores y distintos optimizadores para un determinado problema, la herramienta hace uso de una sintaxis bastante específica que requerirá, no solo de un proceso de aprendizaje de la misma previo, sino también una necesidad de volver a implementar modelos previamente existentes para adaptarlos a dicha sintaxis.

Con el objetivo de poner en situación, al menos a grandes rasgos, de cómo está estructurada la sintaxis de CasADi, se va a hacer una pequeña introducción a la misma (al menos a la parte que se ha tratado en este trabajo) antes de hablar de la implementación del modelo de la red de aguas y del problema de optimización planteado. Además, se realizará también una introducción a las funciones que CasADi implementa que van a tener mayor relevancia en este trabajo.

5.1. Introducción a CasADi

5.1.1 Tipos de expresiones

La forma en la que CasADi trabaja es similar a Matlab, en el sentido de que todas las variables son consideradas como matrices (los escalares serían matrices 1×1 y los vectores, matrices $n \times 1$). CasADi, las variables en CasADi serán expresiones simbólicas que podrán ser de tres tipos:

- Expresiones tipo SX, que operarán mediante operaciones escalares.
- Expresiones tipo DM, donde los elementos que no son cero serán valores numéricos en lugar de expresiones simbólicas, y que por lo demás son idénticos a las expresiones SX.
- Expresiones tipo MX, que operarán mediante operaciones de matrices.

Un ejemplo para ilustrar las diferencias entre la operación con variables SX y con variables MX puede ser el siguiente: una expresión del tipo $A + B$, donde tanto A como B son matrices $n \times m$, para las variables tipo MX esta expresión resultaría en una única operación de suma, mientras que para las variables SX, esto podría resultar en hasta $m \cdot n$ operaciones de suma.

La definición de una expresión en CasADi es bastante sencilla:

```
A=SX.eye(2);
A(1,1)=SX.sym('x');
A(2,1)=0;
```

Con esto se crearía una matriz A 2×2 cuyo valor es:

$$A = \begin{bmatrix} x & 00 \\ 0 & 1 \end{bmatrix}$$

Como se puede observar, CasADi hace una distinción entre ceros numéricos (0 en la matriz anterior) y ceros estructurales (00 en la matriz anterior).

De igual forma, también pueden definirse directamente matrices de simbólicos:

```
x=SX.sym('x',2,2)
```

Proporcionaría una matriz x cuyas componentes serán:

$$x = \begin{bmatrix} x_0 & x_2 \\ x_1 & x_3 \end{bmatrix}$$

5.1.2 Herramientas de CasADi más relevantes para la implementación del modelo de la red y su optimización

Integradores

En primer lugar, para llevar a cabo la implementación del modelo de la red y su simulación, ha sido necesario utilizar un integrador que resuelva las ecuaciones algebraico-diferenciales del modelo dinámico de la red.

Para ello, CasADi incorpora distintos tipos de integradores que permiten resolver distintos tipos de problemas. La definición de un integrador en CasADi se realizaría de la siguiente forma:

```
F=integrator('F','idas', dae, op);
```

Donde 'F' es el nombre del integrador, 'idas' sería el tipo de integrador, que en este caso sería el IDAS de la suite de SUNDIALS, y 'dae' sería la estructura que incorpora la ecuaciones algebraico-diferenciales del modelo en cuestión. La definición del DAE se explicará más adelante cuando se hable de la implementación del modelo de la red de aguas en CasADi. Por último, 'op' incluiría las opciones del integrador que se haya elegido.

En cuanto a los tipos de integradores, CasADi incorpora por defecto 5 tipos de integradores: CVODES, un integrador de la suite de SUNDIALS para sistemas ODE (ecuaciones diferenciales ordinarias), IDAS, también de la suite de SUNDIALS y pensada para sistemas DAE (ecuaciones algebraico-diferenciales), collocation y oldcollocation, dos formas de implementar integradores Runge-Kutta implícitos con paso fijo basados en métodos de colocación para ODEs y DAEs, y 'rk', un integrador Runge-Kutta explícito con paso fijo para ODEs que implementa RK4.

La elección del tipo de integrador dependerá del tipo de sistema con el que se vaya a trabajar. En este caso, se ha probado a trabajar tanto con el integrador IDAS de SUNDIALS como con el de métodos de colocación, siendo este último el que mejores resultados ha conseguido en términos de convergencia del problema de optimización, mientras que el IDAS, correctamente configurado, podía llegar a converger al óptimo de manera mucho más rápida, pero con mayor dificultad.

Una vez definido el integrador, se puede llamar a éste pasando el estado inicial, los parámetros y, si fuese necesario, lo solución inicial para el DAE:

```
Fk=F('x0', x0, 'p', p, 'z0', z0);
```

De esta forma, el integrador iteraría durante el tiempo que se haya configurado en las opciones que se pasan al

integrador cuando se define y devolvería, a través de la variable Fk, la siguiente información al final del intervalo de integración:

- 'Fk.xf' incluiría los estados diferenciales al final de intervalo de integración.
- 'Fk.zf' incluiría los valores de las variables algebraicas.
- 'Fk.qf' se correspondería con el valor de cuadratura en el instante final.

Solvers de problemas de optimización

CasADi incorpora además una serie de solvers para distintos tipos de problemas, como solvers para problemas de programación no lineal o de programación cuadrática. En este caso, ya que el problema de optimización que se ha tratado para la red de aguas es un problema NLP, se van a describir los distintos tipos de solvers NLP que CasADi incorpora y su implementación.

La definición del problema NLP se realiza a través de la función 'nlpsol' de CasADi:

```
S=nlpsol('S', 'ipopt', nlp, options);
```

Donde 'S' sería el nombre asignado al problema NLP, 'ipopt' se correspondería con el solver utilizado, en este caso, el algoritmo de punto interior IPOPT (Interior Point Optimizer) del proyecto COIN-OR y 'nlp' sería una estructura donde se definen el coste y las restricciones no lineales del problema de la siguiente forma:

```
nlp=struct('f', J, 'g', vertcat(G{:}), vertcat(w{:}));
```

Donde J sería la variable asociada al coste del problema, G sería el vector de restricciones no lineales y w el vector de variables de decisión.

Una vez definido el problema no lineal, se puede pasar a resolver el mismo llamando a 'S':

```
r=S('lbx', lbw, 'ubx', ubw, 'x0', x0, 'lbg' lbg, 'ubg', ubg);
```

En esta expresión, 'lbx' y 'ubx' serían los límites inferiores y superiores, respectivamente, de las variables de decisión, 'lbg' y 'ubg' serían los límites inferiores y superiores de las restricciones no lineales, G, que por lo general van a ser $-\infty$ para 'lbg' y 0 para 'ubg' en el caso de restricciones de desigualdad, y 0 y 0 para restricciones de igualdad. Por último, 'x0' sería la solución inicial que se pasa al solver. Con esto, el solver intentaría encontrar una solución, que colocaría en la variable $r.x$.

Además del algoritmo IPOPT, CasADi incluye por defecto otros solvers de problemas de programación no lineal, como son, por ejemplo, los algoritmos 'blocksqp', 'scpgen' o 'sqpmethod', que incorporan distintos algoritmos de métodos sqp, u otros como 'bonmin', 'knitro' o 'snopt'.

Una vez descritas las herramientas de CasADi con las que se ha trabajado para la optimización de la red de aguas, se pasa ahora a la descripción de la implementación del modelo de la misma, así como el problema de optimización planteado.

5.2. Implementación del modelo de la red y el optimizador en CasADi

Para la implementación del modelo de la red (llevada a cabo en el script 'Casadi.m'), el primer paso es definir el modelo DAE que va a utilizar el integrador. Para ello, se definen como expresiones SX de CasADi las entradas manipulables, demandas, variables algebraicas y parámetros (en este caso, el coste económico asociado a la electricidad que consume la bomba):

```
u=SX.sym('u',3);
d=SX.sym('d',8);
q=SX.sym('q',11);
hn=SX.sym('hn',8);
hx=SX.sym('hx',1);
pr=SX.sym('pr',1);
```

A continuación, se define el vector de pérdidas en las tuberías G, descrito en la primera sección de este trabajo, y se definen las ecuaciones diferenciales y algebraicas, en las variables f y g, respectivamente:

```
G1=[0;0;0;0;0;0;0;0;-55*u(1)^2;0;0];
G2=[0;0;0;0;0;0;0;0;0;0;0].*q;
G3=[0;0;0;0;0;0;0;0;5.8e-4;0;0].*q.^2;
G4=[1.418e-6;0.001873;0.001873;0.0285;0.00615;0.006153;0.04277;
7.088e-4;0;u(2);u(3)].*q.*abs(q);

G=G1+G2+G3+G4;

f=B*q;
g=[A*q-d;A'*hn+G+Ar'*hr+At'*hx];
```

Por último, se definen los términos de la función de cuadratura, que serán los términos de la función de coste en cada intervalo de tiempo del integrador. Estos términos serán funciones de las variables simbólicas del DAE y su suma se asignará a la variable h asociada al mismo.

Los términos elegidos han sido equivalentes a los que se tenían para los optimizadores con el simulador de EPANET, con la excepción del término asociado al coste económico de la operación de la bomba, que en este caso ha utilizado el valor del coste eléctrico de la bomba al cuadrado, y en lugar de utilizar el valor de la velocidad relativa de la bomba se utilizan el caudal y la diferencia de presión de ésta proporciona, ambos también elevados al cuadrado.

Así, los términos de la función de costes serán entonces:

```
h=pr^2*1/(1600000)*q(9)^2*(hn(2)-hn(1))^2/0.8
h=h+1/10*(Mdl.Hxnsmin(1)-min(Mdl.Hxnsmin(1),hx(1)))^2;
h=h+1/10*(max(Mdl.Hxnsmax(1),hx(1))-Mdl.Hxnsmax(1))^2;
```

Y ahora puede definirse el DAE como:

```
dae=struct('x',hx,'p',[u;d;pr],'z',[q;hn],'ode',f,'alg',g,'quad',h);
```

Ahora se define el integrador F para que utilice métodos de colocación e integre con un paso de 1 hora:

```
op=struct('t0',0,'tf',1,'collocation_scheme','radau');
F=integrator('F','collocation',dae,op);
```

Con esto se puede pasar ahora a empezar a definir el problema de programación no lineal. Este problema se ha planteado de manera muy similar a como se hizo para el apartado anterior, con la diferencia de que ahora el nivel del depósito en cada instante será también una variable de decisión, la cual se forzará mediante las restricciones no lineales a que sea el estado diferencial que proporciona como salida el integrador en cada paso de integración. Aunque en principio podría parecer contradictorio, ya que se están añadiendo más variables de decisión al problema, se ha comprobado que, planteándolo de esta forma, el optimizador trabaja de una forma mucho más eficiente y se obtienen resultados notablemente mejores. Este planteamiento sigue los siguientes pasos:

1) Se parten de unos valores iniciales x_0, u_0, del nivel del depósito y las variables de actuación, respectivamente.
2) Se integra con los valores actuales de x y u y se obtiene un valor del instante final del nivel del depósito, x_{k+1}.
3) Se fuerza mediante una restricción de igualdad que $w_x(k+1) = x_{k+1}$ obtenida del integrador.
4) Se deciden unos nuevos valores para u_{k+1} y se vuelve al paso 2 hasta que $k = N$, en cuyo paso se sigue en el siguiente paso.
5) En el instante N, se fuerza la restricción de periodicidad haciendo que $x_N = x_0$.

Con esto queda planteado el problema. Su implementación en CasADi sería la siguiente:

En primer lugar se define el valor inicial del nivel del depósito y se crean los vectores de límites inferiores y superiores de las variables de decisión, w, y las restricciones no lineales, g:

```
Xk=MX.sym('X0',1);
w{end+1}=Xk;
lbw=[lbw;40];
ubw=[ubw;60];
ubg=[];
lbg=[];
```

A continuación, se inicia un bucle desde 1 hasta N en el que se lleva a cabo lo siguiente:

1) Se definen las variables simbólicas de las entradas manipulables en el instante k y se asignan al vector de variables de decisión, con sus respectios límites inferiores y superiores. Además, se guarda el valores de la acción de control en el instante anterior, si $k = 1$:

```
U=['U' num2str(k-1)];
Uk=MX.sym(U,3);
w{end+1}=Uk;
if k==1
    Ukant=Uk;
end
lbw=[lbw;u1min;umin;umin];
ubw=[ubw;u1max;umax;umax];
```

2) Si $k > 1$, se define la restricción lineal sobre la variación máxima en la velocidad relativa de la bomba en el instante k, y se actualiza el valor de la variable que almacena el valor de la velocidad en el instante anterior:

```
if k>1
    Gr{end+1}=vertcat(Uk(1)-Ukant(1));
    ubg=[ubg;0.15];
    lbg=[lbg;-0.15];
    Ukant=Uk;
end
```

3) Se integra durante una hora en el instante k, se actualiza el valor de la variable J que va acumulando los costes de etapa y se definen las restricciones no lineales que se imponen a las variables de salida del integrador (nivel del depósito y variables algebraicas. Debido a la extensión de algunas de estas expresiones, a continuación se muestran abreviadas:

```
Fk=F('x0',Xk,'p',[Uk;Dt(:,k);precio(k)],'z0', z0)
J=J+Fk.qf;
Gr{end+1}=vertcat(…)
ubg=[ubg;0;0;…]
lbg=[lbg;-inf;-inf;…]
```

4) Por último, se crea la variable simbólica del estado en el instante $k + 1$, la cual se asociará al vector de variables de decisión y se forzará que sea la salida del integrador. Una vez hecho esto, se vuelve al principio del bucle:

```
X=['X' num2str(k)];
Xk=MX.sym(X,1);
w{end+1}=Xk;
lbw=[lbw;40];
ubw=[ubw;60];
Gr{end+1}=Fk.xf-Xk;
ubg=[ubg;0];
lbg=[lbg;0];
```

Una vez fuera del bucle, se fuerza la restricción de periodicidad, $x_{N+1} = x_0$, y ya pueden definirse las opciones del solver, general la estructura del problema y resolverlo:

```
Gr{end+1}=Xk-w{1};
ubg=[ubg;0];
lbg=[lbg;0];
```

```
% Se forma el NLP y se asigna el solver
 options.ipopt.max_iter=10000;
 options.ipopt.fixed_variable_treatment='make_constraint';
 options.ipopt.tol=1e-5;
 options.ipopt.acceptable_tol=1e-3;
 options.ipopt.linear_solver='mumps';

% Se resuelve el problema y se obtienen las acciones de control y niveles
% del tanque óptimos a partir de una solución inicial
nlp=struct('f',J,'g',vertcat(Gr{:}),'x',vertcat(w{:}));
S=nlpsol('S','ipopt',nlp); %Se genera el solver para el problema no lineal

%Se resuelve el problema
r=S('lbx',lbw,'ubx',ubw,'x0',x0,'lbg',lbg,'ubg',ubg);
```

Con esto finalizaría la implementación del DAE, el problema de optimización y el solver con algoritmo de punto interior que se han utilizado en este trabajo. A continuación, se muestran algunos resultados que se han obtenido y se hace una comparación con los obtenidos con el simulador de EPANET en la sección anterior del trabajo.

5.3. Resultados obtenidos mediante implementación en CasADi del modelo de la red y del optimizador

Los resultados van a ser presentados de la misma forma en que ya se hizo en la sección anterior: por un lado, se mostrarán las alturas y caudales medidos en cada uno de los nodos y tuberías de la red, y a continuación se mostrarán la evolución del nivel del depósito y la de las distintas variables manipulables.

Caudales

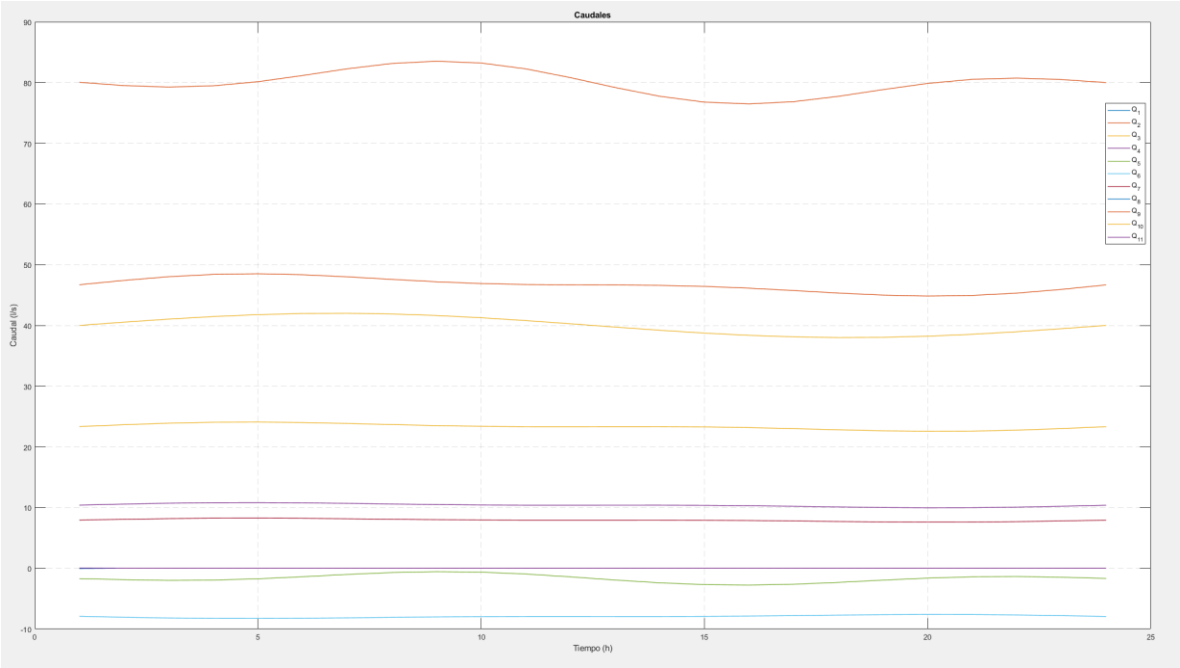

Figura 20 - Evolución de los caudales con el optimizador de CasADi.

Alturas

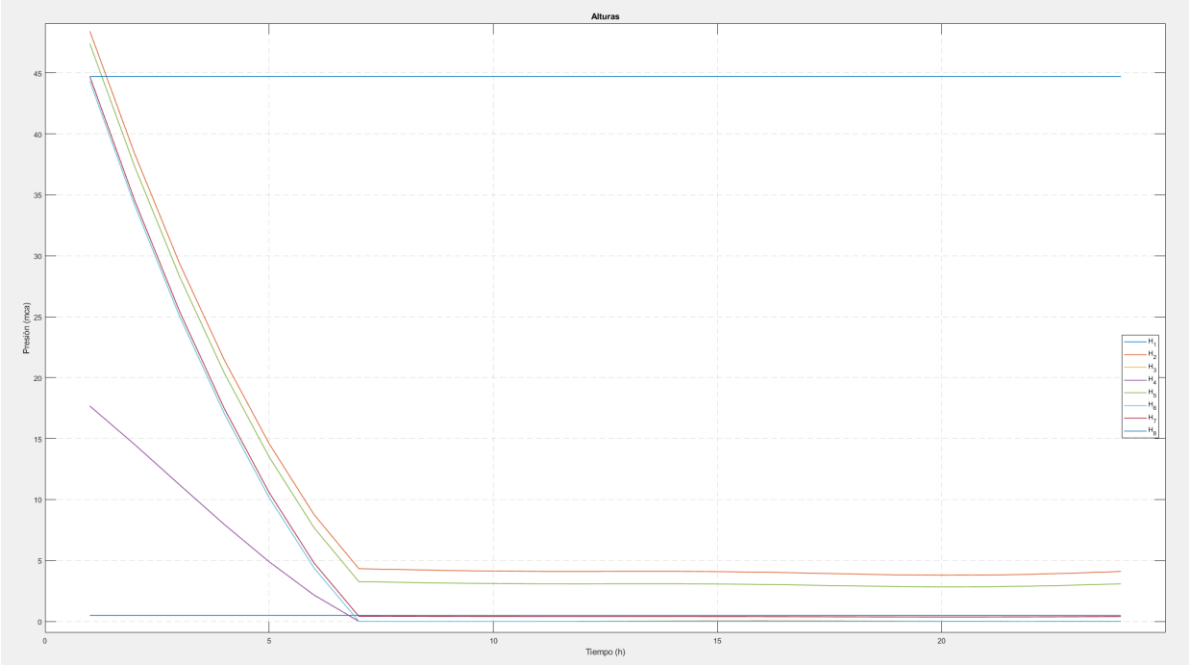

Figura 21 - Evolución de las alturas con el optimizador de CasADi.

Nivel del depósito

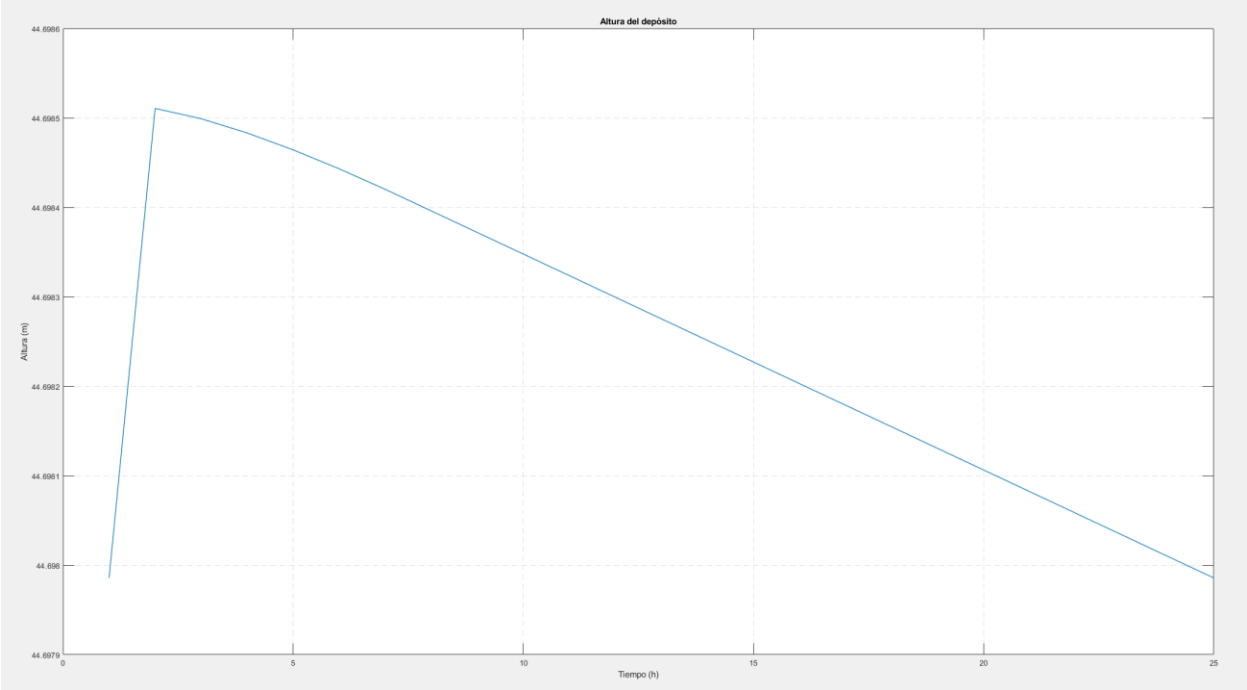

Figura 22 - Evolución del nivel del depósito con el optimizador de CasADi.

Entradas manipulables

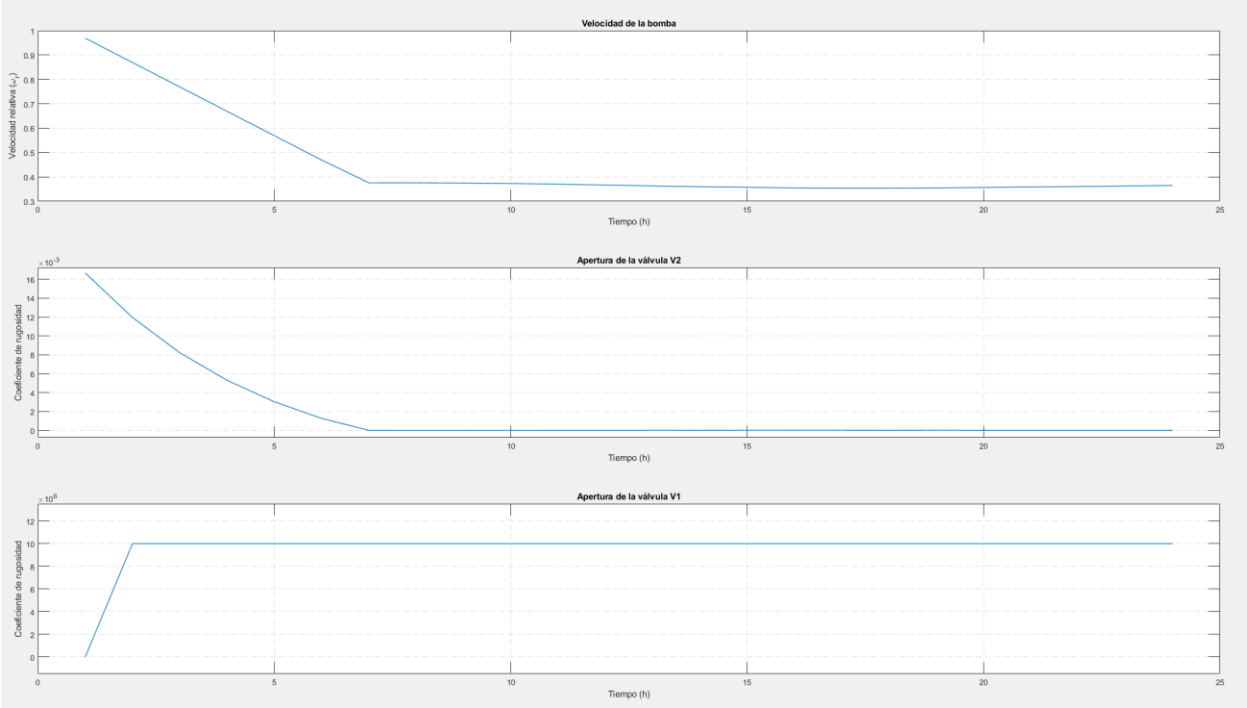

Figura 23 - Evolución de las entradas manipulables con el optimizador de CasADi.

En este caso, los resultados obtenidos son mucho mejores a los obtenidos en la sección anterior del trabajo. En primer lugar, la velocidad de la válvula es mayor al principio, coincidiendo con el instante en el que el coste económico es menor, con el objetivo de dar un margen con el que el nivel del depósito pueda trabajar sin dejar de cumplir la restricción de periodicidad, y luego pasa a tener la velocidad mínima necesaria para poder cubrir las demandas de la red.

Esto en sí ya es una notable mejora: las demandas con las que se ha trabajado no necesitaban una velocidad tan elevada de la bomba, pero los algoritmos anteriores no eran capaces de darse cuenta de esto y reducir la velocidad de la bomba, al contrario que ocurre aquí.

Por otro lado, la implementación de la restricción de periodicidad en sí ha sido bastante más sencilla y no se ha tenido que recurrir a convertirla en un término de la función de coste.

El coste que se obtiene al pasar esta solución por la función 'CalculoCostes' que se utilizó en la sección anterior es $J = 126.72$, un valor de aproximadamente la mitad del que se tenía para los resultados anteriores.

Otro de los factores a favor de la utilización del optimizador de CasADi es la gran diferencia en la rapidez con la que el optimizador converge hasta el punto final. Con los algoritmos de la sección anterior, incluso utilizando tolerancias relajadas que permitían converger con mayor rapidez a puntos quizá no tan bueno, el tiempo de ejecución seguía siendo muy elevado, de entre 5 y 10 minutos para los algoritmos de fmincon y aún más elevados para el algoritmo genético, para el cual se ha llegado incluso a poner una restricción de tiempo de 20 minutos sin que éste llegase a converger. Sin embargo, el solver IPOPT que utiliza CasADi consigue llegar a una solución en menos de un minuto de ejecución, e incluso mejor si se utiliza un integrador tipo IDAS que, si bien presenta bastantes fallos a la hora de trabajar con el optimizador si no se parte de un punto inicial y una configuración favorables, puede llegar a resolver el problema en cuestión de segundos.

En general, las primeras impresiones que se tienen favorecen mucho el uso de las herramientas de CasADi y de la diferenciación algorítmica en general para el problema de la optimización de redes de agua, en comparación a las técnicas usadas en la sección anterior con optimizadores que interactúan de manera opaca con el simulador de EPANET.

A continuación, se han modificado las restricciones del problema de manera que las entradas manipulables

también tengan la restricción de periodicidad. Para conseguir esto, se ha tenido que modificar el problema ya que en caso contrario el optimizador no conseguía llegar a converger. Los cambios que se han realizado han sido:

1) En primer lugar, se ha modificado el coste asociado a la bomba, pasando ahora a ser:

```
h=pr^2*1/(160000)*q(9)^2*(hn(2)-hn(1))^2/0.8;
```

2) Se ha añadido un término de coste adicional en el que se penaliza la distancia con respecto al nivel mínimo absoluto del depósito. Este coste será 10000 cuando el nivel del depósito sea igual al mínimo, y menor que uno cuando la distancia entre ambos niveles sea, al menos, de 1 metro. La expresión del coste es la siguiente:

```
h=h+1*(1/(1e-4+(hx(1)-Mdl.Hxmin(1))));
```

3) Se ha modificado la variación máxima en la velocidad de la bomba a $|\Delta\omega_r| \leq 0.06$
4) Se han planteado las restricciones de periodicidad como restricciones blandas para las variables manipulables, de manera que el valor final ω_r podrá variar en un rango de ± 0.01 con respecto al valor inicial, y el valores de los coeficientes de rugosidad de las válvulas podrá variar en un rango de ± 0.001.

Así, los resultados que se han obtenido en este caso han sido los siguientes:

Caudales

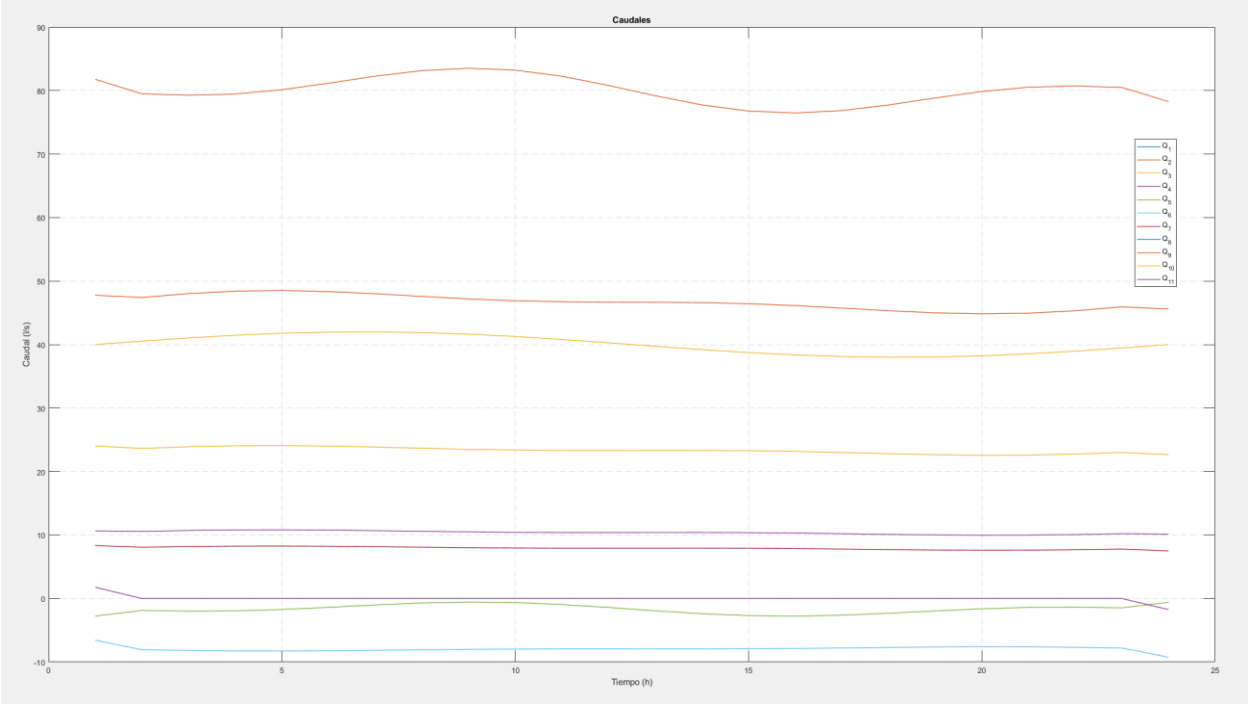

Figura 24 - Evolución de los caudales con CasADi y todas las restricciones de periodicidad.

Alturas

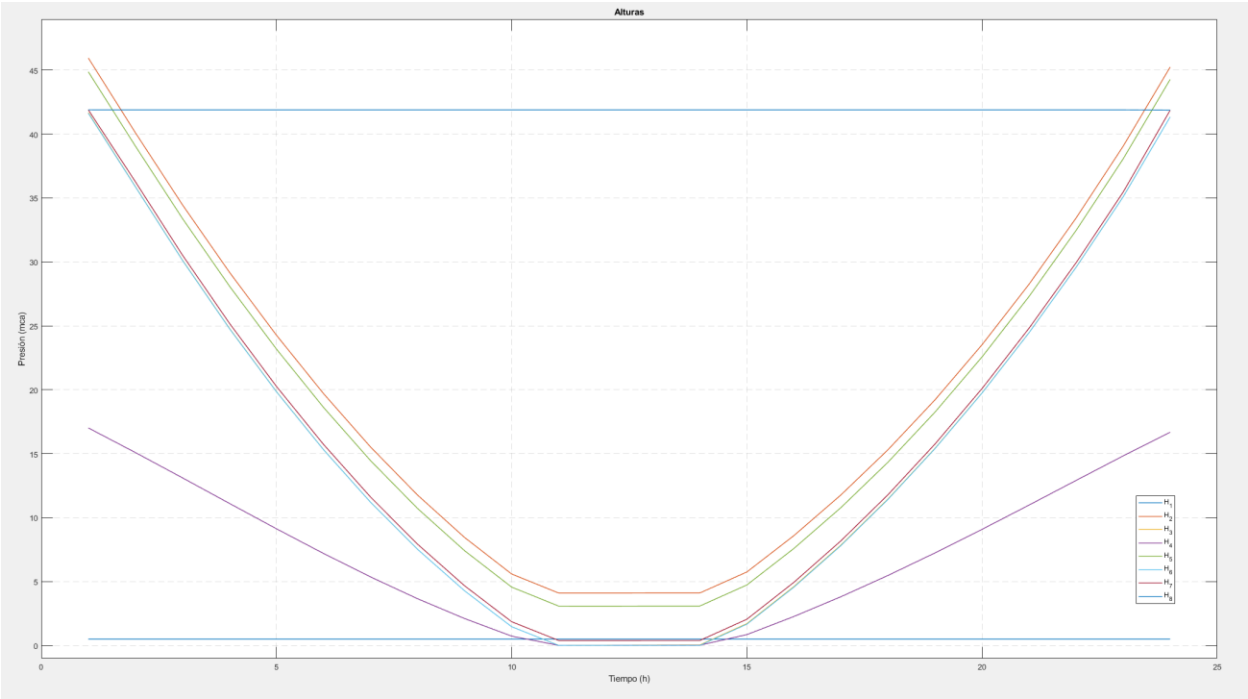

Figura 25 - Evolución de las alturas con CasADi y todas las restricciones de periodicidad.

Nivel del depósito

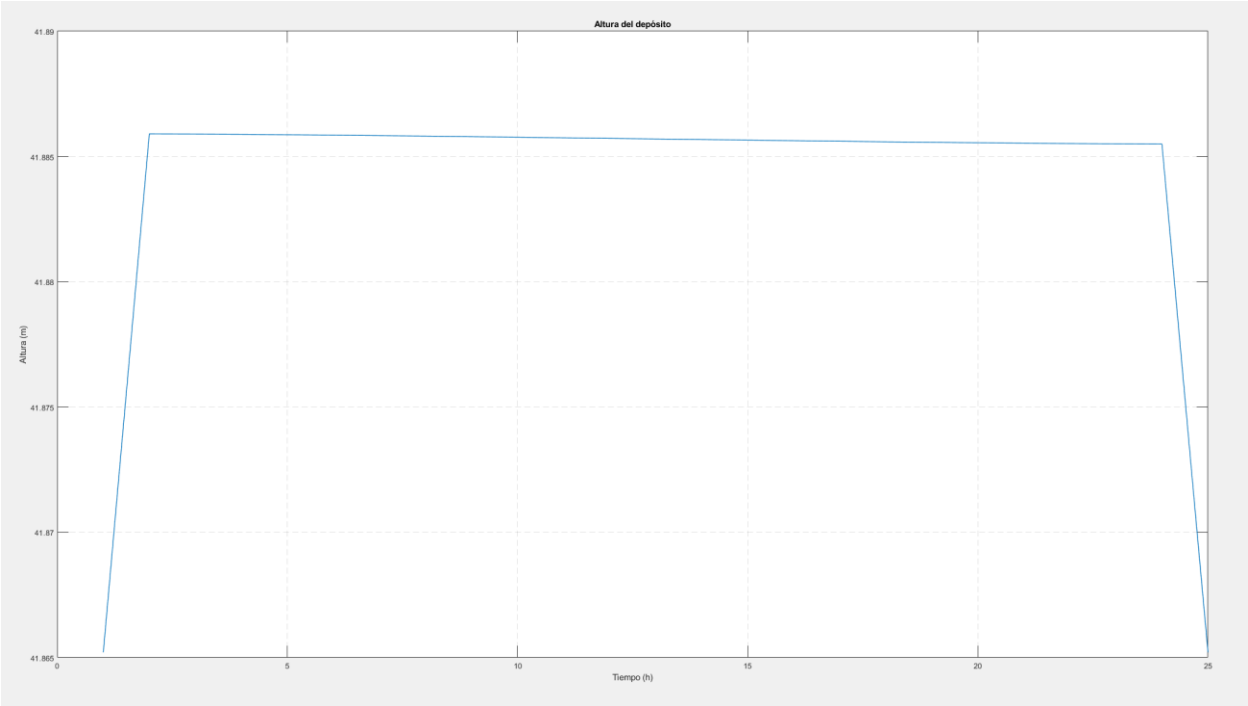

Figura 26 - Evolución del nivel del depósito con CasADi y todas las restricciones de periodicidad.

Entradas manipulables

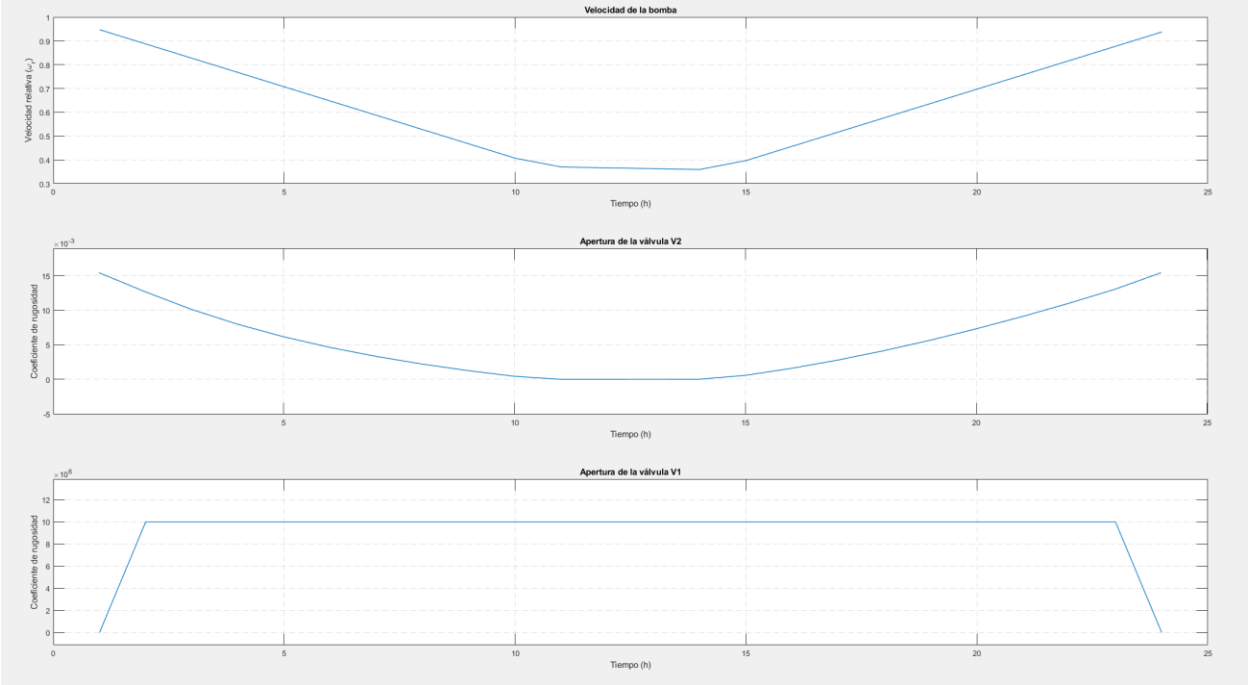

Figura 27 – Evolución de las entradas manipulables con CasADi y todas las restricciones de periodicidad.

Como puede observarse, los resultados han sido bastante buenos. Se puede comprobar cómo la velocidad de la bomba es menor para el intervalo de tiempo en el que el precio alcanza su valor máximo, y cómo vuelve a subir al final del período para volver a su valor inicial. El coste calculado ha sido $J = 196.44$, que aunque es mayor que el valor que se tenía antes, sigue siendo un muy buen resultado teniendo en cuenta que se ha restringido mucho más el problema.

Por último, se han cambiado las demandas de la red para ver cuál es el funcionamiento de CasADi en un punto de operación distinto al que se había utilizado hasta ahora. Las nuevas demandas han sido:

$$D = \begin{bmatrix} 0 \\ 10 - 3\sin\left(\frac{\pi}{6}\right) \\ 5 \\ 30 + 2\sin\left(\frac{\pi}{12}\right) \\ 5 \\ 20 + 1\sin\left(\frac{\pi}{6}\right) \\ 10 \\ 10 \end{bmatrix}$$

La solución inicial que se ha pasado a CasADi, calculada de la misma forma en que se hizo en las ocasiones anteriores, ha sido un punto de equilibrio tal que:

$$H_x = 42.9, \qquad \omega_r = 1, \qquad u_1 = 0.01, \qquad u_2 = 0.01$$

Así, los resultados obtenidos han sido los siguientes:

Alturas

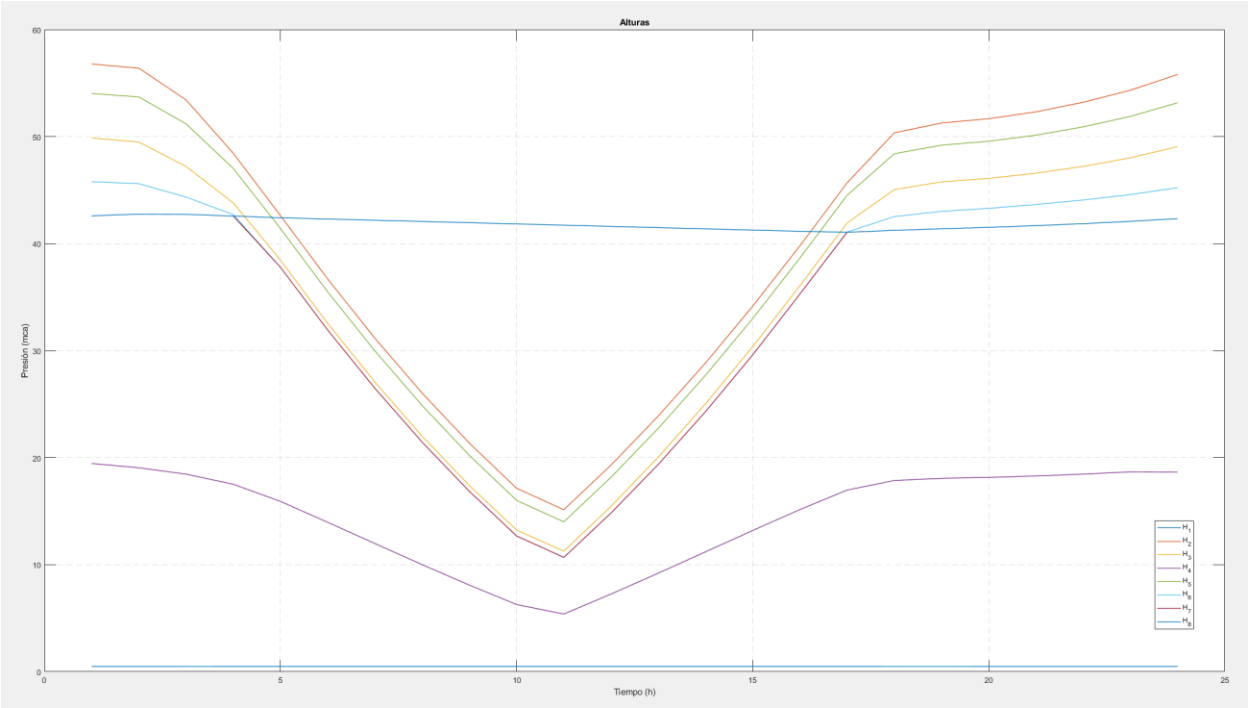

Figura 28 – Alturas de la red para el caso con demandas alternativas y CasADi.

Caudales

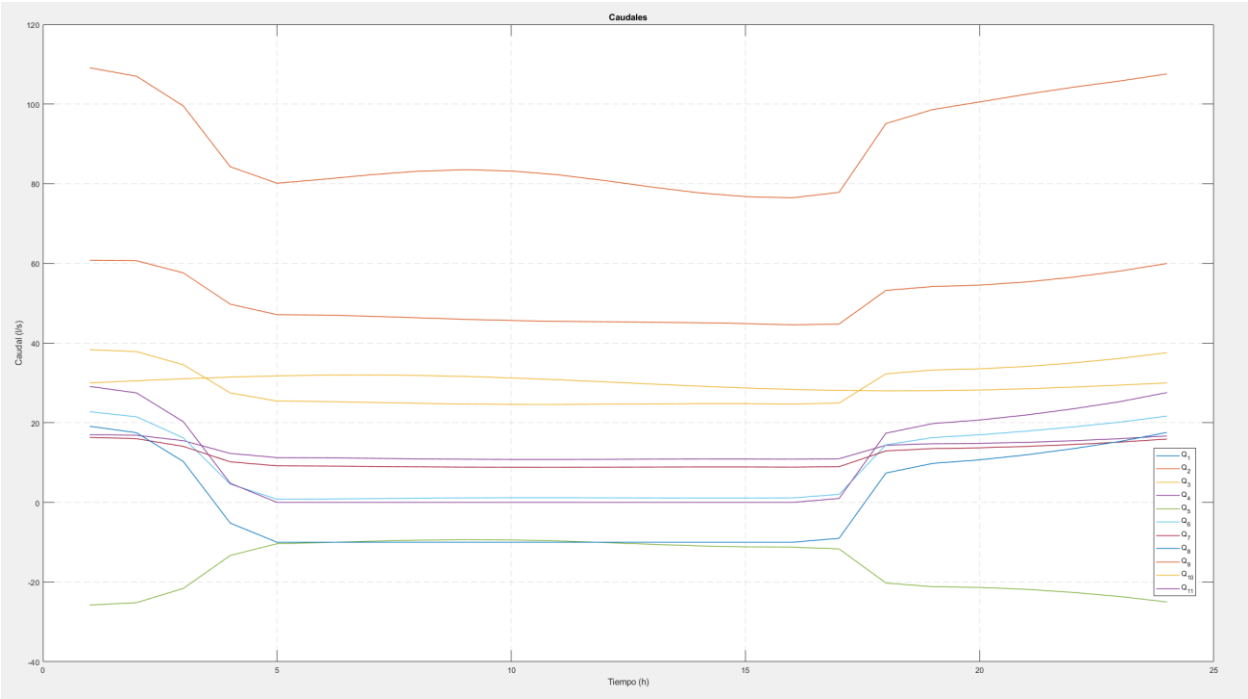

Figura 29 – Caudales de la red para el caso con demandas alternativas y CasADi.

Nivel del depósito

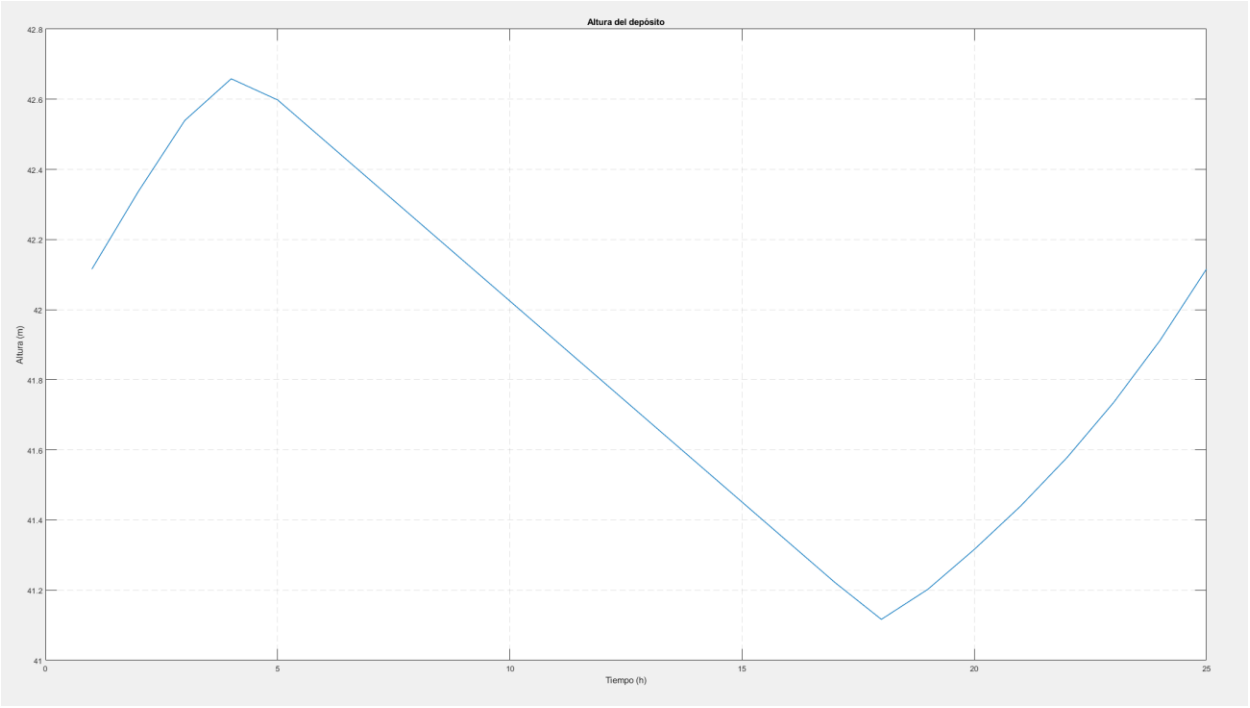

Figura 30 – Nivel del depósito para el caso con demandas alternativas y CasADi.

Entradas manipulables

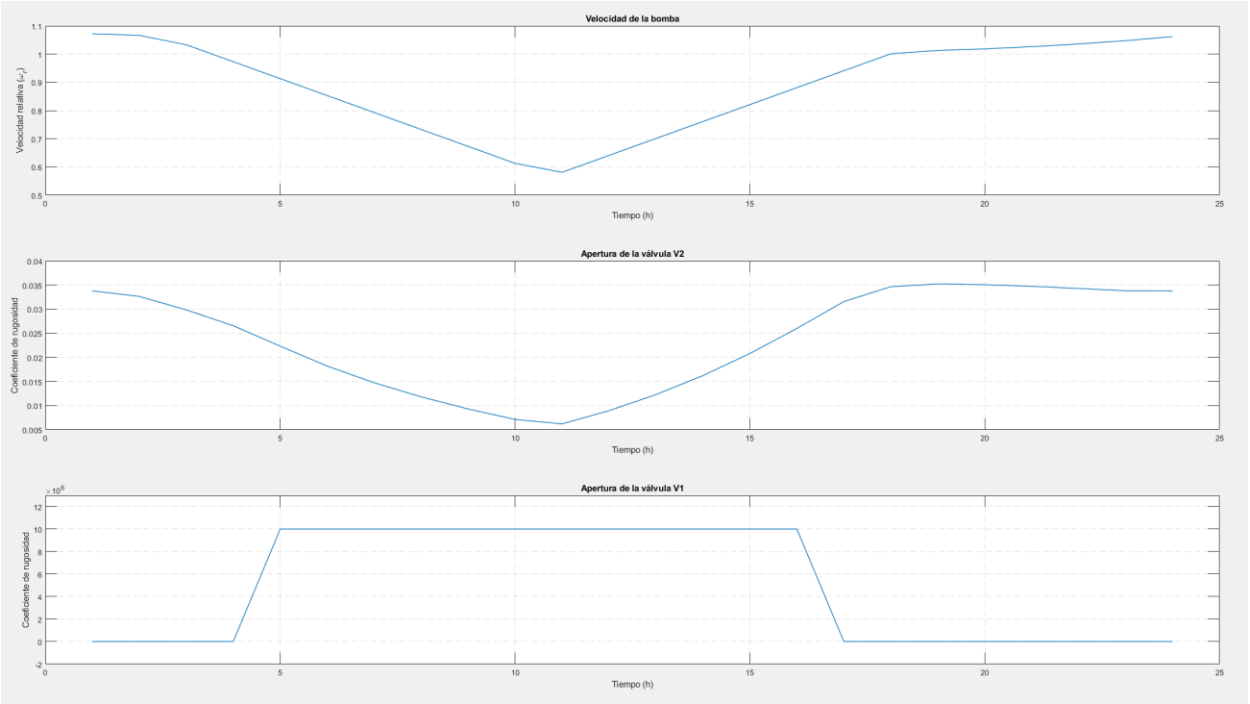

Figura 31 – Variables manipulables para el caso con demandas alternativas y CasADi.

El optimizador de CasADi ha conseguido resolver el problema sin necesidad de realizar ningún cambio excepto por la solución inicial del problema. Los resultados son bastante buenos: la presencia de una demanda en el nodo 8, colocado justo a la salida del depósito, acaba haciendo que la trayectoria óptima haga un mayor uso del

depósito, cuyo nivel hasta ahora, con las demandas anteriores, se había mantenido bastante constante. Por otro lado, se puede comprobar como las restricciones de periodicidad se cumplen sin problemas y los estados y entradas vuelven a sus valores iniciales al final del período. El coste calculado ha sido $J = 250.53$, lo cual es bastante aceptable y comparable con el coste que se tenía para los otros optimizadores en un punto en el que la suma de las demandas era 10 lps inferior.

En definitiva, queda bastante claro que el optimizador implementado con CasADi, al menos con la formulación del problema con la que se ha trabajado, proporciona mucho mejores resultados, en menor tiempo y con una facilidad para la convergencia en presencia de restricciones mucho mayor.

Con esto, se da por terminada la parte de implementación de RTO de este Trabajo de Fin de Master. A continuación, se pasa a describir la implementación del SSTO y MPC con los que se va a trabajar para controlar las planta en cada momento y llevarla hasta el punto óptimo que ha proporcionado el RTO implementado en CasADi.

6 IMPLEMENTACIÓN DEL SSTO Y MPC

Una vez implementado el RTO, el siguiente nivel en la jerarquía de control trataría de implementar la optimización dinámica de la red a través del SSTO y el MPC. Como ya se describió antes, la función del SSTO es esencialmente la de adaptar las referencias proporcionadas por el RTO, que han sido obtenidas a partir de un modelo no lineal de la red, al modelo dinámico lineal que utilizará el MPC para llevar a cabo el control de ésta.

Esta sección del trabajo estará dividida en tres partes: en primer lugar, se describirá cuál va a ser el modelo lineal de la red con el que van a trabajar tanto el MPC con el SSTO. A continuación, se hará una descripción más detallada de cómo funciona el SSTO y se tratará su implementación para dicho modelo y los resultados que se obtienen. Por último, se detallará la implementación del MPC para la red y se mostrarán algunos ejemplos de las trayectorias calculadas óptimas calculadas por el MPC para llegar a las referencias del SSTO con distintos factores de ponderación del seguimiento de referencias y peso de la acción de control.

6.1. Descripción del modelo dinámico lineal de la red

El modelo lineal que se va a utilizar para trabajar con el MPC va a ser un modelo en caudales de la red, para el cual se prescinde de las presiones de la red, dejando que el control de bajo nivel se encargue del control de éstas. El modelo estará definido mediante las siguientes ecuaciones de balance de masa en los nodos:

$$x(k+1) = Ax(k) + Bu(k) + B_d d(k)$$
$$0 = A_e x(k) + B_e u(k) + E_d d(k)$$

Donde x son las alturas de los depósitos, u los caudales de la red y d las demandas en cada uno de los nodos libres.

En esencia, la primera ecuación se corresponderá con la ecuación de balance de masas en los depósitos de una red, donde el término $Ax(k)$ se corresponde con la altura actual del depósito, $Bu(k)$ se corresponde con los caudales entrantes o salientes de los depósitos, y $B_d d(k)$ se corresponde con las demandas que pudieran estar asociadas directamente a estos depósitos.

Por otro lado, la segunda ecuación hace balance de masas en los nodos libres de la red: el término $Eu(k)$ se refiere a los caudales que entran o salen de dicho nodo, y el término $E_d d(k)$ recoge la matriz de incidencia de cada una de las demandas de la red sobre los nodos libres.

Por otro lado, las matrices del modelo en cuestión para la red que es caso de estudio son las siguientes:

- $A = 1$ será una matriz con dimensiones de $n_t \times n_t$, siendo n_t el número de depósitos de la red.
- $B = \left[0\ 0\ 0\ 0\ 0\ 0\ 0\ \frac{3.6}{314}\ 0\ 0\ 0\right]$, de dimensiones $n_t \times n_q$, siendo n_q el número de caudales de la red. En este caso, la matriz B es exactamente la misma que en el modelo no lineal de la red con el que se ha trabajado hasta ahora.
- $B_d = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]$, de dimensiones $n_t \times n_n$, siendo n_n el número de nodos libres. En este caso no hay ninguna demanda asociada directamente al depósito, por lo que todos los términos son cero.
- A_e es la matriz traspuesta de B_d.

- $B_e = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$,

que se corresponde con la matriz A del modelo no lineal y que tiene dimensiones $n_n \times n_q$.

- $E_d = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$,

que tiene dimensiones $n_n \times n_n$.

6.1.1 Planteamiento del problema del SSTO

La implementación del SSTO se plantea como un problema de programación multiparamétrica, para el cual la referencia serán los caudales y alturas del depósito en cada instante que se reciben del RTO, y cuya formulación será [1]:

$$\min_{x,u} V_N(\boldsymbol{x}, \boldsymbol{u}) = \sum_{i=0}^{N} \left|\left|x(i) - x_r\right|\right|_Q^2 + \left|\left|u(i) - u_r\right|\right|_R^2$$

$$s.a. \quad x(N) = x(0)$$

$$x(j+1) = Ax(j) + Bu(j) + d(j)$$

$$A_e x(j) + B_e u(j) + E_e d(j) = 0$$

$$H_{x_i,min} \leq H_{x_i} \leq H_{x_i,max}$$

$$u_{i,min} \leq u_i \leq u_{i,max}$$

Y teniendo en cuenta que:

$$\left|\left|x - x_r\right|\right|_Q^2 = x^T Q x - 2x_r^T Q x + x_r^T Q x_r$$

Donde las matrices Q y R ponderan el coste del seguimiento de referencias y el coste de actuación, respectivamente, entonces la función de coste se puede escribir como:

$$V_N(z) = \frac{1}{2} z^T H z + h^T(x_r, u_r) z + C$$

Donde C es irrelevante para la solución del problema de optimización y la matriz H y el vector h son:

$$H = \begin{bmatrix} 2Q & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2R & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2Q & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2R & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ddots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & Q & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & R & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & P \end{bmatrix}$$

$$h^T = -2(x_r^T Q,\ u_r^T R,\ x_r^T Q,\ u_r^T R, \dots,\ x_r^T Q, u_r^T R, x_r^T P)$$

Por otro lado, las restricciones de igualdad del problema se pueden escribir como $F\boldsymbol{z} = f(x, d)$, donde la matriz F y el vector f serán:

$$F = \begin{bmatrix} A & B & -I & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & A & B & -I & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & A & B & -I \\ A_e & B_e & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & A_e & B_e & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & A_e & B_e & 0 \end{bmatrix}, \qquad f = \begin{bmatrix} -d(0) \\ -d(1) \\ \vdots \\ -d(N-1) \\ -E_e d(0) \\ -E_e d(1) \\ \vdots \\ -E_e d(N-1) \end{bmatrix}$$

Además, se añade un término adicional a F y f de restricción terminal para hacer que el nivel del depósito al final del intervalo sea igual al inicial. Esta restricción se impone mediante la matriz F_t y el escalar (que sería un vector si hubiese más de una restricción) f_t:

$$F_t = (-I, 0, 0, \dots, 0, I)$$
$$f_t = 0$$
$$F_t \boldsymbol{z} = f_t$$

Y en este caso, en el que no hay restricciones de desigualdad, el problema puede ser formulado ahora como un problema de programación cuadrática, con la siguiente estructura:

$$\min_z \frac{1}{2} \boldsymbol{z}^T H_{SSTO} \boldsymbol{z} + h^T(x_r, u_r)\boldsymbol{z}$$
$$s.a. \quad F_{SSTO}\boldsymbol{z} = f_{SSTO}(x, \boldsymbol{d})$$

Donde el vector $\boldsymbol{z}$ se corresponde con el vector de variables de decisión (en este caso, la altura del depósito y los valores de las entradas manipulables en cada momento), $\boldsymbol{d}$ sería el vector de demandas en cada instante, H y h se corresponden con la matriz y el vector de costes del problema y F y f serían la matriz y el vector de restricciones de igualdad.

Este problema de programación cuadrático podrá ser resuelto fácilmente con la función quadprog de Matlab, obteniendo la nueva trayectoria (x_{rSSTO}, u_{rSSTO}) a partir de la trayectoria original (x_r, u_r) del RTO.

6.1.2 Planteamiento del problema del MPC

El problema del MPC es muy similar al del SSTO, con la diferencia de que el nivel inicial será el nivel del depósito en el instante en el que se calcula la trayectoria del MPC, el nivel al final del horizonte de predicción, N_p del MPC se fijará a la referencia que da el SSTO para ese instante mediante la restricción terminal y, por otro lado, las matrices Q y R podrán ser distintas (dando lugar a matrices H y h distintas), dando más peso a alcanzar el nivel del tanque más rápido o con una actuación más suave.

De esta forma, el vector de restricciones de igualdad será:

$$F = \begin{bmatrix} I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A & B & -I & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & A & B & -I & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & A & B & -I \\ A_e & B_e & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & A_e & B_e & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & A_e & B_e & 0 \end{bmatrix}, \qquad f = \begin{bmatrix} H_x ini \\ -d(0) \\ -d(1) \\ \vdots \\ -d(N-1) \\ -E_e d(0) \\ -E_e d(1) \\ \vdots \\ -E_e d(N-1) \end{bmatrix}$$

Y, por otro lado, la restricción terminal será tal que:

$$F_t = (0, 0, 0, \dots, 0, I), \qquad f_t = x_{rSSTO}(Np)$$

E igual que antes, se tratará de un problema de programación cuadrática con la estructura:

$$\min_z \frac{1}{2} \boldsymbol{z}^T H_{MPC} \boldsymbol{z} + h_{MPC}^T(x_{rSSTO}, u_{rSSTO}) \boldsymbol{z}$$
$$s.a. \qquad F_{MPC} \boldsymbol{z} = f_{MPC}(H_{xini}, \boldsymbol{d})$$

A continuación, se va a tratar la implementación de tanto el SSTO como el MPC para la red que es caso de estudio en Matlab, y se mostrarán algunos resultados donde se comprobarán cuáles son las trayectorias generadas por cada uno.

6.2. Implementación del SSTO y MPC en Matlab

6.2.1. Implementación del modelo lineal y del SSTO

La implementación del modelo lineal en caudales de la red y del SSTO y MPC se ha realizado mediante el script "SSTO_MPC.m". Este script se una vez cada cierto tiempo (por lo general, una vez cada hora) para generar la trayectoria del horizonte de predicción N_p que la red deberá seguir durante la siguiente hora, para volverse a calcular una hora después, generando una nueva trayectoria de N_p horas de las cuales se volverá a seguir sólo la referencia del primer intervalo de tiempo.

En este script, en primer lugar, se implementan las matrices que definen el modelo en caudales de la red, las cuales se vieron antes:

```
A=1;
B=Mdl.B; % B=[0 0 0 0 0 0 0 3.6/314 0 0 0]
Bd=[0 0 0 0 0 0 0 0];
Ae=zeros(8,1);
E=Mdl.A;
Ed=[-1 0 0 0 0 0 0 0;0 -1 0 0 0 0 0 0;0 0 -1 0 0 0 0 0;
0 0 0 -1 0 0 0 0;0 0 0 0 -1 0 0 0; 0 0 0 0 0 -1 0 0;
0 0 0 0 0 0 -1 0;0 0 0 0 0 0 0 -1];
```

A continuación, se definen las matrices Q y R que darán peso al seguimiento de referencias y a la acción de control del SSTO. En este caso, lo que se pretende es que la trayectoria referencia del modelo lineal sea lo más parecida posible a la que proporciona el RTO, por lo que los valores de estas matrices (en este caso, Q es un escalar porque sólo hay un depósito) son:

```
Q=1e3;
R=eye(11);
```

Estas matrices definen un coste de etapa cuadrático del sistema lineal cuyo controlador óptimo es un LQR:

$$L(x,u) = x^T Q x + u^T R u$$

Se puede llegar a demostrar que el punto óptimo de este coste se da para una acción de control tal que:

$$u = -(R + B^T P B)^{-1} B^T P A x$$

Que proporcionaría un coste óptimo:

$$J^*(x) = x^T P x = x^T (Q + A^T P A) x - x^T A^T P B (R + B^T P B)^{-1} B^T P A x$$

Y donde esta matriz P sería:

$$P = Q + A^T P A - A^T P B (R + B^T P B)^{-1} B^T P A$$

Que es la ecuación de Riccati.

El valor de esta matriz P se obtiene de manera recursiva, empezando por un valor inicial de $P = Q$ e iterando hasta que se alcanza el valor que proporciona el coste óptimo. Este proceso se realiza mediante la función 'dlqr' de Matlab, que recibe como argumentos las matrices A, B, Q y R y genera dicha matriz P:

```
[~,P,~]=dlqr(A,B,Q,R);
```

Para lo cual se obtiene:

$$P = 3.303 \cdot 10^3$$

Una vez obtenida la matriz P, se definen los límites inferiores y superiores para el nivel del depósito y los caudales. En este caso, se han elegido como niveles mínimo y máximo los que se habían definido para el depósito, y como caudales mínimo y máximo se han elegido:

$$0 < Q_i < 120, \qquad si\ i = 1$$
$$-120 < Q < 120, \qquad si\ i \neq 1$$

Ya que la suma de las demandas en el caso base es de 80 lps, y de 90 lps mayor para el caso en el caso alternativo de demandas.

A continuación, se definen las referencias $[x_r, u_r]$ que se obtienen del RTO y se desplazan éstas dependiendo del instante del intervalo en el que se encuentra la red actualmente, determinado por la variable k que se pasará como entrada al script. Este desplazamiento se realiza también sobre las demandas de la red:

```
xr=Hxopt;
ur=Qopt;
xr=[xr(:,k:end),xr(:,1:k-1)];
ur=[ur(:,k:end),ur(:,1:k-1)];
D=[D(:,k:end),D(:,1:k-1)];
```

Una vez definidas las demandas y referencias a seguir por el SSTO, se definen unas variables auxiliares que contendrán los tamaños de los vectores de estados (nx), el vector de entradas manipulables (nu), el número de variables de decisión (nz, que será de nx más nu por el intervalo de tiempo del RTO) y el número de restricciones de igualdad (que se corresponde con el número de nodos, ya que las restricciones de igualdad son las ecuaciones de balance de masas en los nodos libres).

Con esto ya se pueden generar las matrices de coste H y h:

```
H=[];
for j=1:N
    H=blkdiag(H,2*Q,2*R);
end
H=blkdiag(H,2*P);
h=[];
for j=1:N
    h=[h;-2*Q*xr(:,j);-2*R*ur(:,j)];
end
h=[h;-2*P*xr(:,N+1)];
```

Y de forma análoga, se generan las matrices F y f de restricciones, así como las de la restricción terminal F_t y f_t.

```
F=zeros(N*nx+N*neq,nz);
ii=0;
ij=0;
for j=1:N
    F(ii+(1:nx),ij+(1:(2*nx+nu)))=[A,B,-eye(nx)];
    ii=ii+nx;
```

```
    ij=ij+(nx+nu);
end
if neq>0
    ij=0;
    for j=1:N
        F(ii+(1:neq),ij+(1:(nx+nu)))=[Ae,E];
        ii=ii+neq;
        ij=ij+(nx+nu);
    end
end

f=[];
for j=1:N
    f=[f;-Bd*D(:,j)];
end
for j=1:N
    f=[f;-Ed*D(:,j)];
end
Ft=zeros(nx,nz);
Ft(:,1:nx)=eye(nx);
Ft(:,N*(nx+nu)+(1:nx))=-eye(nx);
ft=zeros(nx,1);
```

Por último, se generan las restricciones de caja para todo el intervalo de tiempo y se obtiene la solución al problema mediante la función 'quadprog' de Matlab, que recibe como argumentos la matriz y vector de costes, la matriz y vector de restricciones (incluida la restricción terminal) y las restricciones de caja y genera las variables de decisión del SSTO, las cuales se dividen a posteriori en estados y variables manipulables:

```
zSSTO=quadprog(H,h,[],[],[F;Ft],[f;ft],zmin,zmax,[],options1);
for j=1:N
    xrt_SSTO(:,j)=zSSTO((j-1)*(nx+nu)+(1:nx));
    urt_SSTO(:,j)=zSSTO((j-1)*(nx+nu)+nx+(1:nu));
end
xrt_SSTO(:,Np+1)=zSSTO(Np*(nx+nu)+(1:nx));
```

Con esto ya se tendrían las referencias que se podrían pasar al MPC. A continuación, se muestran algunos resultados donde se comparan las trayectorias que proporcionaba el RTO con CasADi con las que se generan mediante el SSTO. Con el objetivo de mostrar las diferencias en las entradas manipulables, se van a mostrar las diferencias entre los caudales Q_9 y Q_{11} que proporcionan el RTO y el SSTO:

Altura del depósito

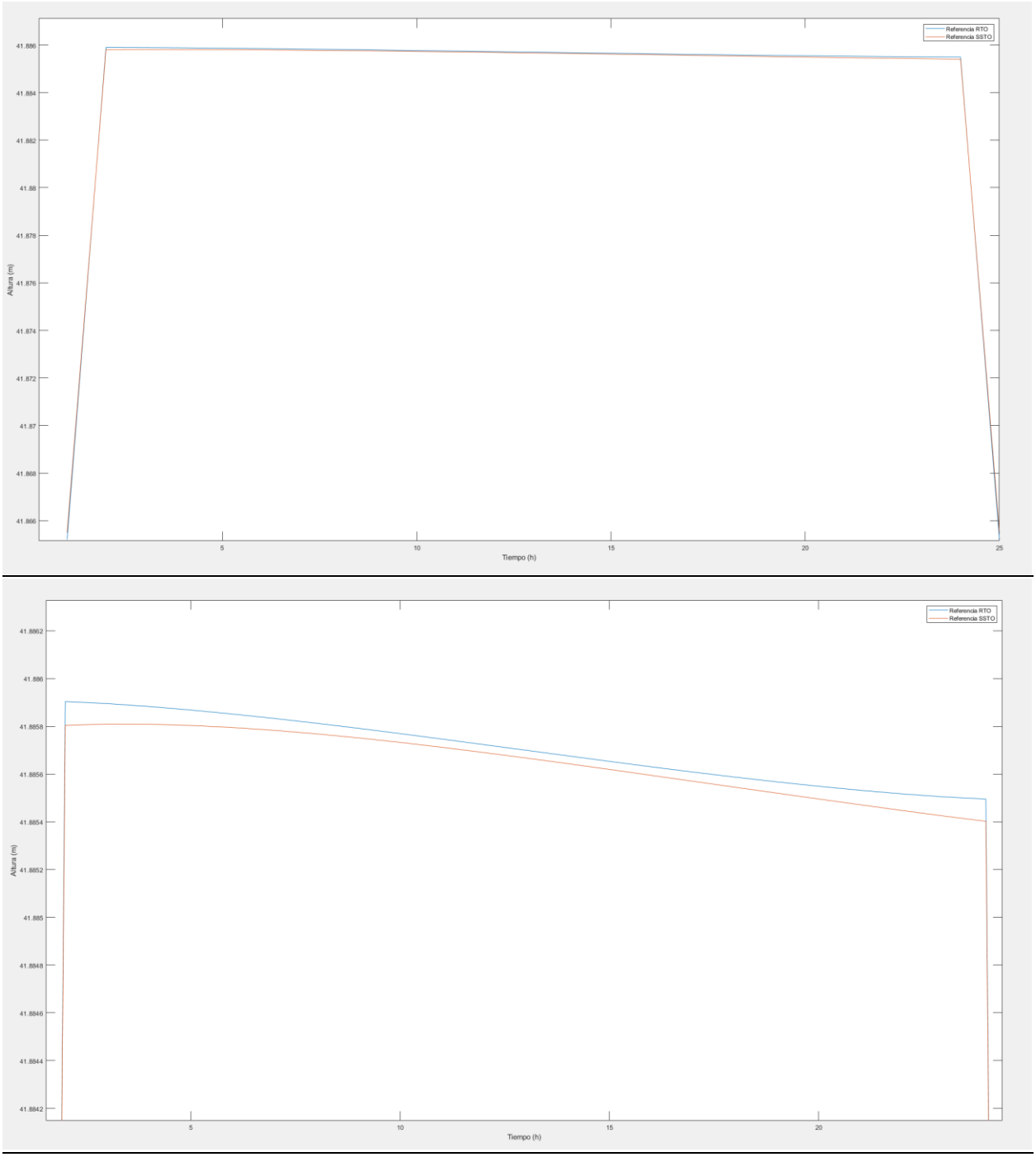

Figura 32 – Comparación de la evolución del nivel del depósito del RTO con la del SSTO.

Caudal Q_9

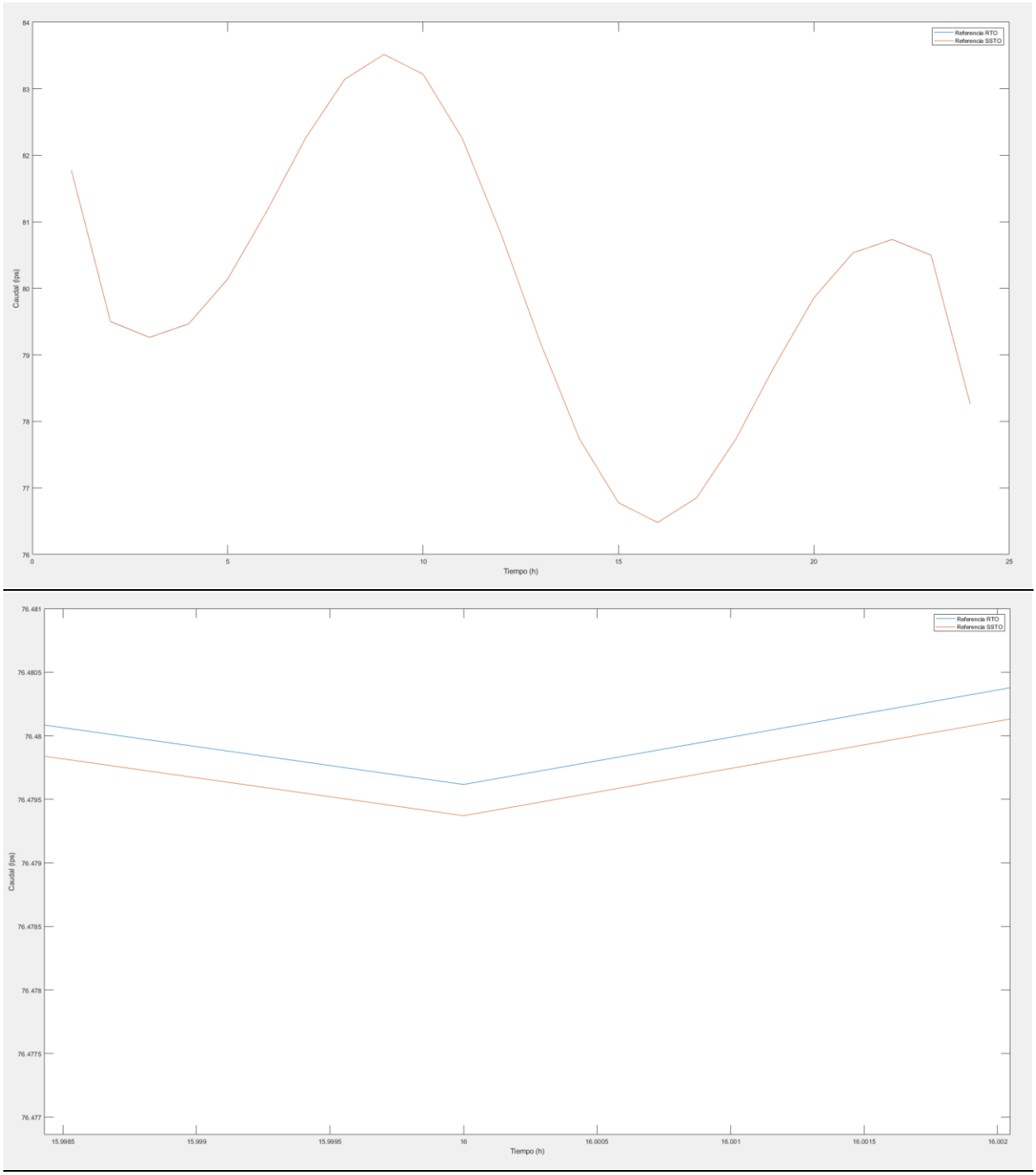

Figura 33 - Comparación de la evolución del caudal Q_9 del RTO con la del SSTO.

Caudal Q_{11}

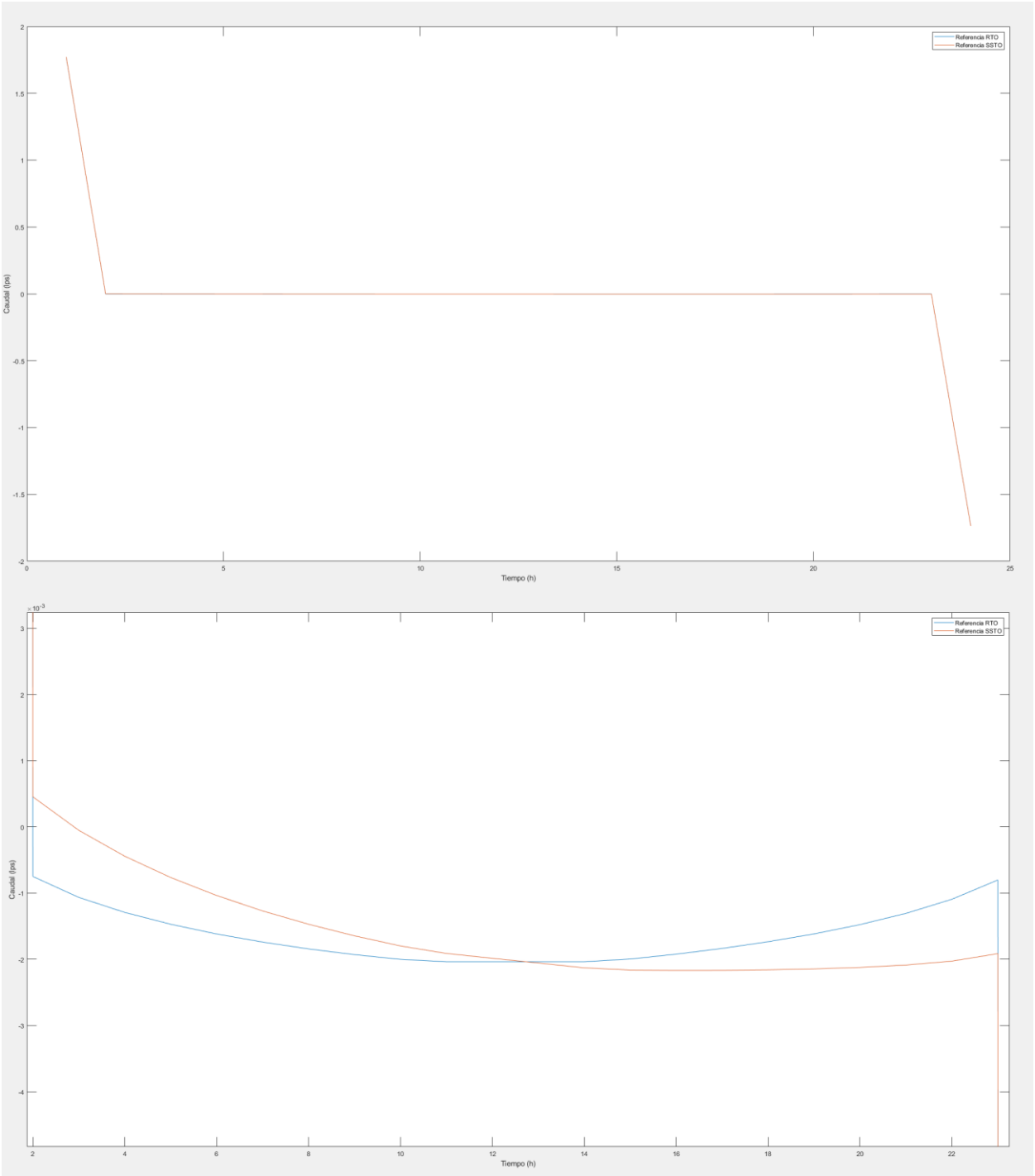

Figura 34 - Comparación de la evolución del caudal Q_{11} del RTO con la del SSTO.

Como puede observarse, aunque ambas trayectorias son muy similares, sobre todo en el caso del caudal Q_9 (que se corresponde con el que pasa por la bomba), hay algunas diferencias apreciables entre ambas trayectorias debido a las diferencias entre los modelo, las cuales se pueden apreciar, sobre todo, en el caudal Q_{11}, que se corresponde con el que pasa por la válvula V1 y el cual llega a tener errores de en torno a 1 lps. Estas pequeñas diferencias podrían traducirse en errores en régimen permanente si se intentase controlar directamente con el MPC utilizando las trayectorias que proporciona el RTO, y es precisamente lo que se ha pretendido solucionar implementando el SSTO.

Por otro lado, los resultados que se obtienen para la red funcionando bajo las demandas alternativas que se mostraron para CasADi, los resultados ahora son:

Altura del depósito

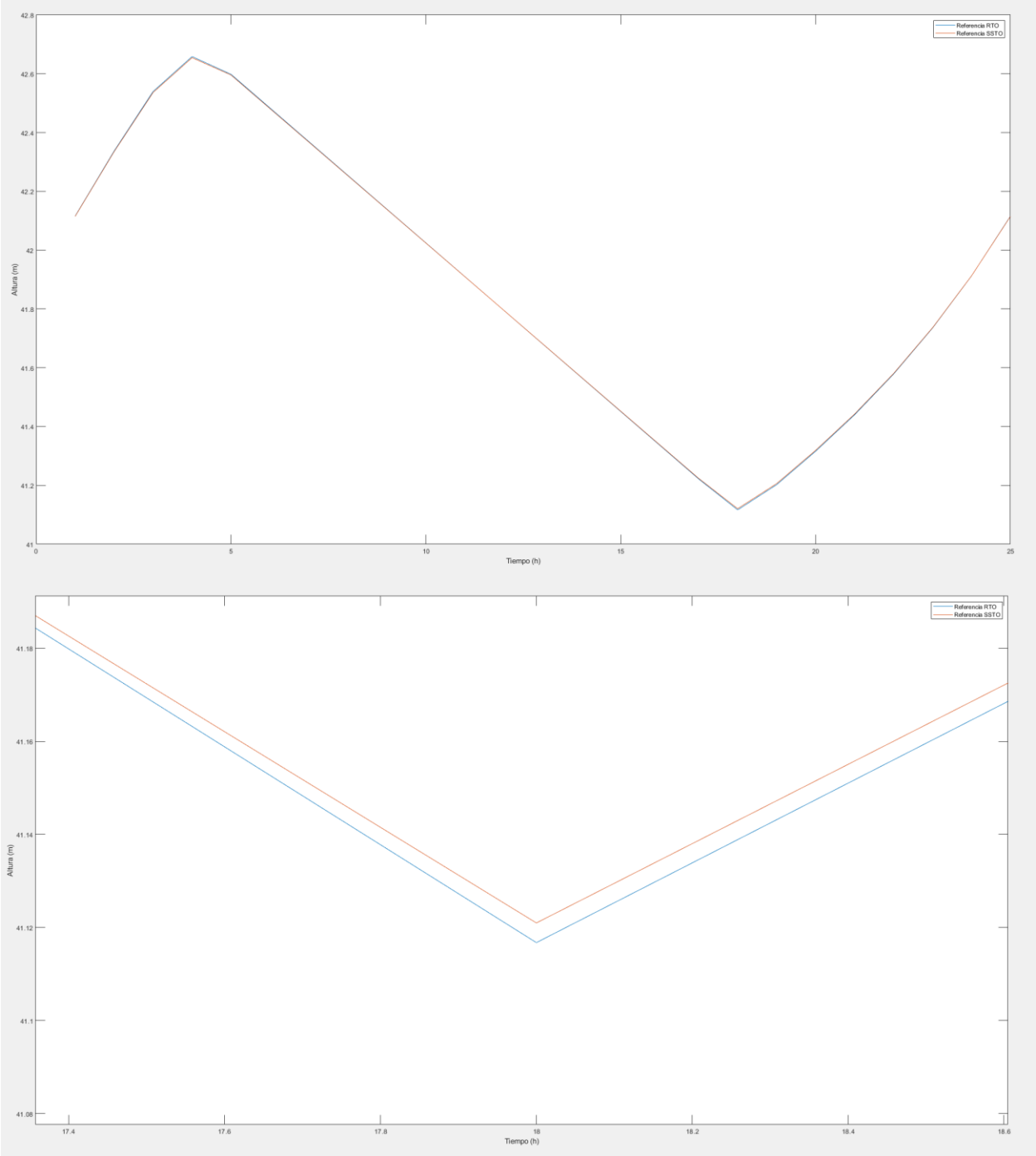

Figura 35 – Comparación de la evolución del nivel del depósito del RTO con la del SSTO para el caso de demandas alternativas.

Caudal Q_9

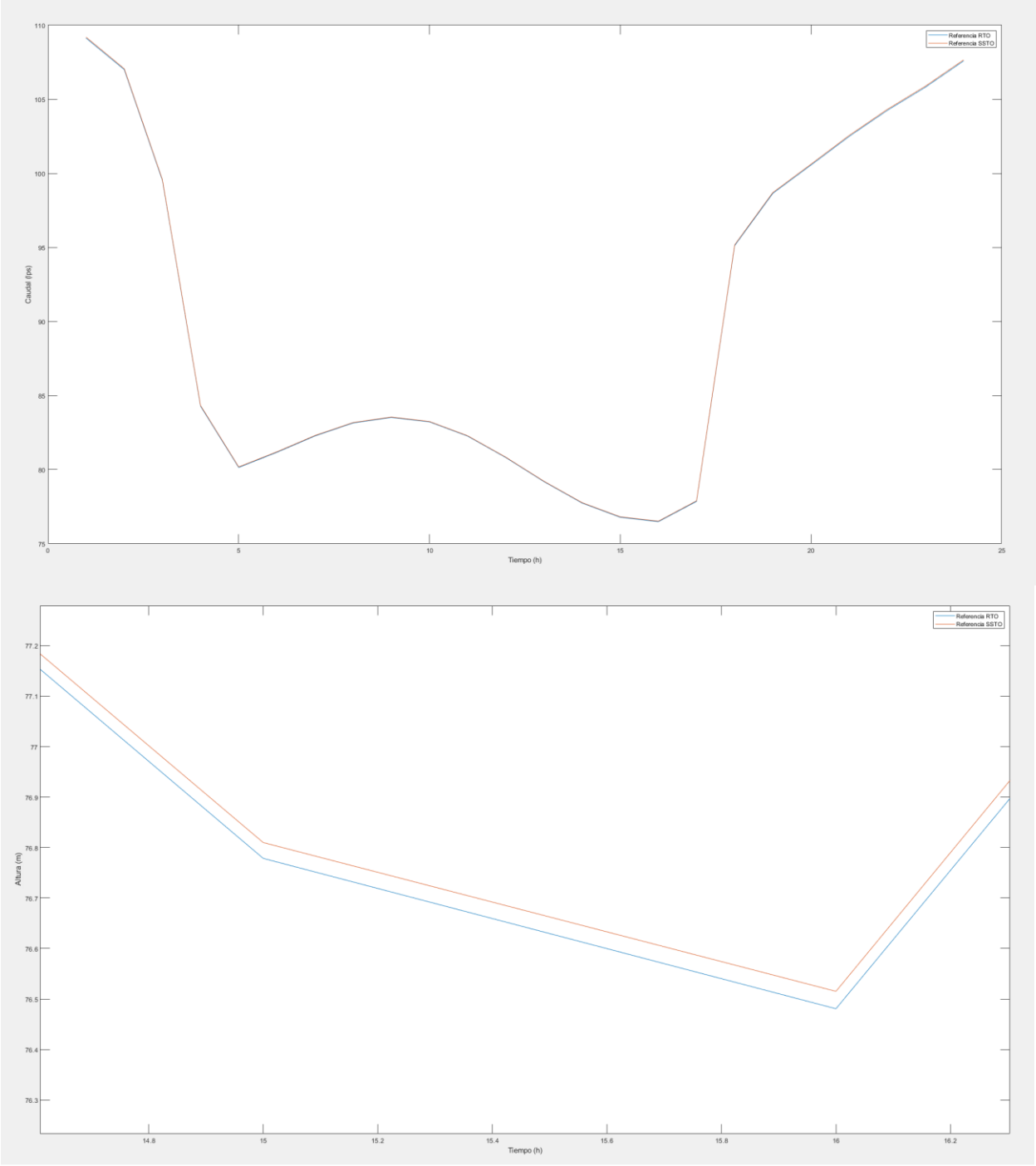

Figura 36 - Comparación de la evolución del caudal Q_9 del RTO con la del SSTO para el caso de demandas alternativas.

Caudal Q_{11}

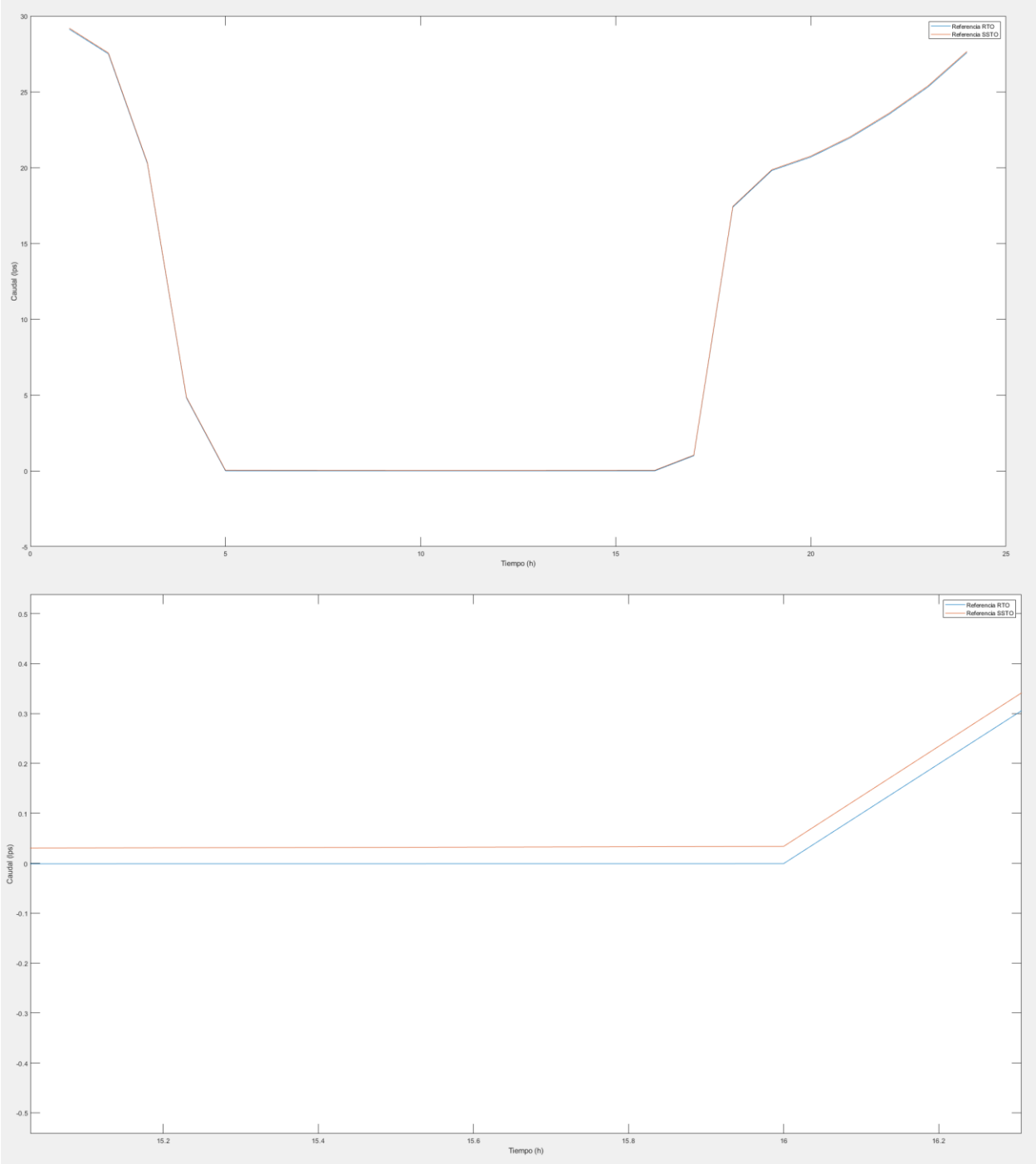

Figura 37 - Comparación de la evolución del caudal Q_{11} del RTO con la del SSTO para el caso de demandas alternativas.

Como se puede ver, los resultados siguen siendo bastante buenos en un punto de funcionamiento distinto, aunque siguen existiendo algunas diferencias entre ambos modelos.

6.2.1. Implementación del MPC

Como ya se ha mencionado antes, la implementación del MPC será muy similar a la del SSTO. De hecho, la generación de las matrices H, h, F y f para el MPC se realizará exactamente de la misma forma, pero sustituyendo ahora las referencias de estados y de entradas manipulables que se tenían del RTO por las que se acaban de calcular con el SSTO, y con unas matrices Q, R y P del RTO que, por lo general serán distintas, pero que en este caso casualmente van a ser las mismas por generar resultados bastante buenos tanto para el SSTO como para el MPC, de manera que:

$$Q_{MPC} = Q_{SSTO} = 1000, \qquad R_{MPC} = R_{SSTO} = I_{11\times 11}$$

Además, el MPC podrá trabajar con un horizonte de predicción que no necesariamente deberá de ser igual al período del RTO. En un primer lugar, este horizonte de predicción, N_p, sí que será:

$$N_p = 24$$

Sin embargo, se comprobará posteriormente cuál es el funcionamiento del MPC con horizontes de predicción menores, con el objetivo de comprobar hasta qué valor se podrá disminuir este horizonte de predicción manteniendo un funcionamiento correcto del MPC y que tendrá un menor coste computacional. Este estudio sobre el horizonte de predicción del MPC se realizará también más a fondo en la última sección del trabajo, cuando se implemente el control de bajo nivel de la red, con el objetivo de ver cómo afecta dicho parámetro a este control.

Por último, la diferencia final entre el planteamiento del problema del MPC y el del SSTO, como ya se había comentado antes, es la diferencia en el estado inicial, que se corresponderá con el nivel del depósito en el instante en el que se simula, y en la restricción terminal, para la que ahora se deberá cumplir que, en el instante final, la referencia de nivel del depósito y de los caudales sean iguales a los del SSTO, de manera que las matrices F_t y f_t serán:

$$F_t = (0, 0, 0, \dots, 0, I), \qquad f_t = x_{rSSTO}$$

Una vez generadas las matrices, igual que antes, se pasan éstas como argumentos a la función 'quadprog' de Matlab para que resuelva el problema de programación cuadrático y obtener la trayectoria óptima para del MPC.

A continuación, se muestran algunos resultados de trayectorias generadas por el MPC para distintos valores de las matrices de peso del LQR. Estos resultados se han obtenido para las referencias proporcionadas por el RTO de CasADi con el caso de demandas alternativas, y las variables mostradas serán la altura del depósito y los caudales Q_9 y Q_{11}, al igual que antes.

$\boldsymbol{Q = 10^3}$

Altura del depósito

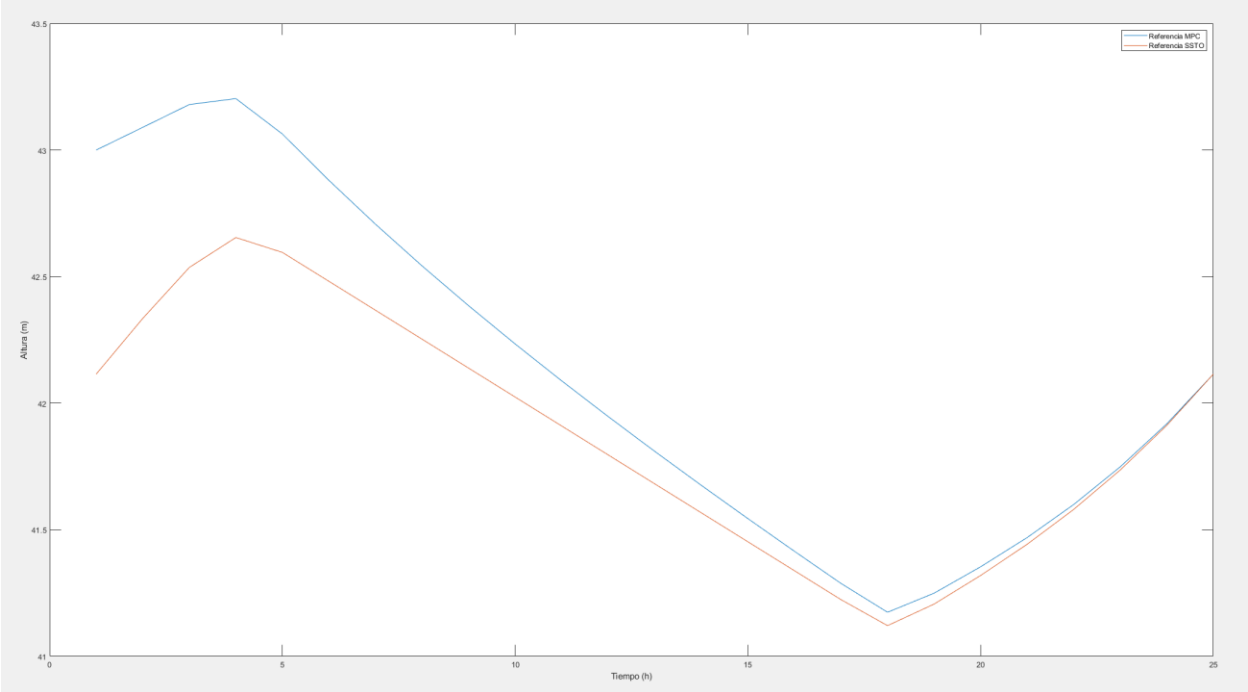

Figura 38 – Seguimiento de referencias de altura del depósito del MPC con $Q=10^3$ y $N_p=24$.

Caudal Q_9

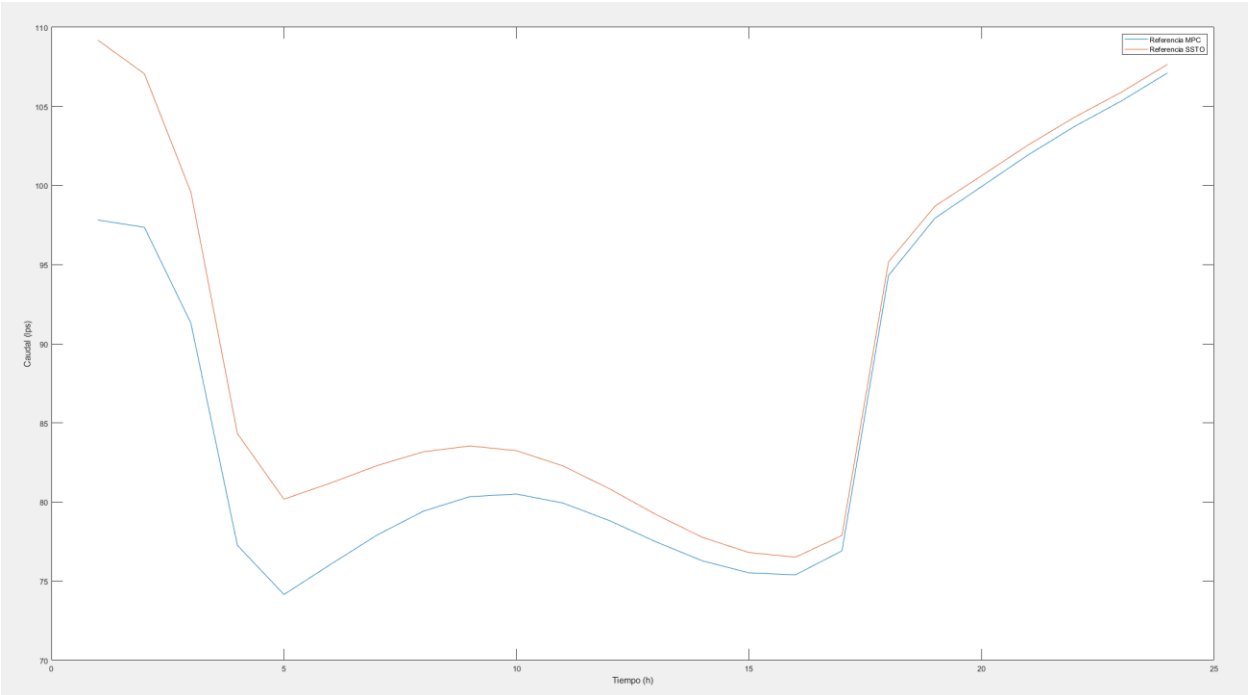

Figura 39 - Seguimiento de referencias del caudal Q_9 del MPC con $Q=10^3$ y $N_p=24$.

Caudal Q_{11}

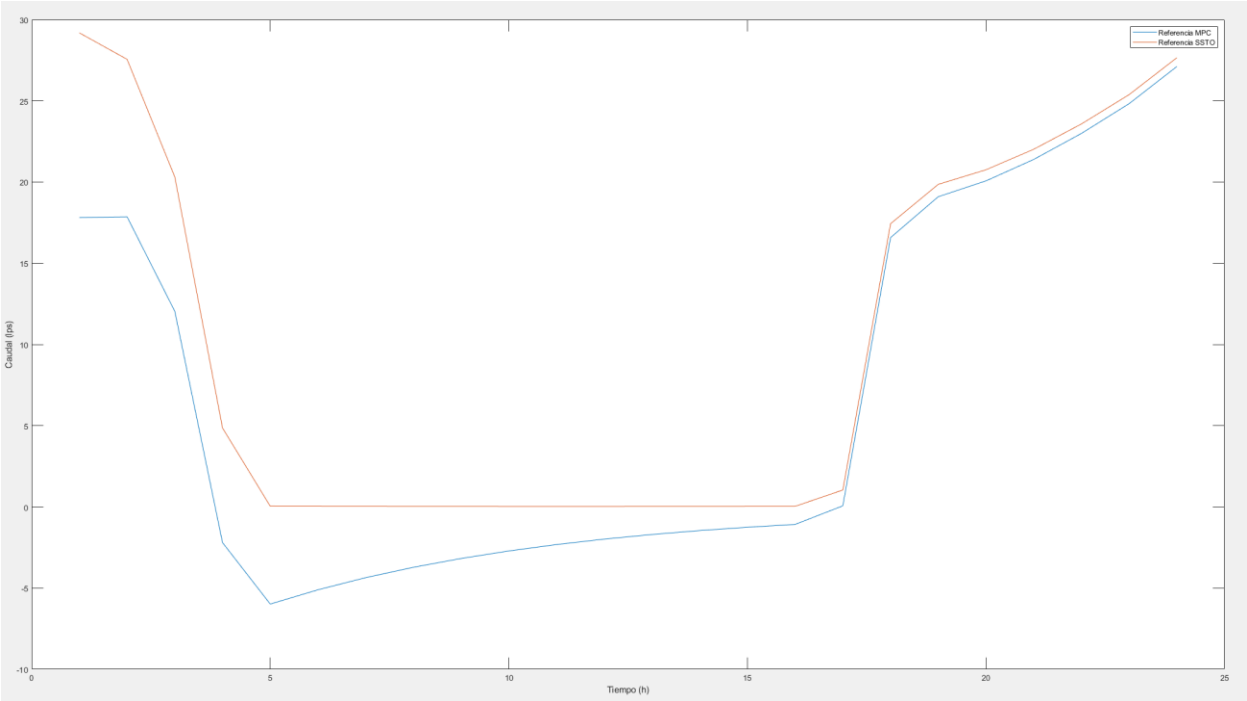

Figura 40 - Seguimiento de referencias del caudal Q_{11} del MPC con $Q=10^3$ y $N_p=24$.

$Q = 10$

Altura del depósito

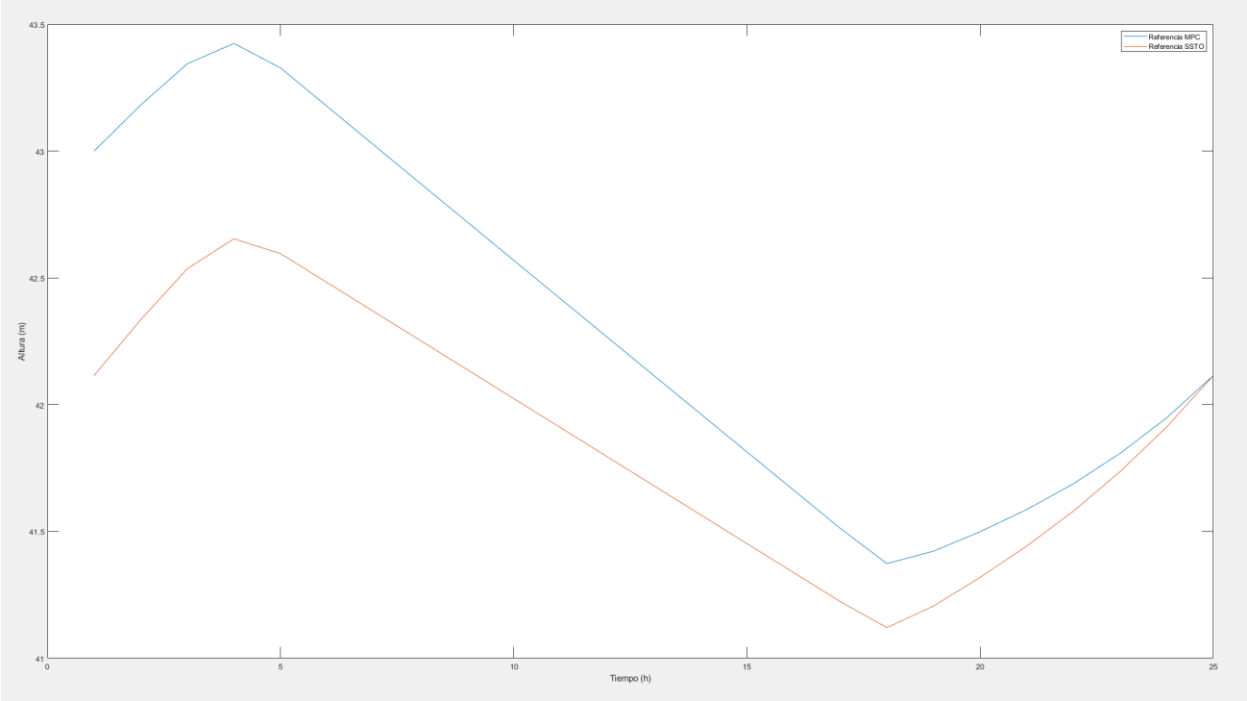

Figura 41 - Seguimiento de referencias de altura del depósito del MPC con Q=10 y $N_p=24$.

Caudal Q_9

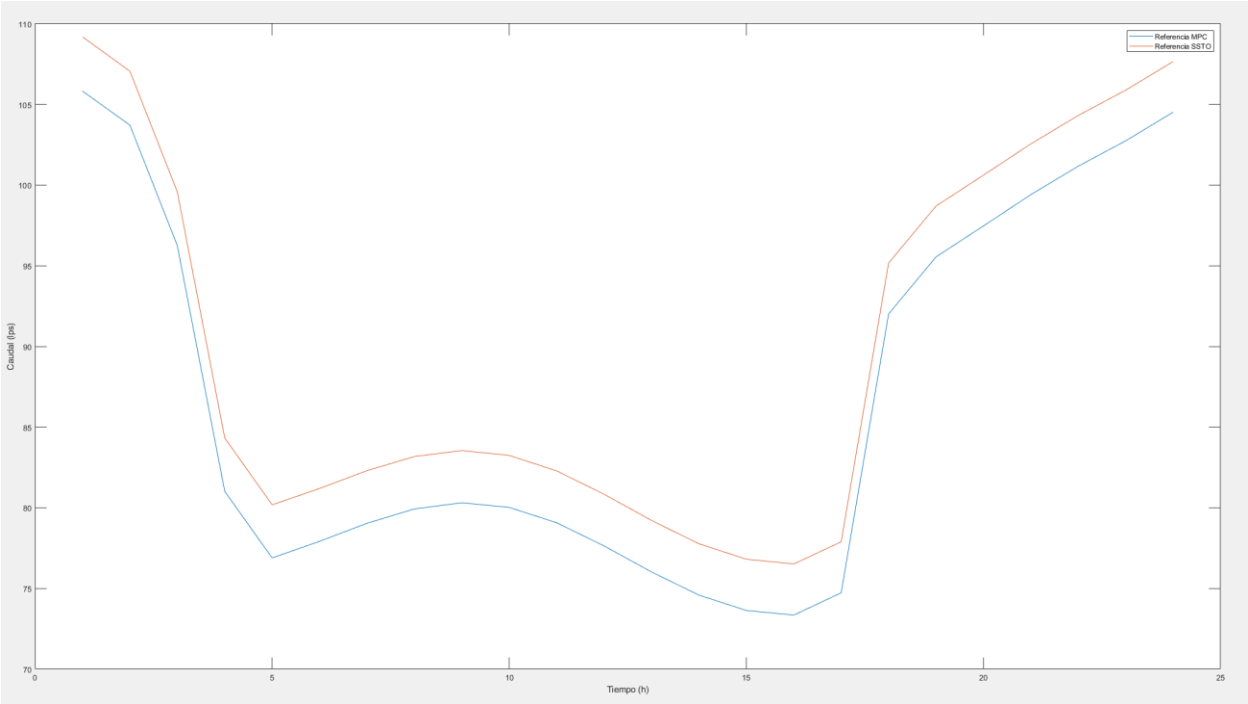

Figura 42 - Seguimiento de referencias del caudal Q_9 del MPC con Q=10 y N_p=24.

Caudal Q_{11}

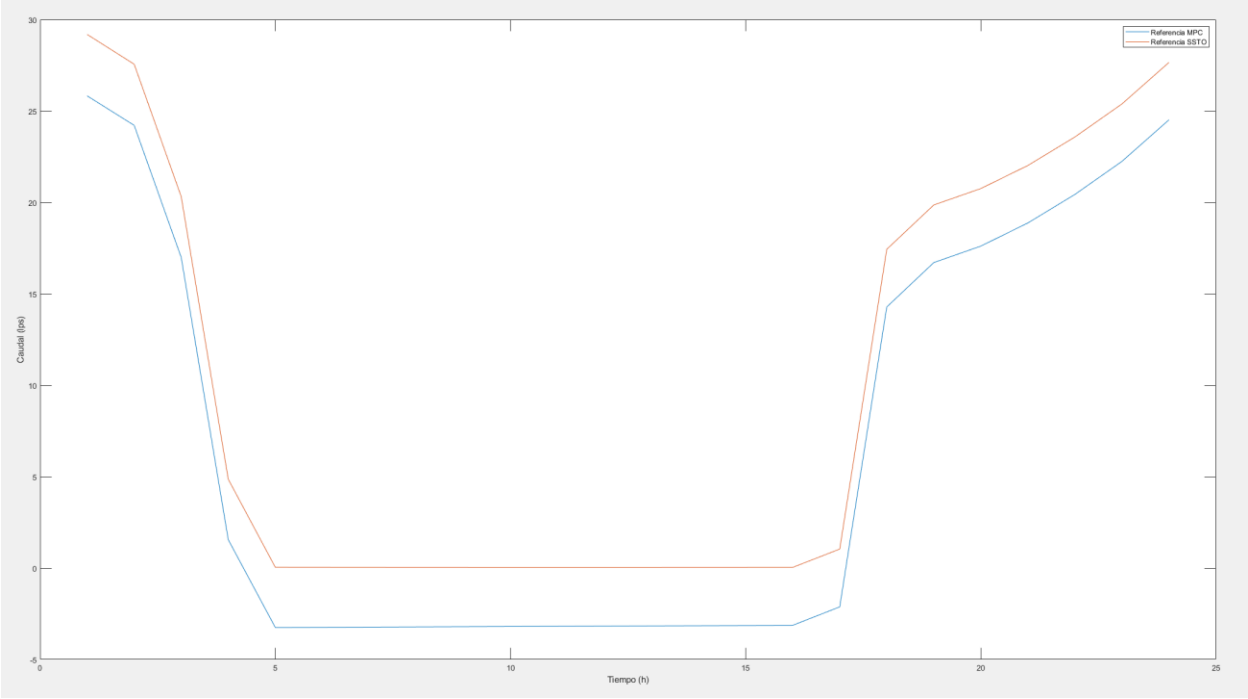

Figura 43 - Seguimiento de referencias del caudal Q_{11} del MPC con Q=10 y N_p=24.

Por lo general, en cualquiera de estos rangos de valores para Q se obtienen resultados bastante aceptables, proporcionando un seguimiento de referencias bastante bueno sin que ello afecte de una manera importante a las variaciones en los caudales. No obstante, se estudiará más en detalle cuál es el efecto de este parámetro, junto con el horizonte de predicción, en la siguiente sección del trabajo, cuando se controlen las variables de actuación de la planta mediante controladores PI siguiendo las referencias que da el MPC.

Con esto termina la sección dedicada a la implementación del SSTO y MPC de la red de aguas, y a continuación se pasa a la última sección de este Trabajo de Fin de Máster, dedicada a la implementación del nivel más bajo de la jerarquía de control y muestra de resultados finales de control de la planta en la escala de tiempo más pequeña con la que se va a trabajar.

7 CONTROL DE BAJO NIVEL DE LA RED DE AGUAS

En esta última sección del trabajo se realizará una descripción de cómo se han implementado las trayectorias que proporciona el MPC en una simulación de duración larga (que será del orden de días, hasta una semana) donde las referencias de caudales del MPC para el instante de tiempo actual se renuevan cada 1 hora y donde se trabaja con un tiempo de integración más pequeño, en este caso, 5 minutos.

Para poder alcanzar las referencias del MPC, que como bien se ha mencionado ya, serán los caudales de la red en cada instante, será necesario implementar el último nivel de la jerarquía de control propuesta para la red: el control de bajo nivel donde cada uno de los caudales o alturas asociados a las entradas manipulables de la red (es decir, la bomba y las dos válvulas) se controlarán mediante controladores PI para poder alcanzar las referencias del MPC.

Por tanto, la sección se dividirá en dos partes principales: en primer lugar, se explicará de forma breve cómo se han diseñado los parámetros de los controladores PI asociados a cada una de las entradas manipulables y, a continuación, se explicará cómo se han implementado estos controladores para tomar las referencias del MPC, obtener las acciones de control que proporcionan los caudales de referencia y simular la respuesta de la planta con un tiempo de integración de 5 minutos. Con esto, se obtendrán una serie de resultados para distintos valores de los parámetros del MPC (matrices Q y R y horizonte de predicción) y se realizará un ajuste de los parámetros de los PI para conseguir el mejor control posible con la estrategia planteada.

7.1. Diseño de controladores PI para las entradas manipulables

Como ya se ha mencionado, las variables a controlar serán los caudales de la red asociados a cada una de la entradas manipulables, es decir, los caudales Q_9, y Q_{11} y la altura H_4, asociados a la velocidad de la bomba, ω_r, la válvula V1 y la válvula V2, respectivamente.

El motivo por el cual la válvula V2 controlará la presión de un nodo y no el caudal de la rama correspondiente siguiendo la referencia del MPC, es porque para el modelo de caudales del MPC no se tiene en cuenta la presión del nodo, y simplemente se fuerza a que el caudal asociado a la rama de la válvula V2 sea igual a la demanda en el nodo N4. Por ello, se ha decidido que sea el control de bajo nivel el que realice el control de dicha presión para mantenerla en un nivel constante razonable para el nodo en las condiciones en las que se trabaja.

Los caudales son variables algebraicas que se verán modificadas en la medida en que las entradas manipulables van a afectar a sus correspondientes términos de la matriz G de pérdidas en las tuberías y de la bomba que, recordando, era la siguiente:

$$G(Q,u) = \begin{bmatrix} 1.418 \cdot 10^{-6} Q_1|Q_1| \\ 0.001873 Q_2|Q_2| \\ 0.001873 Q_3|Q_3| \\ 0.0285 Q_4|Q_4| \\ 0.00615 Q_5|Q_5| \\ 0.00615 Q_6|Q_6| \\ 0.04277 Q_7|Q_7| \\ 7.088 \cdot 10^{-4} Q_8|Q_8| \\ -55\omega_r^2 + 5.8 \cdot 10^{-4} Q_9^2 \\ u_1 Q_{10}|Q_{10}| \\ u_2 Q_{11}|Q_{11}| \end{bmatrix}$$

Donde los términos que interesan ahora son:

$$G_{Q_9} = -55\omega_r^2 + 5.8 \cdot 10^{-4} Q_9^2, \qquad G_{Q_{10}} = u_1 Q_{10}|Q_{10}|, \qquad G_{Q_{11}} = u_2 Q_{11}|Q_{11}|$$

Por tanto, las variables a controlar dependerán directamente del valor de las entradas manipulables, sin existir ninguna dinámica en el sistema siempre y cuando se considere que la dinámica de los actuadores es despreciable, lo cual ha sido el caso para este trabajo.

Teniendo esto en cuenta, las ganancias proporcionales de los controladores PI que asocian entradas manipulables (la bomba y la válvula V1) se van a elegir como las inversas de las ganancias observadas mediante un escalón en cada los caudales con respecto a sus respectivas entradas manipulables. Por otro lado, la ganancia proporcional de la válvula V2 se elegirá inicialmente como la inversa de la variación de la altura en el nodo N4.

Dicho esto, los escalones en las entradas manipulables y su efecto en las correspondientes salidas asociadas se muestran a continuación:

Escalón en ω_r

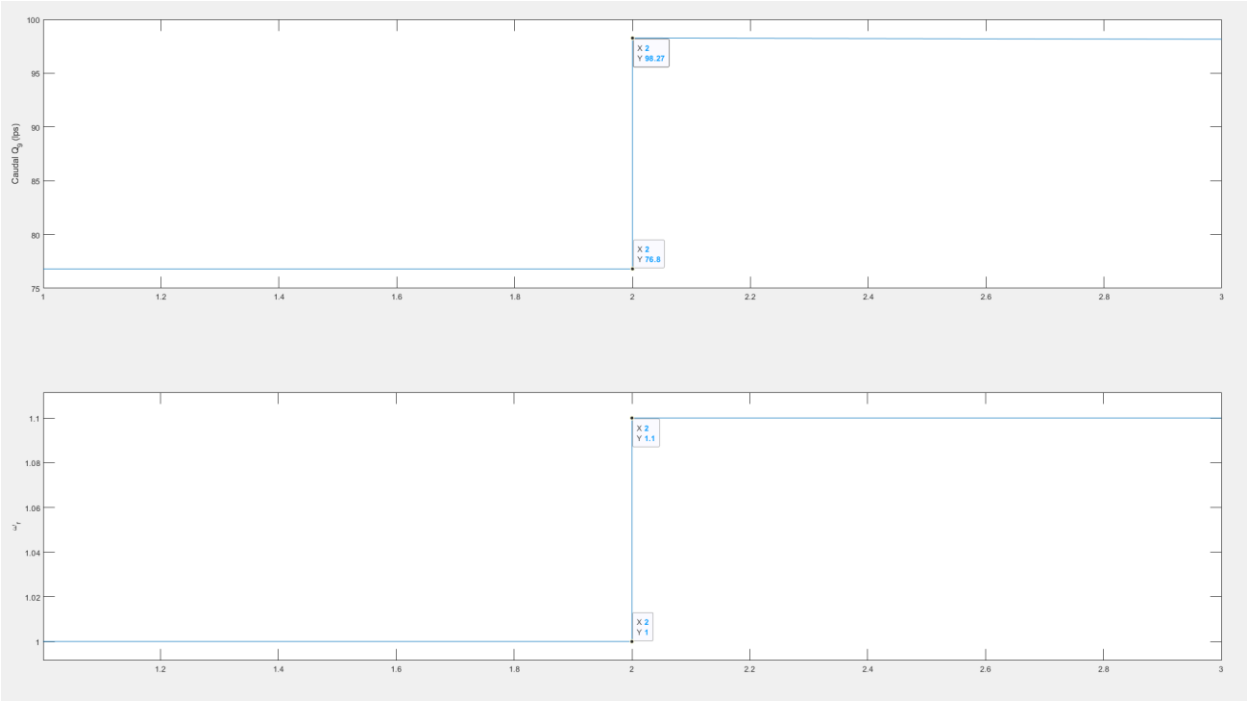

Figura 44 – Escalón en ω_r para elección de K_{p1}.

Escalón en u_2

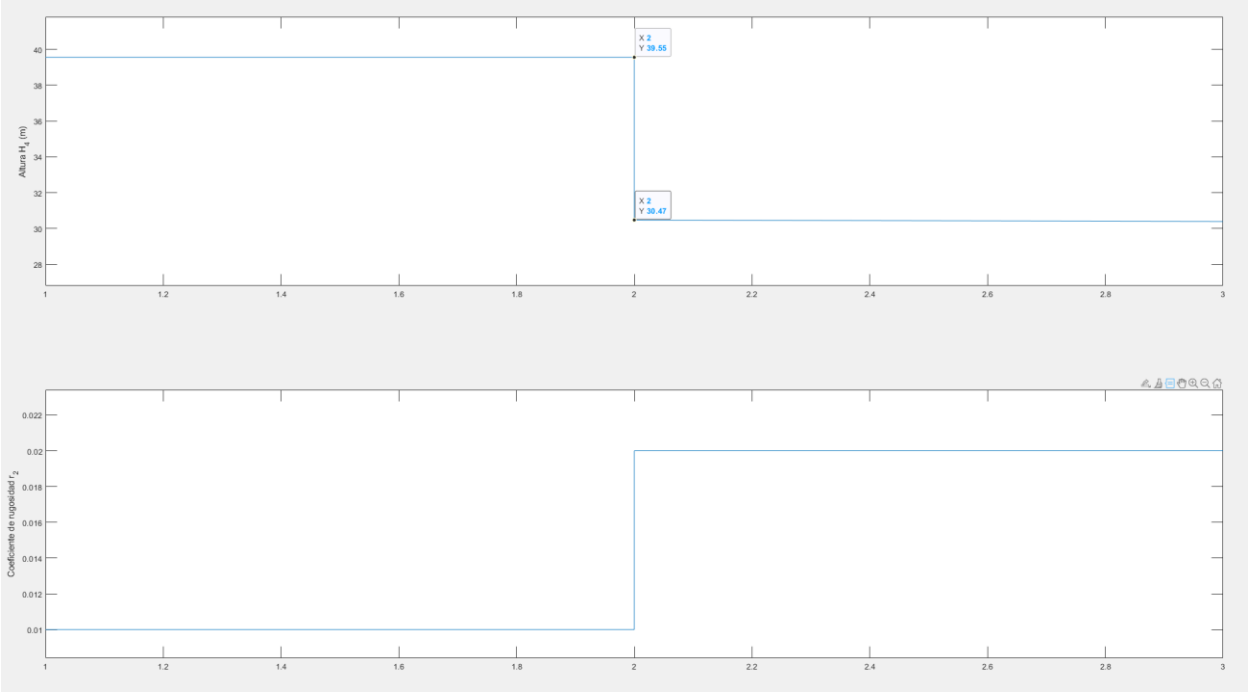

Figura 45 – Escalón en u_2 para elección de K_{p2}.

Escalón en u_1

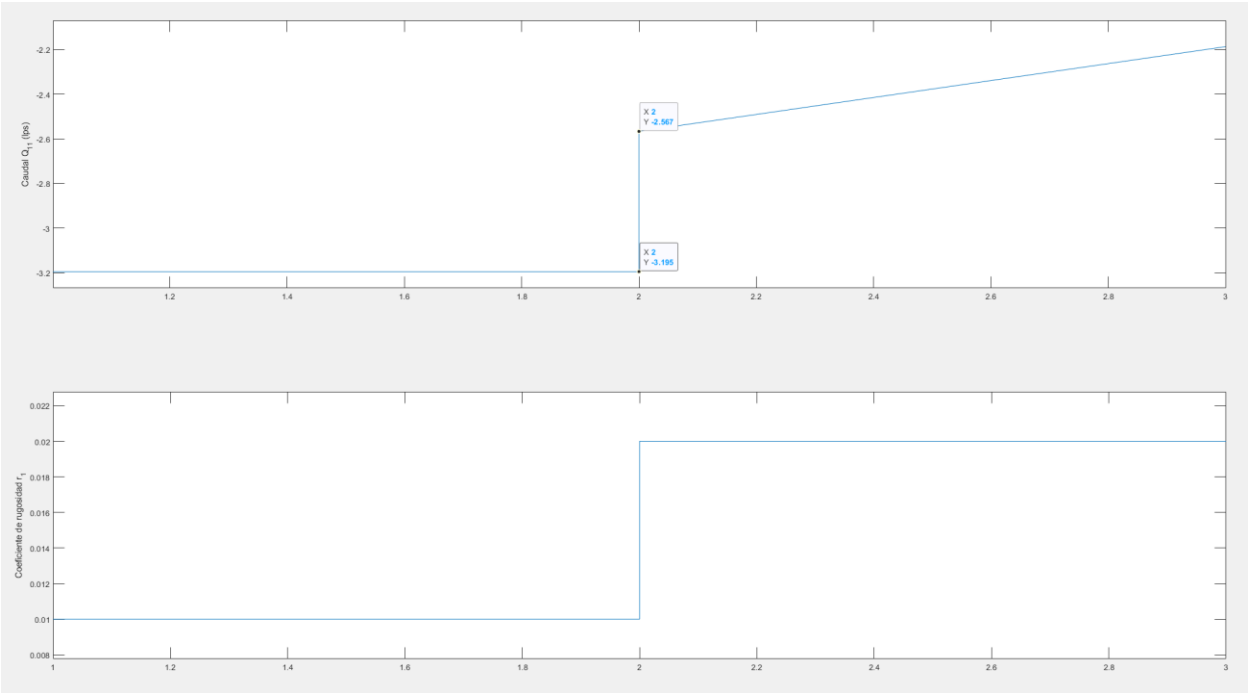

Figura 46 - Escalón en u_1 para elección de K_{p3}.

A partir de los resultados, se han obtenido las siguientes ganancias para los PI:

$$K_{p1} = \frac{\Delta\omega_r}{\Delta Q_9} = \frac{0.01}{21.47} = 4.66 \cdot 10^{-4}$$

$$K_{p2} = \frac{\Delta u_2}{\Delta H_4} = \frac{0.01}{-9.08} = -1.1 \cdot 10^{-3}$$

$$K_{p3} = \frac{0.01}{0.628} = 0.015$$

La ganancia K_{p3}, en realidad, tendrá signo negativo, ya que cuando el caudal es negativo, cerrar la válvula (es decir, aumentar su coeficiente de rugosidad) provoca en realidad una disminución del caudal en términos absolutos. Es conveniente comentar que, idealmente, se debería trabajar con valores absolutos tanto de referencia como medida del caudal, pero se ha comprobado que esto hacer que el control se vuelva inestable hasta el punto en que se llega a un punto que no tiene solución. No obstante, como se comprobará a continuación, aun trabajando con los valores originales de los caudales, se obtienen resultados bastante razonables.

Los tiempos integrales de cada uno de los controladores PI se han obtenido experimentalmente partiendo de un valor de $t_{i1} = t_{i2} = t_{i3} = 10$, y ajustando para cada uno de los controladores conforme se ha comprobado que era necesario un efecto integral mayor o menor.

Con esto, se pasa ahora al apartado de simulación de la planta e implementación de los controladores PI, donde se mostrarán los resultados que estos controladores proporcionan y se realizará el ajuste de los mismos para conseguir mejorar su funcionamiento en la medida de lo posible.

7.2. Simulación de la red y resultados obtenidos por los controladores

Como ya se ha comentado antes, para la simulación de la red se empleará un bucle en el que cada 5 minutos se integrará la red con las consignas de las entradas manipulables que los controladores PI van generando en función a la evolución de la red. De igual manera, la red se simulará en cada instante de integración mediante el script "Simul_EPANET.m" con las demandas y entradas manipulables aplicables en el momento. El script donde se incorpora este bucle para la simulación de la red es 'Simulacion.m'

Por otro lado, en el instante inicial, y cada hora, se ejecutará el MPC, que generará las referencias de caudales y del nivel del depósito a seguir en cada momento en base a las referencias de 24 horas que se han obtenido del RTO de CasADi, por una duración total de 120 horas (menos el horizonte de predicción del MPC, que en un principio será de 24 horas), donde las referencias del RTO se repetirán periódicamente. Estas referencias serán las que se intenten seguir por los controladores de bajo nivel durante la siguiente hora de ejecución del bucle, hasta que el MPC vuelva a generar unas nuevas referencias para la siguiente hora.

De esta manera, en primer lugar, se definen las variables a utilizar por los controladores y el MPC, así como el nivel inicial del depósito y los valores iniciales de las entradas manipulables con los que se va a trabajar, y se inicia el bucle, simulando para obtener los valores iniciales de caudales y alturas. A continuación, se calculan las referencias del MPC y se definen las mismas para que los controladores de bajo nivel trabajen con ellas:

```
    if (mod(k*Tint,Tm)==0)
        [xt,ut]=
MPC_nuevaRed(HxtRTO,QtRTO,Mdl,Hk(end),Dt(1:8,:),1+floor(rem(k*Tint,N)));
        refQ9=ut(9,mod(k*Tint,Tm)+1);
        refH4=35;
        refQ11=ut(11,mod(k*Tint,Tm)+1);
    end
```

La condición mod(k*Tint,Tm)==0 sólo se cumple cuando el instante actual, k*Tint es un múltiplo del tiempo de muestreo de una hora, es decir, cada hora.

A continuación, se actualizan los valores de las variables de control mediante los PI, que incorporan anti wind-up para mantener constante el término integral si la variable de control está saturada:

```
deltauKp=Kp1*(refQ9-Qk(9));
        if u1==Mdl.u1max
            if (deltauKi+Tint*(refQ9-Qk(9))*Kp1/Ti1 > 0)
                u1=u1+deltauKp;
            else
                deltauKi=deltauKi+Tint*(refQ9-Qk(9));
                u1=u1+deltauKp+deltauKi*Kp1/Ti1;
            end

        elseif u1==Mdl.u1min
            if (deltauKi+Tint*(refQ9-Qk(9))*Kp1/Ti1 < 0)
                u1=u1+deltauKp;
            else
                deltauKi=deltauKi+Tint*(refQ9-Qk(9));
                u1=u1+deltauKp+deltauKi*Kp1/Ti1;
            end
        else
            deltauKi=deltauKi+Tint*(refQ9-Qk(9));
            u1=u1+deltauKp+deltauKi*Kp1/Ti1;
        end
```

Y las cuales se filtran para suavizar la acción de control:

```
        uf1=u1+Tint/5*(u1-uf1);

        if uf1<Mdl.u1min
            uf1=Mdl.u1min;
        end

        if uf1>Mdl.u1max
            uf1=Mdl.u1max;
        end
```

Este procedimiento se repite para las tres variables de control, y tras ello se guardan los valores actuales de las referencias, las acciones del control y variables de la red y se vuelve a iterar hasta que se alcance el instante final de simulación.

A continuación, se presentan los resultados obtenidos por los controladores, primero para los parámetros iniciales que se han calculado en el apartado anterior, y posteriormente con valores ajustados de los mismos y distintos horizontes de predicción del MPC, para comprobar cuáles son las diferencias entre éstos.

Es conveniente mencionar que para el valor inicial de $Q = 10^3$ que se había querido utilizar anteriormente, la red llegaba a un punto en el que no tenía solución, por lo que se ha cambiado el valor de Q a 100 para los resultados que se presentan a continuación.

7.2.1. Resultados obtenidos por los controladores de bajo nivel

Parámetros iniciales de los controladores PI y horizonte de predicción N_p=24

Caudal Q_9

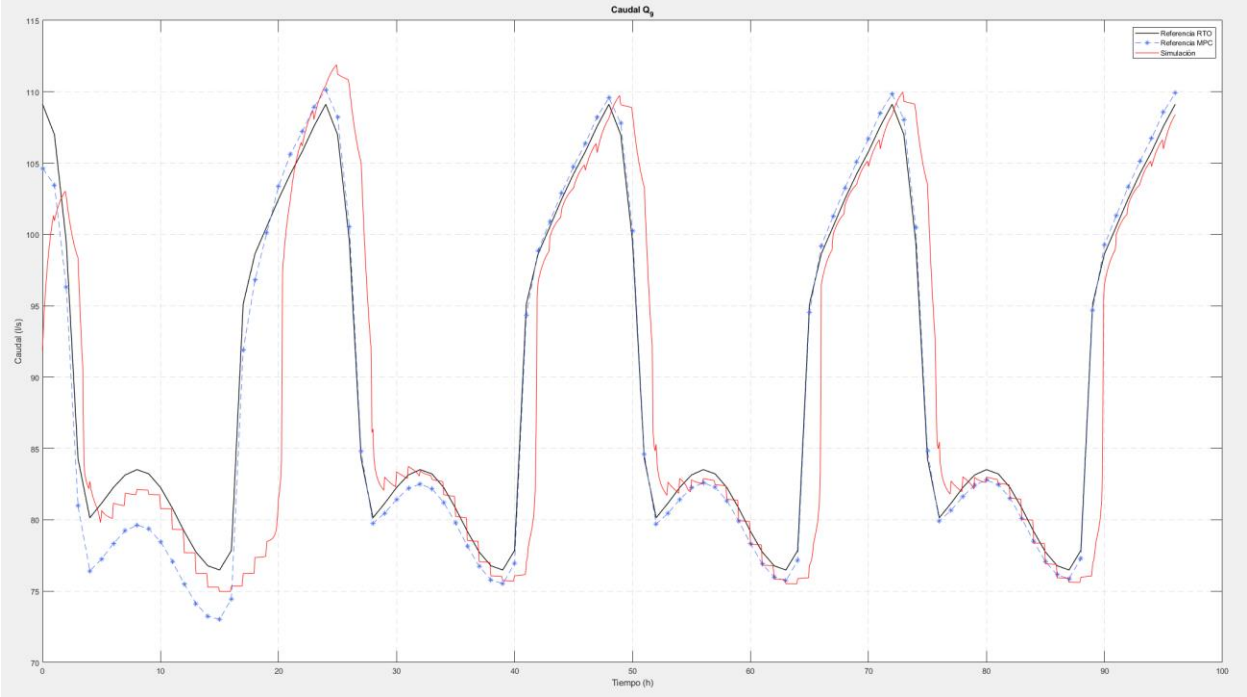

Figura 47 – Comparación de las referencias de RTO y MPC con resultados de simulación para Q_9.

Caudal Q_{11}

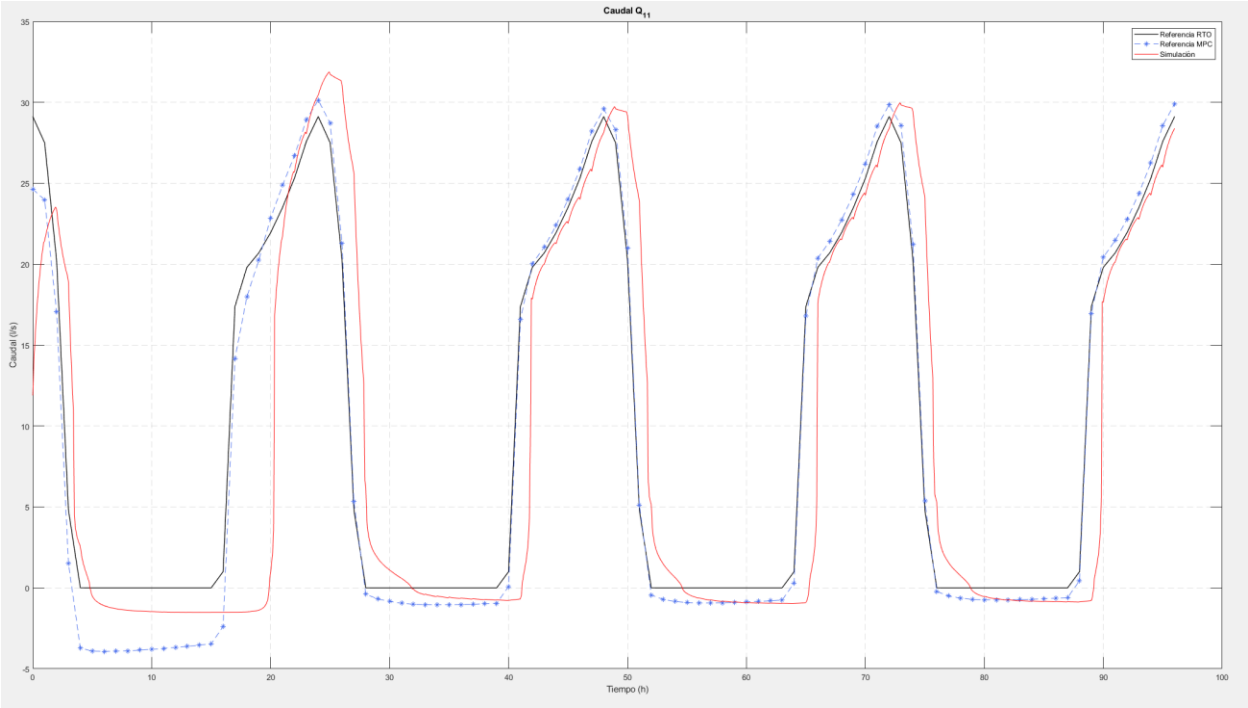

Figura 48 - Comparación de las referencias de RTO y MPC con resultados de simulación para Q_{11}.

Altura del nodo 4

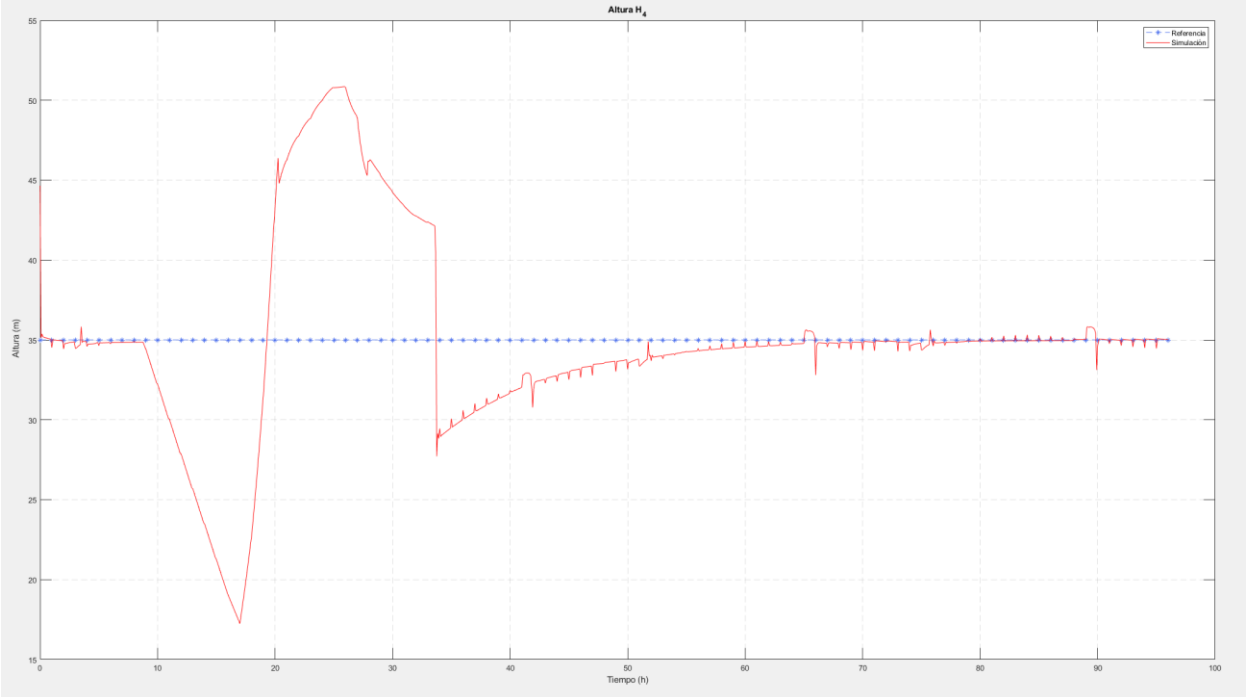

Figura 49 - Comparación de las referencias de RTO y MPC con resultados de simulación para H_4.

Altura del depósito

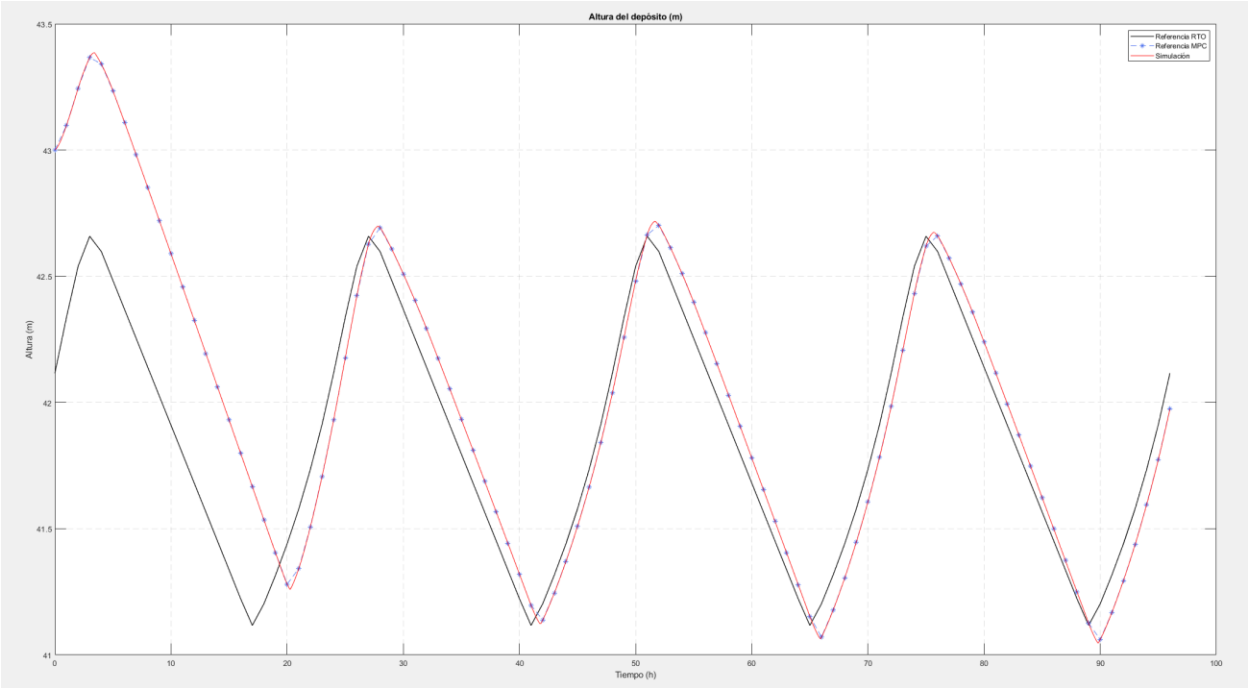

Figura 50 - Comparación de las referencias de RTO y MPC con resultados de simulación para la altura del depósito.

En general, los resultados son bastantes buenos en cuanto al seguimiento de referencias del nivel del depósito, pero son bastante mejorables para ambos caudales y, sobre todo, para la altura del nodo N4. Además, parece el seguimiento de la referencia del nivel óptimo del depósito se lleva a cabo con un cierto retraso.

En la siguiente tanda de resultados, se muestran los cambios que se han producido al cambiar algunos de los parámetros de los PI.

Controladores PI ajustados y horizonte de predicción N_p=24

En este caso, los parámetros de los PI han pasado a ser:

$$K_{p1} = 0.001, \qquad t_{i1} = 10$$
$$K_{p2} = -0.001, \qquad t_{i2} = 5$$
$$K_{p3} = -0.003, \qquad t_{i3} = 10$$

En general, se ha comprobado que había un gran acoplamiento entre el controlador de Q_{11} y el de H_4, haciendo que cuando el primero fuese más agresivo, el segundo llegase a situaciones con presiones negativas. Por ello, se ha relajado este controlador y se han hecho más agresivos los otros dos, proporcionando los siguientes resultados:

Caudal Q_9

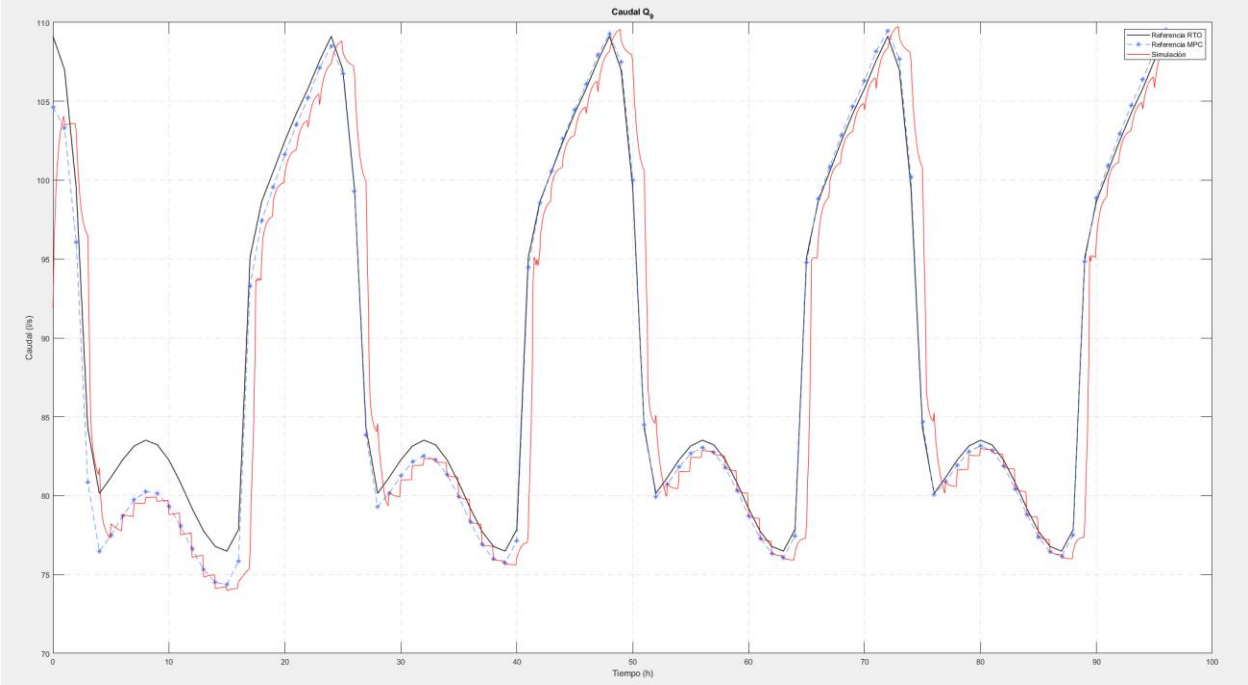

Figura 51 - Comparación de las referencias de RTO y MPC con resultados de simulación para Q_9. Parámetros de los controladores PI modificados.

Caudal Q_{11}

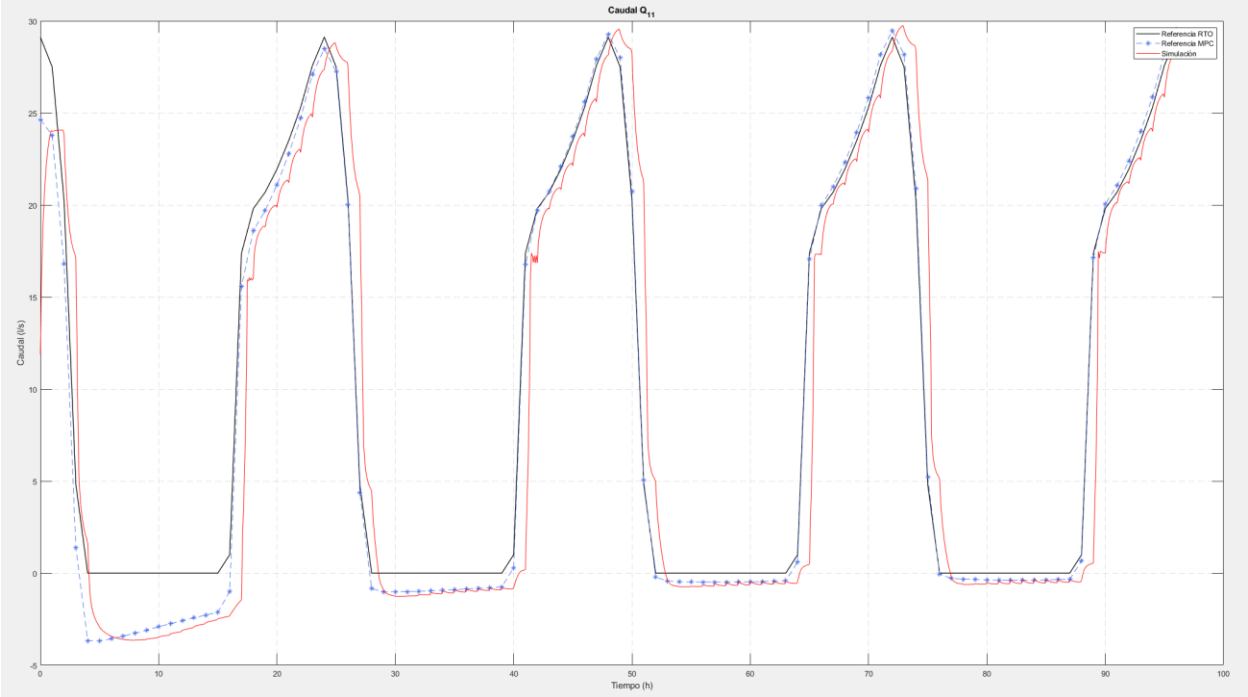

Figura 52 - Comparación de las referencias de RTO y MPC con resultados de simulación para Q_{11}. Parámetros de los controladores PI modificados.

Altura del nodo 4

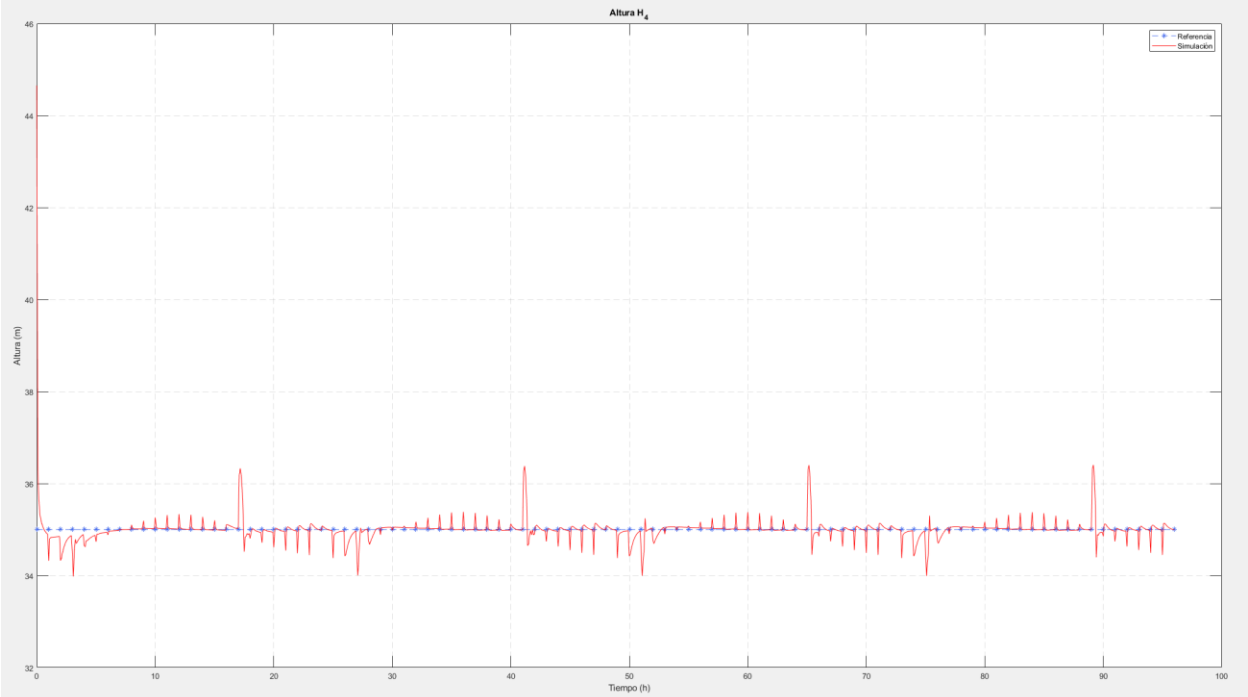

Figura 53 – Comparación de las referencias de RTO y MPC con resultados de simulación para H_4. Parámetros de los controladores PI modificados.

Altura del depósito

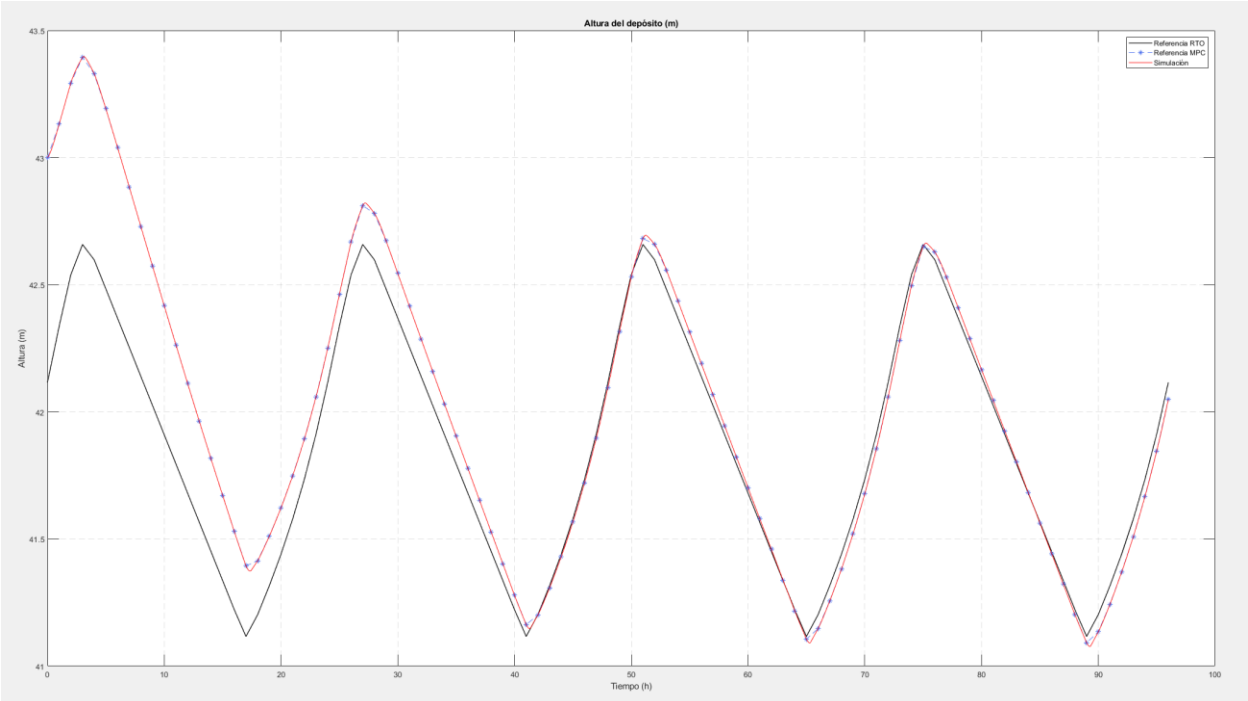

Figura 54 - Comparación de las referencias de RTO y MPC con resultados de simulación para el nivel del depósito. Parámetros de los controladores PI modificados.

En este caso, se obtienen resultados bastante mejores para todas las variables controladas. Ambos caudales siguen las referencias que proporciona el MPC con mayor rapidez, la altura del nodo 4 se lleva hasta la de referencia desde el primer instante y no hay mucha variación con respecto a ésta, y el retraso que existía antes en el seguimiento del nivel del depósito ha desaparecido.

Por lo general, se pueden aceptar estos parámetros de los PI como finales. Ahora se pasa a comprobar cuál es el efecto de un cambio en el horizonte de predicción del MPC en las trayectorias que sigue la red.

Horizonte de predicción Np=8

Caudal Q_9

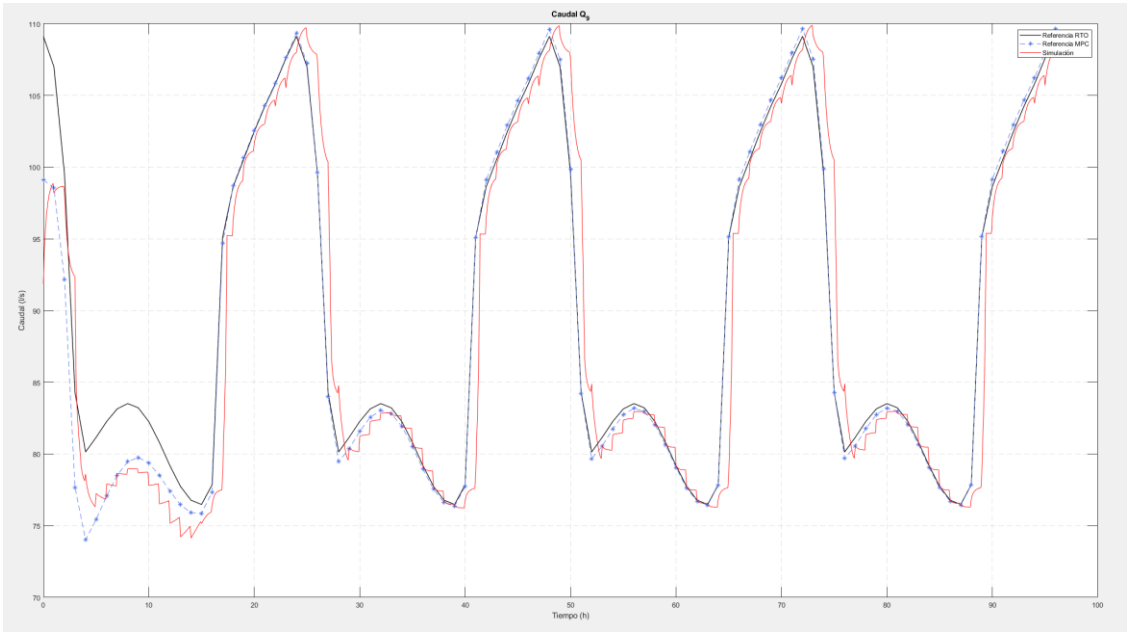

Figura 55 - Comparación de las referencias del RTO y MPC con resultados de simulación para el caudal Q_9. Horizonte de predicción Np=8.

Caudal Q_{11}

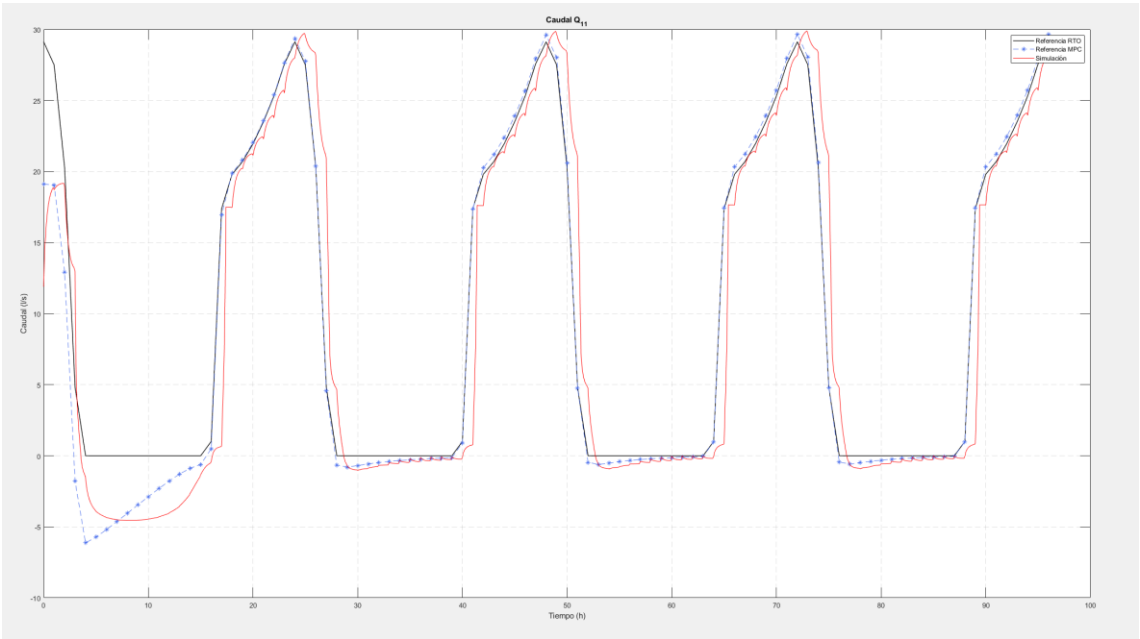

Figura 56 - Comparación de las referencias del RTO y MPC con resultados de simulación para el caudal Q_{11}. Horizonte de predicción Np=8.

Altura del nodo 4

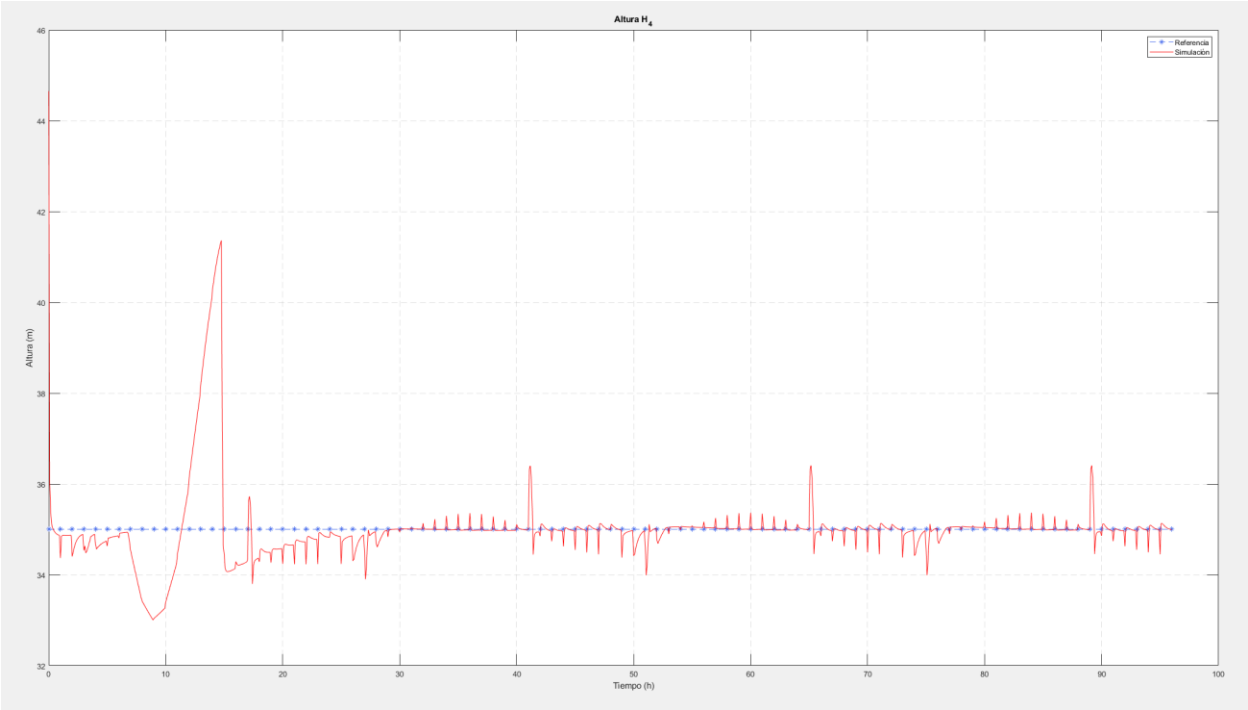

Figura 57 - Comparación de las referencias del RTO y MPC con resultados de simulación para la altura del nodo 4. Horizonte de predicción Np=8.

Altura del depósito

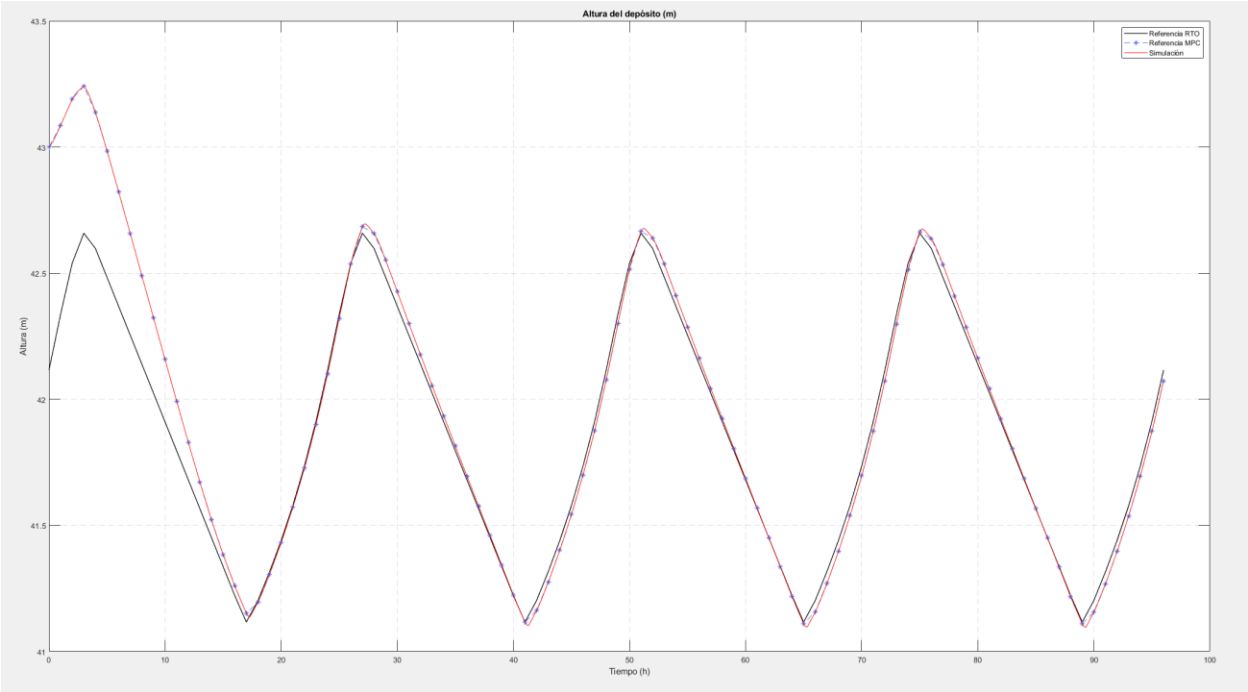

Figura 58 - Comparación de las referencias del RTO y MPC con resultados de simulación para la altura del depósito. Horizonte de predicción Np=8.

Horizonte de predicción Np=4

Caudal Q_9

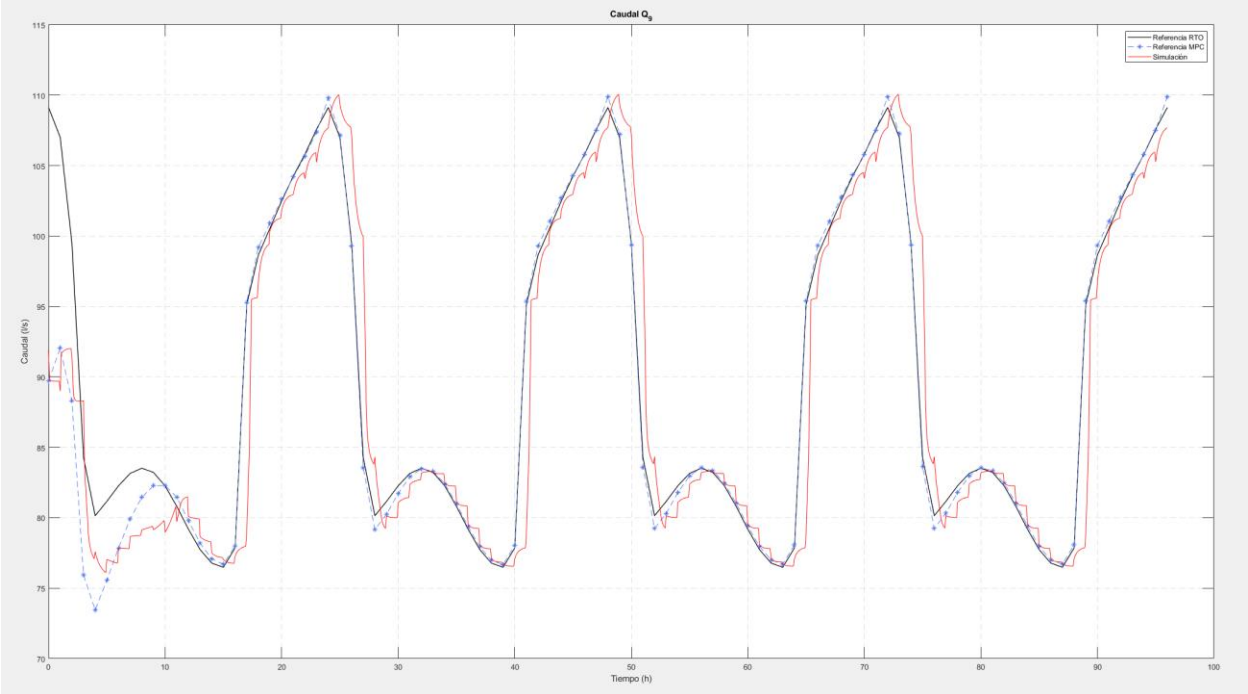

Figura 59 – Comparación de las referencias del RTO y MPC con resultados de simulación para el caudal Q_9. Horizonte de predicción Np=4.

Caudal Q_{11}

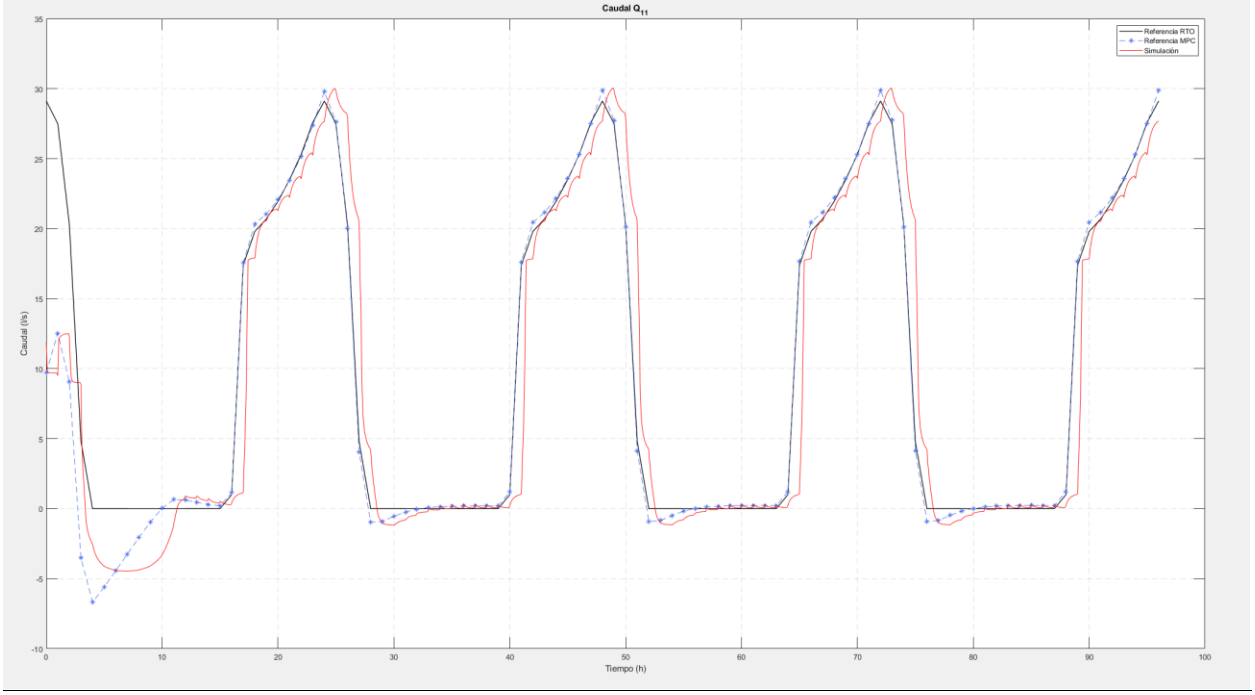

Figura 60 - Comparación de las referencias del RTO y MPC con resultados de simulación para el caudal Q_{11}. Horizonte de predicción Np=4.

Altura del nodo 4

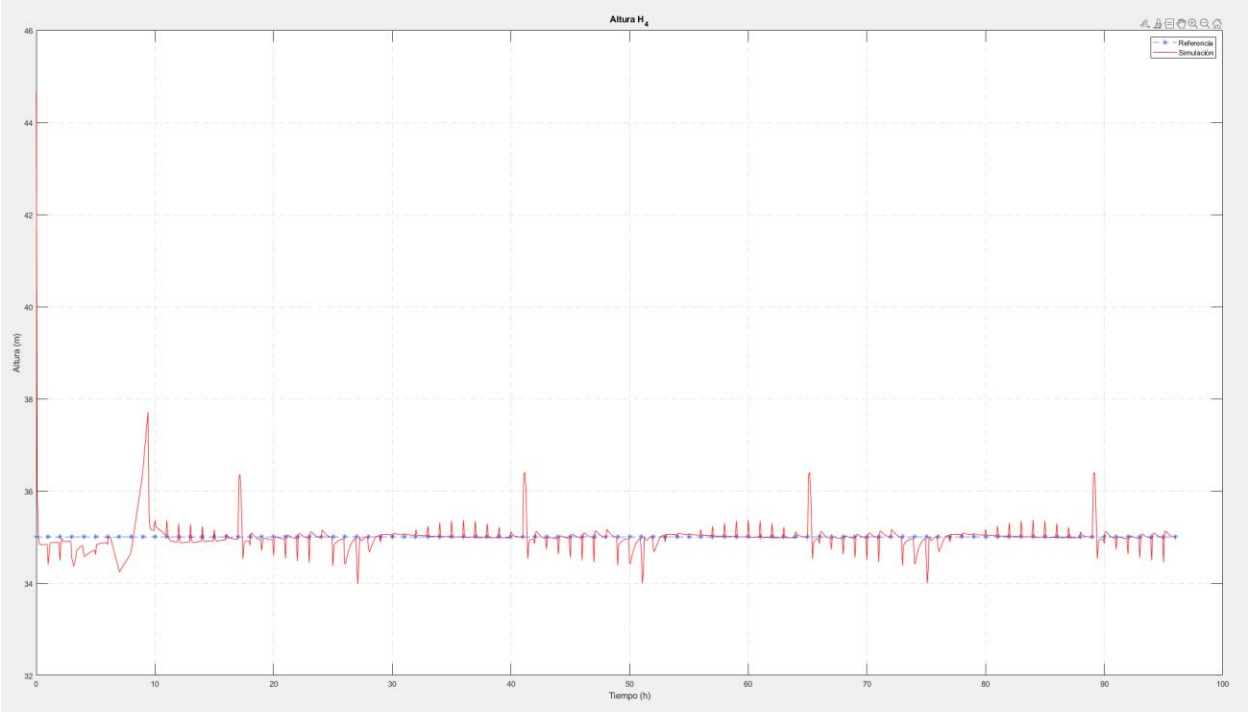

Figura 61 - Comparación de las referencias del RTO y MPC con resultados de simulación para la altura del nodo 4. Horizonte de predicción Np=4.

Altura del depósito

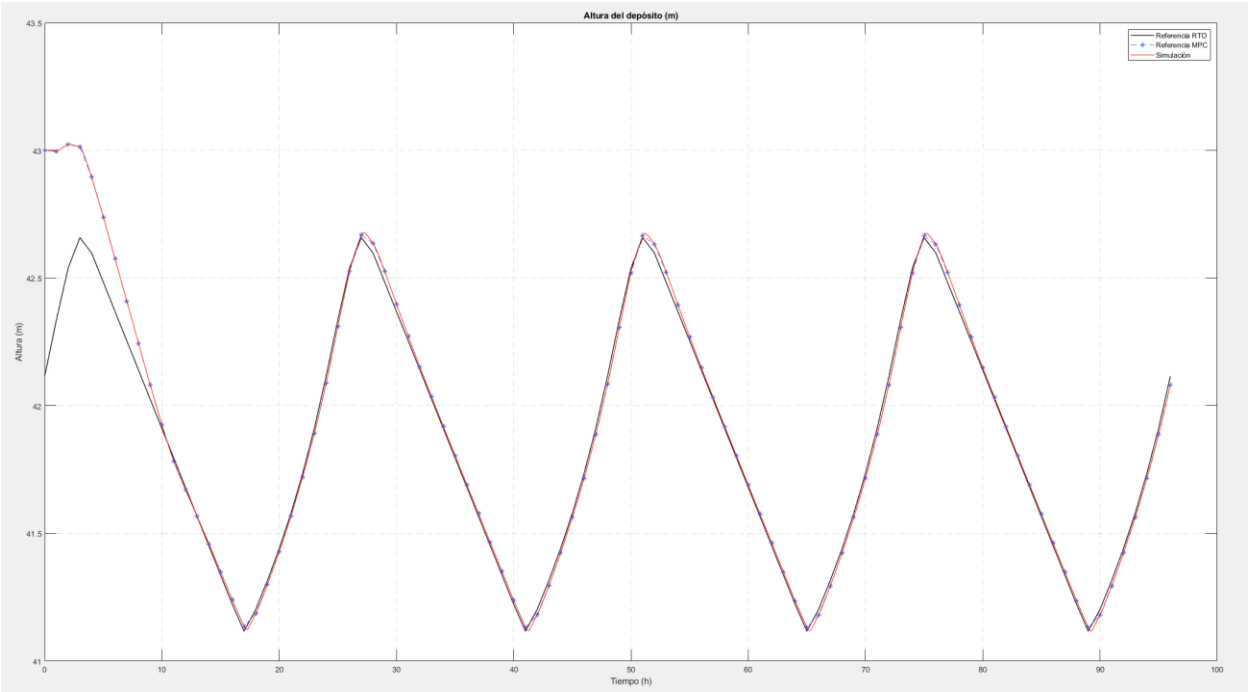

Figura 62 - Comparación de las referencias del RTO y MPC con resultados de simulación para la altura del depósito. Horizonte de predicción Np=4.

Los resultados con horizonte de predicción Np=8 son bastante comparables a los que se tenían para horizonte de predicción Np=24, siendo la mayor diferencia el período inicial, desde que se parte de un punto lejano a la trayectoria del RTO, hasta que se consigue llegar hasta ésta, punto en el cual la diferencia entre ambos horizontes de predicción son prácticamente inapreciables, e incluso hacen mejor el seguimiento de referencias del caudal Q_{11} y de la altura del depósito, aunque algo peor para la altura del nodo 4.

Por otro lado, con horizonte de predicción Np=4, el seguimiento de referencias en el período inicial es algo peor que para el caso de Np=8, con la excepción del control de la altura del nodo 4 que, al verse menos perturbado por los controles de los caudales, incluso llega a mejorar con respecto al caso con horizonte de predicción mayor. En todo caso, los resultados a partir de dicho período inicial siguen siendo, al igual que para Np=8, bastante buenos, lo cual es muy positivo ya que, a la vez, se está reduciendo el coste computacional del MPC a prácticamente la mitad.

En general, tras los cambios realizados sobre los parámetros del PI, los resultados obtenidos para el conjunto de la jerarquía de control han sido buenos. La trayectoria generada por el MPC con bastante facilidad la referencia marcada por el RTO, y los controladores PI son capaces de llevar a la red a dicha referencia, además de mantener el nivel del nodo 4 en una altura controlada.

Con esto termina la implementación de la jerarquía de control completa que se había planteado para este Trabajo Fin de Máster. A continuación, se realizan algunas conclusiones sobre el trabajo realizado y propuestas para futuras mejoras con respecto a algunas partes del trabajo.

8 CONCLUSIONES Y POSIBLES MEJORAS

Durante este Trabajo de Fin de Máster se ha llevado a cabo la implementación del modelo de una red de aguas y de una jerarquía de control predictivo para la cual se han probado distintas alternativas tanto en lo que respecta a implementación de la red en sí para su simulación, como para la implementación del RTO que determina la trayectoria que debe seguir la planta para conseguir el coste económico óptimo asociado a su operación.

En primer lugar, se ha hecho una descripción de cómo se comporta y cómo se modela una red de aguas, los costes y restricciones más importante que existen en su operación, y cuál ha sido la red de aguas que ha sido caso de estudio.

A continuación, se ha hecho una introducción a EPANET como software de simulación de redes de agua y sus diferentes funcionalidades, y del toolbox de implementación de EPANET en Matlab. Con este toolbox se ha trabajado para simular la red desde Matlab y se ha utilizado en conjunto con los algoritmos de optimización de punto interior, sqp y algoritmo genético que incorpora Matlab para implementar un RTO que calcule trayectorias óptimas bajo restricciones durante un período de 24 horas, para lo que cual se han obtenido resultados mejorables.

Después, se ha introducido a CasADi como herramienta de diferenciación algorítmica con la cual se ha propuesto una implementación alternativa del simulador de la red y del optimizador. En este caso, los resultados han sido bastante mejores que para los algoritmos anteriores y han sido los que se han utilizado para el resto de los niveles de la jerarquía de control.

Una vez obtenidos los resultados de CasADi, se ha implementado el nivel de control avanzado de la red, compuesto por un SSTO que adapta las referencias que se obtienen del RTO a las de un modelo lineal, y el MPC que, partiendo de la situación actual en la que se encuentra la red, calculará la trayectoria óptima que deberá seguir la planta para alcanzar la referencia marcada por el SSTO.

Por último, se ha implementado el nivel de control más bajo de la jerarquía, compuesto por controladores PI mediante los cuales se controlan las entradas manipulables de la red para conseguir alcanzar las referencias marcadas por el MPC. Los resultados obtenidos controlando la red a partir de la jerarquía de control completa han sido mostrados en esta sección y, por lo general, han sido bastante buenos tras realizar ajustes en los parámetros de los controladores PI.

No obstante, el trabajo realizado podría ser mejorado en el futuro en distintos aspectos. Específicamente, la implementación de los optimizadores de la red utilizando los algoritmos de Matlab y el simulador de EPANET no ha dado muy buenos resultados y, para lo cual, quizá se podría plantear el problema de optimización de una forma distinta que sea más acorde a la forma en que estos algoritmos trabajan, de manera que dicho problema tenga un mayor grado de convergencia y se consiga llegar a una solución razonable.

Por otro lado, se han tenido dificultades a la hora de implementar el simulador de la red y el optimizador de CasADi con distintos tipos de integradores y optimizadores a los utilizados en este trabajo, algunos de los cuales parecen funcionar de una manera más eficiente y proporcionan los mismos resultados en un período de tiempo mucho menor.Estas dificultades han venido tanto a la hora de conseguir el mismo grado de convergencia que se tiene con la configuración actual, como a la hora de conseguir que muchos de los optimizadores que CasADi incorpora funcionen, ya que se han tenido problemas obteniendo numerosos plugins de los que éstos hacen uso.

9 REFERENCIAS

[1] «Asignatura: Control de Sistemas de Distribución,» Máster en Ingeniería Electrónica, Robótica y Automática, Escuela Técnica Superior de Ingeniería, Universidad de Sevilla, 2021.

[2] L. T. Biegler, Nonlinear programming: concepts, algorithms, and applications to chemical processes, Society for Industrial and Applied Mathematics: Mathematical Programming Society, 2010, p. 12.

[3] M. K. S. V. a. M. P. D.G. Eliades, «PANET-MATLAB Toolkit: An Open-Source Software for Interfacing EPANET with MATLAB",» de *Proc. 14th International Conference on Computing and Control for the Water Industry (CCWI)*, The Netherlands, Nov 2016.

[4] R. A. F. &. B. L. T. Findeisen, Assessment and future directions of nonlinear model predictive control. (1st ed. 2007.), Springer, 2007.

[5] A. &. T. M. Nemirovski, Interior-point methods for optimization. Acta Numerica, 17, 191-234, Cambridge University Press, 2008.

[6] P. &. T. J. Boggs, Sequential Quadratic Programming. Acta Numerica, 4, 1-51, Cambridge University Press, 1995.

[7] J. McCall, «Genetic algorithms for modelling and optimisation,» *Journal of Computational and Applied Mathematics,* vol. 184, nº 1, pp. 205-222, 2005.

[8] J. G. a. G. H. Joel A E Andersson, «CasADi - A software framework for nonlinear optimization,» *Mathematical Programming Computation,* vol. 11, nº 1, pp. 1-36, 2019.

ANEXO I: CÓDIGOS

1.1. Script 'Parametros_Modelo.m'

```
A=[1,0,0,0,0,0,0,0,-1,0,0;
   0,-1,-1,0,0,0,0,0,1,0,0;
   0,1,0,0,1,0,0,0,0,-1,0;
   0,0,0,0,0,0,0,0,0,1,0;
   0,0,1,-1,0,0,-1,0,0,0,0;
   0,0,0,1,-1,-1,0,0,0,0,0;
   0,0,0,0,0,1,1,0,0,0,-1;
   0,0,0,0,0,0,0,-1,0,0,1];

Ar=[-1 0 0 0 0 0 0 0 0 0 0];
At=[0 0 0 0 0 0 0 1 0 0 0];
Hr=0.5;
S=[314];
B=[0 0 0 0 0 0 0 3.6/S 0 0 0]; %Se divide por 1000 para trabajar con litros
D=[0 10 5 40 5 20 0 0]';
Hxmin=[41];
Hxmax=[59];

Hxnsmin=[45];
Hxnsmax=[55];

Vmin=[sum(D)*3.6*10]; %El volumen mínimo es el correspondiente a suplir
durante 2 horas la demanda (en m^3)

Hmin=0; %Presiones minima y maxima en cada uno de los nodos (en m)
Hmax=70;

Z=[0.5;3;2;2;0.5;4;17;17];

Tm=1; %Tiempo de muestreo en horas
```

```
umax=1e7;
u1max=1.4;

umin=1e-7;
u1min=0.2;

Mdl.A=A;
Mdl.Ar=Ar;
Mdl.At=At;
Mdl.B=B;
Mdl.S=S;
Mdl.Hr=Hr;
Mdl.Tm=Tm;
Mdl.Vmin=Vmin;
Mdl.Hmin=Hmin;
Mdl.Hmax=Hmax;
Mdl.Hxmin=Hxmin;
Mdl.Hxmax=Hxmax;
Mdl.Hxnsmin=Hxnsmin;
Mdl.Hxnsmax=Hxnsmax;
Mdl.umin=umin;
Mdl.u1min=u1min;
Mdl.umax=umax;
Mdl.u1max=u1max;
Mdl.Z=Z;
```

1.2. Script ‘Simul_Epanet.m’

```
function [H,Q]=Simul_EPANET(r,tsim,Hxini,uk,D)

r.setTimeSimulationDuration(tsim);
r.setTimeReportingStep(300);
r.openHydraulicAnalysis;
r.initializeHydraulicAnalysis;
```

```
%Se asignan las alturas iniciales de los tanques y se actualizan los
%valores de las entradas y de las demandas

r.setNodeTankInitialLevel(Hxini'-38);
if not(isempty(uk))
    r.setLinkSettings(9,uk(1));
    r.setLinkSettings(10,(300^4/8.2627e4)*uk(2));
    r.setLinkSettings(11,(300^4/8.2627e4)*uk(3));
end
r.setNodeBaseDemands(D');

%El tiempo de integración es de 5 minutos

tstep=300;P=[];T_H=[];D=[];H=[];Q=[];
while (tstep>0)
    t=r.runHydraulicAnalysis;
    P=[P; r.getNodePressure];
    D=[D; r.getNodeActualDemand];
    H=[H; r.getNodeHydaulicHead];
    Q=[Q; r.getLinkFlows];
    T_H=[T_H; t];
    tstep=r.nextHydraulicAnalysisStep;
end
r.closeHydraulicAnalysis;
```

1.3. Script ‘Prediccion.m’

```
function [Hxt,Ht,Qt]=Prediccion(r,Hx0,Ut,Dt)
Hxt=[];
Ht=[];
Qt=[];
N=(size(Dt,2));
Ut=reshape(Ut,3,N+1);
```

```
for k=0:N

    if k==0
        [Hk,Qk]=Simul_EPANET(r,1,Hx0,Ut(:,1),Dt(:,1));
        Hxk=Hk(1,10)';
        Hxt=[Hxt,Hxk];
        Ht=[Ht,Hk(1,1:8)'];
        Qt=[Qt,Qk(1,:)'];
    else
        uk=Ut(:,k+1);
        Dk=Dt(:,k);
        [Hk,Qk]=Simul_EPANET(r,3600,Hxk,uk,Dk); %Se simula durante 1 hora,
el valor de Tm
        Hxk=Hk(end,10)';
        Hxt=[Hxt,Hxk];
        Ht=[Ht,Hk(end,1:8)'];
        Qt=[Qt,Qk(end,:)'];

    end

%Si se detecta que el problema no tiene solucion, se rompe el bucle y se
%devuelven vacias las alturas y los caudales

if isempty(Hk)
    break;
end

end

if isempty(Hk)
    Hxt=[];
    Ht=[];
    Qt=[];
end
```

1.4. Script 'RTO.m'

```
%% Inicializacion de variables
clear all;

ParametrosModelo;

u01=1;
u02=0.001;
u03=0.001;

%Definición de la demanda

for i=1:24
    Dt(1,i)=0;
    Dt(2,i)=-3*sin(2*pi*(i-1)/11.5);
    Dt(3,i)=0;
    Dt(4,i)=2*sin(2*pi*(i-1)/23);
    Dt(5,i)=0;
    Dt(6,i)=1*sin(2*pi*(i-1)/11.5);
    Dt(7,i)=0;
    Dt(8,i)=0;
    Dt(9,i)=0;
    Dt(10,i)=0;
end
%Dn=repmat([0 10 5 40 5 20 0 0 0 0]',1,24);
Dn=repmat([0 10 5 30 5 20 10 10 0 0]',1,24); %Demanda alternativa
Dt=Dt+Dn;
N=size(Dt,2)/Mdl.Tm;

%Carga de la red Epanet

r=epanet('EjemploEPANET2_bomba_cambiada.inp');


%Niveles de los tanques y entradas iniciales
```

```
Hxini=48.07;
uk=[u01 u02 u03]';

%Se inicializa el vector de entradas para todo el período de demandas

Utini=kron(ones(N+1,1),uk);

%Definicion de las matrices de restriccion lineales para la variacion en
%ambas entradas

Ai=zeros(2*N,N);
bi=zeros(2*N,1);
Ai2=zeros(2*N,N);
bi2=zeros(2*N,1);

for jj=1:N
    Ai(jj,jj*3:jj*3+5)=[-1 0 0 1 0 0];
end
Ai(N+1,3)=1;
Ai(N+1,3*(N+1))=-1;

bi(1:2*(N+1))=0.1;
Ai(N+2:2*(N+1),:)=-Ai(1:(N+1),:);

%Calculo RTO


Coste=@(z)CalculoCostes(r,Mdl,z(1),z(2:end),Dt);
Restr=@(z)Restricciones(r,Mdl,z(1),z(2:end),Dt);

z0=[48.07;Utini];
zmin=[Mdl.Hxmin';repmat([Mdl.u1min Mdl.umin Mdl.umin]',N+1,1)];
zmax=[Mdl.Hxmax';repmat([Mdl.u1max Mdl.umax Mdl.umax]',N+1,1)];

Az=Ai(:,2:end);
```

```
bz=bi;


%Opciones para el fmincon


options=optimoptions('fmincon');
options=optimoptions(options,                    'ConstraintTolerance',1e-
3,'MaxFunctionEvaluation',5000,'MaxIterations',5000,'StepTolerance',1e-
9,'display','iter');
options=optimoptions(options,'display','iter');
%options=optimoptions(options,'Algorithm','sqp');
%options=optimoptions('ga');
%options=optimoptions(options,'InitialPopulationMatrix',z0','ConstraintTo
lerance',1e-2,'MaxTime',600,'display','iter');


%[zopt,Jopt,exitflag,population,scores]=ga(Coste,1+size(Utini,1),Az,bz,[]
,[],zmin',zmax',Restr,options);
[zopt,Jopt,exitflag]=fmincon(Coste,z0,Az,bz,[],[],zmin,zmax,Restr,options
);


Hxopt=zopt(1)';
Utopt=zopt(2:end)';


[HxtRTO,HtRTO,QtRTO]=Prediccion_nuevaRed(r,Hxini,Utini,Dt);
```

1.5. Script 'CalculoCostes.m'

```
function J = CalculoCostes(r,Mdl,Hx0,Ut,Dt)


%Se define un valor máximo del coste si no se encuentra solucion factible
Jmax=1e4;
Hx0=Hx0';
Ut=Ut';
[Hxt,Ht,Qt]=Prediccion(r,Hx0,Ut,Dt);


pr=[];
```

```
%Definición del coste económico

for i=1:24
pr(i)=1+0.3*sin(2*pi*(i-1)/48);
end

if isempty(Hxt)
    J=Jmax;
else
    N=size(Dt,2);
    J=0;
    for k=1:N

        %Coste asociado a la bomba

        %Jbk=(1/1000)*Qt(9,k)*(Ht(2,k)-Ht(1,k))/0.8;
        Jbk=10*pr(k)*Ut(1+k*3)^2;

        %Coste asociado a los niveles de seguridad del tanque
        %Se penaliza al cuadrado con respecto a la distancia (por debajo)
        %al nivel mínimo

        Jtsk1=1/10*(Mdl.Hxnsmin(1)-min(Mdl.Hxnsmin(1),Hxt(1,k)))^2;
        Jtsk2=1/10*(max(Mdl.Hxnsmax(1),Hxt(1,k))-Mdl.Hxnsmax(1))^2;
        J=J+Jbk+Jvk+Jtsk1+Jtsk2;
    end

    % Coste de restricción blanda de periodicidad

    Jprk=1e4*(Hxt(end)-Hx0)^2;
    %Jprk=0;
    J=J+Jprk;
end
```

1.6. Script 'Restricciones.m'

```
function [Out1,Out2]=Restricciones(r,Mdl,Hx0,Ut,D)

%Numero total de restricciones

N=size(D,2);
Ut=Ut';

% ga pasa el vector de valores iniciales como fila, mientras que fmincon lo hace como columna
% La comprobación que viene a continuación asegura que el valor que se pasa a Prediccion tiene el formato correcto
% para Hx0 y Ut

if size(Hx0,1)==1
    Hx0=Hx0';
    Ut=Ut';
end

[Hxt,Ht,Qt]=Prediccion(r,Hx0,Ut,D);
Rk=[];

if isempty(Hxt)

    %Los resultados deberan ser siempre negativos para que se cumplan las
    %restricciones de desigualdad y cero para que se cumplan las de
    %igualdad.

    %Si no se ha obtenido solucion en la llamada a prediccion
    %para algun punto, se asigna un valor de 1000 a cada una de las
    %restricciones para asegurar que estas no se cumplen.

    RD=[];
    RI=[];
    Out1=RD;
```

```
    Out2=RI;

else

    RD=[];
    RI=[];

    for k=1:N+1

        %Nivel maximo del tanque
        Rk=[Rk;Hxt(:,k)-Mdl.Hxmax'];

        %Nivel minimo del tanque
        Rk=[Rk;Mdl.Hxmin'-Hxt(:,k)];

        %Caudal que sale del estanque mayor que cero
        Rk=[Rk;-Qt(1,k)];

        %Alturas de los nodos con demandas mayores que la altura minima
         Rk=[Rk;-Ht(1,k)];
         Rk=[Rk;-Ht(2,k)];
         Rk=[Rk;-Ht(3,k)];
         Rk=[Rk;-Ht(4,k)];
         Rk=[Rk;-Ht(5,k)];
         Rk=[Rk;-Ht(6,k)];
         Rk=[Rk;-Ht(7,k)];
         Rk=[Rk;-Ht(8,k)];

        %Alturas de los nodos con demandas menores que la altura maxima
         Rk=[Rk;Ht(2,k)-Mdl.Hmax];
         Rk=[Rk;Ht(3,k)-Mdl.Hmax];
         Rk=[Rk;Ht(4,k)-Mdl.Hmax];
         Rk=[Rk;Ht(5,k)-Mdl.Hmax];
         Rk=[Rk;Ht(6,k)-Mdl.Hmax];
         Rk=[Rk;Ht(7,k)-Mdl.Hmax];
```

```
    end

    %Restricciones de igualdad

    %Restricciones de periodicidad puestas como restricciones de igualdad

    %RI=Hxt(:,1)-Hxt(:,N+1);
    %RI=[RI;Ht(:,1)-Ht(:,N+1)];
    %RI=[RI;Qt(:,1)-Qt(:,N+1)];

    RD=[RD;Rk];
    Out1=RD;
    Out2=RI;
end
```

1.7. Script 'Casadi.m'

```
%Inicialización Casadi

clear
addpath('C:\Program Files\MATLAB\R2018a\casadi-matlabR2016a-v3.5.5')
import casadi.*

%Inicialización del modelo

ParametrosModelo;

N=24;
for i=1:24
    Dt(1,i)=0;
    Dt(2,i)=-3*sin(2*pi*(i-1)/11.5);
    Dt(3,i)=0;
    Dt(4,i)=2*sin(2*pi*(i-1)/23);
    Dt(5,i)=0;
    Dt(6,i)=1*sin(2*pi*(i-1)/11.5);
```

```
    Dt(7,i)=0;
    Dt(8,i)=0;
end

%Dn=repmat([0 10 5 40 5 20 0 0]',1,24); %Demanda original
Dn=repmat([0 10 5 30 5 20 10 10]',1,24); %Demanda alternativa
Dt=Dt+Dn;
%Dt=Dn;
for i=1:24
precio(i)=1+0.3*sin(2*pi*(i-1)/48);
end
hr=0.5;

% Definición variables Casadi

u=SX.sym('u',3);
d=SX.sym('d',8);
q=SX.sym('q',11);
hn=SX.sym('hn',8);
hx=SX.sym('hx',1);
pr=SX.sym('pr',1);

%Definición del DAE

G1=[0;0;0;0;0;0;0;0;-55*u(1)^2;0;0];
G2=[0;0;0;0;0;0;0;0;0;0;0].*q;
G3=[0;0;0;0;0;0;0;0;5.8e-4;0;0].*q.^2;
G4=[1.418e-6;0.001873;0.001873;0.0285;0.00615;0.006153;0.04277;7.088e-
4;0;u(2);u(3)].*q.*abs(q);
G=G1+G2+G3+G4;

f=B*q; % Ecuaciones diferenciales
g=[A*q-d;A'*hn+G+Ar'*hr+At'*hx]; %Ecuaciones algebraicas

h=pr^2*1/(160000)*q(9)^2*(hn(2)-hn(1))^2/0.8; % Coste asociado a la bomba
%h=10*pr^2*u(1)^2; % Coste asociado a la bomba
```

```
h=h+1*(1/(1e-4+(hx(1)-Mdl.Hxmin(1)))); % Coste asociado al nivel mínimo del tanque

h=h+1/10*(Mdl.Hxnsmin(1)-min(Mdl.Hxnsmin(1),hx(1)))^2;     %%     Costes asociados a niveles mínimos de seguridad del tanque
h=h+1/10*(max(Mdl.Hxnsmax(1),hx(1))-Mdl.Hxnsmax(1))^2;     %%     Costes asociados a niveles máximos de seguridad del tanque

%x=variables diferenciales, p=parámetros/entradas, z=variables algebraicas
%g=ecuaciones algebraicas, f=ecuaciones diferenciales, h=coste de etapa

dae=struct('x',hx,'p',[u;d;pr],'z',[q;hn],'ode',f,'alg',g,'quad',h);

%Definición del integrador

%op=struct('t0',0,'tf',1,'max_num_steps',100000,'abstol',1e-4,'reltol',1e-2);  % Opciones idas
op=struct('t0',0,'tf',1,'collocation_scheme','radau');     %     Opciones collocation
F=integrator('F','collocation',dae,op);

%Definición del NLP y variables de decisión

w={}; lbw=[]; ubw=[]; % El vector w va a contener los niveles de los tanques y las acciones de control a lo largo del período del RTO
Gr={}; J=0; % Gr es el vector de restricciones y J el coste total en el período

%Valores iniciales de los estados (niveles de los tanques)

Xk=MX.sym('X0',1);
w{end+1}=Xk;
lbw=[lbw;40];
ubw=[ubw;60];
ubg=[];
lbg=[];
```

```
x0=[repmat([42.9,1,0.01,0.01],1,24),42.9]; %Para la demanda alternativa
%x0=[41.3891, 0.945862, 0.0163354, 0.001, 41.3888, 0.946828, 0.0262424,
0.001, 41.3886, 0.947567, 0.0263298, 0.001, 41.3878, 0.947887, 0.0256765,
0.001, 41.3852, 0.947733, 0.0251462, 0.001, 41.3796, 0.947203, 0.0248159,
0.001, 41.3707, 0.946488, 0.0247093, 0.001, 41.3586, 0.945781, 0.0248392,
0.001, 41.3445, 0.94523, 0.0252088, 0.001, 41.3304, 0.944929, 0.0258106,
0.001, 41.3185, 0.944939, 0.0266242, 0.001, 41.3109, 0.945267, 0.027611,
0.001, 41.3092, 0.945715, 0.0287062, 0.001, 41.3134, 0.946152, 0.0298231,
0.001, 41.3229, 0.946378, 0.0308552, 0.001, 41.3361, 0.946227, 0.0316918,
0.001, 41.3508, 0.945697, 0.0322408, 0.001, 41.3646, 0.944961, 0.0324473,
0.001, 41.3758, 0.94427, 0.0323011, 0.001, 41.3836, 0.943845, 0.031834,
0.001, 41.3879, 0.943811, 0.0311096, 0.001, 41.3896, 0.944187, 0.030224,
0.001, 41.3897, 0.944907, 0.0295125, 0.001, 41.3893, 0.945856, 0.033522,
0.001, 41.3891];


for k=1:N


    % Variables de control en cada instante k del período del RTO


    U=['U' num2str(k-1)];
    Uk=MX.sym(U,3);
    w{end+1}=Uk;
    if k==1
        Ukant=Uk;
        Uini=Uk;
    end
    % Valores mínimos y máximos de las acciones de control


    lbw=[lbw;u1min;umin;umin];
    ubw=[ubw;u1max;umax;umax];


    if k>1
        Gr{end+1}=vertcat(Uk(1)-Ukant(1)); % Restricción lineal en el
cambio de u(1)
        ubg=[ubg;0.06];
        lbg=[lbg;-0.06];
         %Gr{end+1}=vertcat(Uk(2)/Ukant(2),Uk(3)/Ukant(3));
%Restricciones en la variación de la apertura de las válvulas
         %ubg=[ubg;10;10];
         %lbg=[lbg;0.1;0.1];
```

```
        Ukant=Uk;
    end

    % Se integra para el estado, acciones de control y demandas en el
    % instante k
    Fk=F('x0',Xk,'p',[Uk;Dt(:,k);precio(k)],'z0',[79.0609         46.1043
22.9567    10.2531    -1.1043    -8.6427     7.7036     0.9391    79.0609
40.0000   -0.9391 0.4912   51.8658   47.8886   31.8982   50.8797   47.8811
48.3406   48.3494]');
    %Fk=F('x0',Xk,'p',[Uk;Dt(:,k)],'z0',[79.0609     46.1043     22.9567
10.2531   -1.1043    -8.6427     7.7036     0.9391    79.0609    40.0000   -
0.9391 0.4912   51.8658   47.8886   31.8982   50.8797   47.8811   48.3406
48.3494]');
    if k==1
        zini=Fk.zf;
    end
    % Se suma el coste de etapa al coste total

    J=J+Fk.qf;

    % Restricciones desigualdad no lineales

    Gr{end+1}=vertcat(Fk.xf(1)-Mdl.Hxmax(1),Mdl.Hxmin(1)-Fk.xf(1),-
Fk.zf(1),Fk.zf(13)-Mdl.Hmax,Fk.zf(14)-Mdl.Hmax,Fk.zf(15)-
Mdl.Hmax,Fk.zf(16)-Mdl.Hmax,Fk.zf(17)-Mdl.Hmax,Mdl.Hmin-
Fk.zf(12),Mdl.Hmin-Fk.zf(13),Mdl.Hmin-Fk.zf(14),Mdl.Hmin-
Fk.zf(15),Mdl.Hmin-Fk.zf(16),Mdl.Hmin-Fk.zf(17),Mdl.Hmin-
Fk.zf(18),Mdl.Hmin-Fk.zf(19));
    ubg=[ubg;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0];
    lbg=[lbg;-inf;-inf;-inf;-inf;-inf;-inf;-inf;-inf;-inf;-inf;-inf;-
inf;-inf;-inf;-inf;-inf];

    % Se actualiza el estado

    X=['X' num2str(k)];
    Xk=MX.sym(X,1);
    w{end+1}=Xk;
    lbw=[lbw;40];
    ubw=[ubw;60];
```

```
    % Restricción de continuidad: el nuevo estado deberá ser igual al que
    % se ha obtenido del integrador

    Gr{end+1}=Fk.xf-Xk;
    ubg=[ubg;0];
    lbg=[lbg;0];
end

% Restricciones suaves de periodicidad en el nivel de los tanques
 Gr{end+1}=vertcat(Xk-w{1},Uk-Uini);
 ubg=[ubg;0;1e-2;1e-3;1e-3];
 lbg=[lbg;0;-1e-2;-1e-3;-1e-3];

% Se forma el NLP y se asigna el solver
 options.ipopt.max_iter=10000; %Numero maximo de iteraciones
 options.ipopt.fixed_variable_treatment='make_constraint';
 options.ipopt.tol=1e-5;
 options.ipopt.acceptable_tol=1e-3;
 options.ipopt.linear_solver='mumps';

% Se resuelve el problema y se obtienen las acciones de control y niveles
% del tanque óptimos a partir de una solución inicial

nlp=struct('f',J,'g',vertcat(Gr{:}),'x',vertcat(w{:})); % Se genera el
problema no lineal
%nlp=struct('f',J,'x',vertcat(w{:})); % Se genera el problema no lineal
sin restricciones
S=nlpsol('S','ipopt',nlp); %Se genera el solver para el problema no lineal

%Se resuelve el problema no lineal

r=S('lbx',lbw,'ubx',ubw,'x0',x0,'lbg',lbg,'ubg',ubg);
%r=S('lbx',lbw,'ubx',ubw,'x0',x0); Por si se quiere resolver el problema
sin restricciones
disp(r.x);
```

1.8. Script 'SSTO_MPC.m'

```
    function [xrt_MPC,urt_MPC]= MPC_nuevaRed(Hxopt,Qopt,Mdl,Hxini,D,k)

% A tiene dimensiones del numero de tanques

A=1;

% B es la misma que la del modelo no lineal

B=Mdl.B;

% Bd tiene dimensiones del numero de tanques por la dimension
% del vector de demandas. Como no hay ninguna demanda asociada directamente
% a los tanques, se deja a cero

Bd=[0 0 0 0 0 0 0 0];

%Ae tiene dimensiones de demandas por el numero de tanques.

Ae=zeros(8,1);

%E serán las ecuaciones de balance de masas en cada uno de los nudos libres

E=Mdl.A;

%Ed es la matriz de términos correspondientes a las demandas en los nudos
%libres

Ed=[-1 0 0 0 0 0 0 0;0 -1 0 0 0 0 0 0;0 0 -1 0 0 0 0 0;0 0 0 -1 0 0 0 0;0
0 0 0 -1 0 0 0;
    0 0 0 0 0 -1 0 0;0 0 0 0 0 0 -1 0;0 0 0 0 0 0 0 -1];

% %Parametros del MPC

 Q=1e3;
```

```
 R=eye(11);

 [~,P,~]=dlqr(A,B,Q,R); % Se obtiene solo la matriz P del dlqr


%% Calculo del punto de equilibrio (SSTO)

%Se escogen como minimos las alturas minimas de los tanques y -120 lps
para
%los caudales (excepto para el estanque, que se escoge cero) y como maximos
%las alturas maximas de los tanques y 120 lps para todos los caudales

zmin1=[Mdl.Hxmin';0;-120*ones(size(Mdl.A,2)-1,1)];
zmax1=[Mdl.Hxmax';120*ones(size(Mdl.A,2),1)];

%Se asignan los valores obtenidos a los vectores xr y ur

xr=Hxopt;

ur=Qopt;

xr=[xr(:,k:end),xr(:,1:k-1)]; %Se actualiza la trayectoria del RTO
dependiendo del punto actual

ur=[ur(:,k:end),ur(:,1:k-1)];

D=[D(:,k:end),D(:,1:k-1)];


%% SSTO

[nx,nu]=size(B); %%B tiene nx filas (numero de estados) y nu columnas
(numero de caudales)

N=size(Qopt,2);
```

```
nz=N*nx+N*nu;
neq=size(Ae,1);

% Se obtienen ahora las matrices H y F y los vectores h y f de la funcion
% de coste y las restricciones, respectivamente, de la trayectoria del RTO
% y de los parámetros R y Q que se han definido

H=[];

for j=1:N
    H=blkdiag(H,2*Q,2*R);
end
H=blkdiag(H,2*P);

h=[];
for j=1:N
    h=[h;-2*Q*xr(:,j);-2*R*ur(:,j)];
end
h=[h;-2*P*xr(:,N+1)];

%Formacion de la matriz F de restricciones

F=zeros(N*nx+N*neq,nz);
ii=0;
ij=0;

for j=1:N
    F(ii+(1:nx),ij+(1:(2*nx+nu)))=[A,B,-eye(nx)];
    ii=ii+nx;
    ij=ij+(nx+nu);
end
if neq>0
    ij=0;
    for j=1:N
        F(ii+(1:neq),ij+(1:(nx+nu)))=[Ae,E];
```

```
        ii=ii+neq;
        ij=ij+(nx+nu);
    end
end

% Vector columna f

f=[];
for j=1:N
    f=[f;-Bd*D(:,j)];
end
for j=1:N
    f=[f;-Ed*D(:,j)];
end



Ft=zeros(nx,nz);

%Se incluyen las restricciones terminales de igualdad entre nivel inicial
y
%final

Ft(:,1:nx)=eye(nx);
Ft(:,N*(nx+nu)+(1:nx))=-eye(nx);

ft=zeros(nx,1);

zmin=[];
zmax=[];

for j=1:N
    zmin=[zmin;zmin1];
    zmax=[zmax;zmax1];
end

%Se les añade al final un término más correspondiente a las restricciones
```

```
%terminales de igualdad

zmin=[zmin;zmin1(1)];
zmax=[zmax;zmax1(1)];
options1=optimset('quadprog');
options1=optimset(options1,'Display','off');

zSSTO=quadprog(H,h,[],[],[F;Ft],[f;ft],zmin,zmax,[],options1);

for j=1:N
    xrt_SSTO(:,j)=zSSTO((j-1)*(nx+nu)+(1:nx));
    urt_SSTO(:,j)=zSSTO((j-1)*(nx+nu)+nx+(1:nu));
end

xrt_SSTO(:,N+1)=zSSTO(N*(nx+nu)+(1:nx));


%% MPC

%Para el MPC se realiza el mismo procedimiento que para el SSTO
%calculando las matrices H y F pero ahora utilizando la trayectoria
%que proporciona el SSTO

Np=8; %Horizonte de prediccion
Q=100;
R=eye(11);
[~,P,~]=dlqr(A,B,Q,R); % Se obtiene solo la matriz P del dlqr
nz=Np*nx+Np*nu;
H=[];

for j=1:Np
    H=blkdiag(H,2*Q,2*R);
end
H=blkdiag(H,2*P);
```

```
h=[];
for j=1:Np
    h=[h;-2*Q*xrt_SSTO(:,j);-2*R*urt_SSTO(:,j)];
end
h=[h;-2*P*xrt_SSTO(:,Np+1)];
F=[];
F=zeros((Np+1)*nx+Np*neq,nz);
F(1:nx,1:nx)=eye(nx);
ii=nx;
ij=0;

for j=1:Np
    F(ii+(1:nx),ij+(1:(2*nx+nu)))=[A,B,-eye(nx)];
    ii=ii+nx;
    ij=ij+(nx+nu);
end
if neq>0
    ij=0;
    for j=1:Np
        F(ii+(1:neq),ij+(1:(nx+nu)))=[Ae,E];
        ii=ii+neq;
        ij=ij+(nx+nu);
    end
end

%Restricción terminal: nivel final igual al del SSTO

Ft=zeros(nx,nz);
Ft(:,Np*(nx+nu)+(1:nx))=eye(nx);

f=[Hxini];
for j=1:Np
    f=[f;-Bd*D(:,j)];
end
for j=1:Np
    f=[f;-Ed*D(:,j)];
```

```
end

ft=xrt_SSTO(:,Np+1);

zmin=[];
zmax=[];

for j=1:Np
    zmin=[zmin;zmin1];
    zmax=[zmax;zmax1];
end

zmin=[zmin;zmin1(1)];
zmax=[zmax;zmax1(1)];

zMPC=quadprog(H,h,[],[],[F;Ft],[f;ft],zmin,zmax,[],options1);

for j=1:Np %Np es el horizonte de prediccion del MPC
    xrt_MPC(:,j)=zMPC((j-1)*(nx+nu)+(1:nx));
    urt_MPC(:,j)=zMPC((j-1)*(nx+nu)+nx+(1:nu));
end
xrt_MPC=[xrt_MPC,zMPC(end)];
```

1.9. Script 'Simulacion.m'

```
%% Soluciones Casadi

%Se ejecuta esta sección si se quieren tomar las soluciones de Casadi

%% Generación vectores entradas
sol=r.x;
sol=full(sol);
u1=sol(2:4:end);
u2=sol(3:4:end);
u3=sol(4:4:end);
```

```
u=[];
for i=1:size(u1,1)
    u=[u;u1(i);u2(i);u3(i)];
end

%% Integrador en todo el intervalo de tiempo N

zetas=[];
for i=1:N
    Xk=sol(1+(i-1)*4);
    Uk=sol(2+(i-1)*4:4+(i-1)*4);
    if i==1
        Fk=F('x0',Xk,'p',[Uk;Dt(:,i);precio(i)],'z0',[79.3904,    46.299,
23.0914, 10.3052, -1.29898, -8.39581, 7.78617, 0.609641, 79.3904, 40, -
0.609641, 0.491063, 92.2854, 87.2705, 71.7705, 88.7867, 89.2601, 102.694,
16.9997]');
    else
        Fk=F('x0',Xk,'p',[Uk;Dt(:,i);precio(i)],'z0',Fk.zf);
    end
    zetas=[zetas;Fk.zf'];
end

zetas=full(zetas');

HxtRTO=sol(1:4:end)';
HtRTO=zetas(12:end,:);
QtRTO=zetas(1:11,:);

%% Control de la red

t=0;
kant=0;
%Inicializacion PI

deltauKi=0;
deltauKi2=0;
```

```
deltauKi3=0;
Tint=1/12; %Tiempo de integracion de 5 minutos
u1=u01;
u2=u02;
u3=u03;
uf1=u1;
uf2=u2;
uf3=u3;
refQ9t=[];
refQ11t=[];
refH4t=[];
Hxt_MPC=[];
%Paramatros PI bomba

Kp1=1e-3;
Ti1=10;

% %Parametros PI valvula

Kp2=-1e-3;
Ti2=5;

Kp3=-0.015/5;
Ti3=10;

%Se simulan 120 horas - las 12 horas del horizonte de prediccion del MPC
Hxt=[];
Ht=[];
Qt=[];
u_t=[];

Nsim=(120-24*Mdl.Tm)/Tint;
Hxini=[43];
for k=0:Nsim
```

```
    if k==0

        [Hk,Qk]=Simul_EPANET_nuevaRed(r,1,Hxini,[u01                    u02
u03],Dt(:,1+floor(rem(k*Tint,N))));
        Hk=Hk(1,:)';
        Qk=Qk(1,:)';

    else

        %Se vuelven a calcular las alturas y caudales con las nuevas
        %entradas

        [Hk,Qk]=Simul_EPANET_nuevaRed(r,300,Hk(10),[uf1                 uf2
uf3],Dt(:,1+floor(rem(k*Tint,N))));
        Hk=Hk(end,:)';
        Qk=Qk(end,:)';
    end

    %El MPC se ejecuta dentro del bucle pero solo cuando el indice es un
    %multiplo de Tm (1 vez cada hora en este caso)

    if (mod(k*Tint,Tm)==0)

        [xt,ut]=
MPC_nuevaRed(HxtRTO,QtRTO,Mdl,Hk(end),Dt(1:8,:),1+floor(rem(k*Tint,N)));

        %Se actualizan las referencias de los caudales
        Hx_MPC=xt(1);
        refQ9=ut(9,1);
        refH4=35;
        refQ11=ut(11,1);

    end

        %Entrada 1
```

```
%Termino proporcional

deltauKp=Kp1*(refQ9-Qk(9));

%Termino integral con anti windup, si la entrada esta saturada, el
%termino integral se mantiene constante

if u1==Mdl.u1max
    if (deltauKi+Tint*(refQ9-Qk(9))*Kp1/Ti1 > 0)
        u1=u1+deltauKp;
    else
        deltauKi=deltauKi+Tint*(refQ9-Qk(9));
        u1=u1+deltauKp+deltauKi*Kp1/Ti1;
    end

elseif u1==Mdl.u1min
    if (deltauKi+Tint*(refQ9-Qk(9))*Kp1/Ti1 < 0)
        u1=u1+deltauKp;
    else
        deltauKi=deltauKi+Tint*(refQ9-Qk(9));
        u1=u1+deltauKp+deltauKi*Kp1/Ti1;
    end
else
    deltauKi=deltauKi+Tint*(refQ9-Qk(9));
    u1=u1+deltauKp+deltauKi*Kp1/Ti1;
end

%Filtrado de la señal de control

uf1=u1+Tint/5*(u1-uf1);

if uf1<Mdl.u1min
    uf1=Mdl.u1min;
end
```

```
if uf1>Mdl.u1max
    uf1=Mdl.u1max;
end


%Entrada 2

%Termino proporcional

deltauKp2=Kp2*(refH4-Hk(4));

%Termino integral con anti windup, si la entrada esta saturada, el
%termino integral se mantiene constante

if u2==Mdl.umax
    if (deltauKi2+Tint*(refH4-Hk(4))*Kp2/Ti2 > 0)
        u2=u2+deltauKp2;
    else
        deltauKi2=deltauKi2+Tint*(refH4-Hk(4));
        u2=u2+deltauKp2+deltauKi2*Kp2/Ti2;
    end

elseif u2==Mdl.umin
    if (deltauKi2+Tint*(refH4-Hk(4))*Kp2/Ti2 < 0)
        u2=u2+deltauKp2;
    else
        deltauKi2=deltauKi2+Tint*(refH4-Hk(4));
        u2=u2+deltauKp2+deltauKi2*Kp2/Ti2;
    end
else
    deltauKi2=deltauKi2+Tint*(refH4-Hk(4));
    u2=u2+deltauKp2+deltauKi2*Kp2/Ti2;
end

%Filtrado de la señal de control
```

```
uf2=u2+Tint/5*(u2-uf2);

if uf2<Mdl.umin
    uf2=Mdl.umin;
end

if uf2>Mdl.umax
    uf2=Mdl.umax;
end

%Entrada 3

%Termino proporcional

deltauKp3=Kp3*(refQ11-Qk(11));

%Termino integral con anti windup, si la entrada esta saturada, el
%termino integral se mantiene constante

if u3==Mdl.umax
    if (deltauKi3+Tint*(refQ11-Qk(11))*Kp3/Ti3 > 0)
        u3=u3+deltauKp3;
    else
        deltauKi3=deltauKi3+Tint*(refQ11-Qk(11));
        u3=u3+deltauKp3+deltauKi3*Kp3/Ti3;
    end

elseif u3==Mdl.umin
    if (deltauKi3+Tint*(refQ11-Qk(11))*Kp3/Ti3 < 0)
        u3=u3+deltauKp3;
    else
        deltauKi3=deltauKi3+Tint*(refQ11-Qk(11));
        u3=u3+deltauKp3+deltauKi3*Kp3/Ti3;
    end
else
```

```
            deltauKi3=deltauKi3+Tint*(refQ11-Qk(11));
            u3=u3+deltauKp3+deltauKi3*Kp3/Ti3;
        end

        %Filtrado de la señal de control

        uf3=u3+Tint/5*(u3-uf3);

        if uf3<Mdl.umin
            uf3=Mdl.umin;
        end

        if uf3>Mdl.umax
            uf3=Mdl.umax;
        end

    %Se guardan las variables en cada instante para su posterior
        %representacion
        refQ9t=[refQ9t refQ9];
        refQ11t=[refQ11t refQ11];
        refH4t=[refH4t refH4];
        Hxt_MPC=[Hxt_MPC Hx_MPC];
        Hxt=[Hxt Hk(end)];
        Ht=[Ht Hk(1:end-2)];
        Qt=[Qt Qk];
        u_t=[u_t [uf1;uf2;uf3]];
        t=[t;t(end)+Tint*(k-kant)];
        kant=k;

end

%% Representacion

figure();plot(t(2:end),Qt','LineWidth',1);legend({'Q_1','Q_2','Q_3','Q_4'
,'Q_5','Q_6','Q_7','Q_8','Q_9','Q_{10}','Q_{11}'});xlabel('Tiempo
(h)');ylabel('Caudal             (l/s)');grid;set(gca,'GridLineStyle','--
```

```
');title('Caudales');

figure();plot(t(2:end),Ht','LineWidth',1);legend({'H_1','H_2','H_3','H_4'
,'H_5','H_6','H_7','H_8'});xlabel('Tiempo           (h)');ylabel('Presión
(mca)');grid;set(gca,'GridLineStyle','--');title('Alturas');

figure();plot(t(2:end),Hxt,'LineWidth',1);xlabel('Tiempo
(h)');ylabel('Altura               (m)');grid;set(gca,'GridLineStyle','--
');title('Altura del depósito');

figure();plot(0:1:96,[repmat(QtRTO(9,:),1,4),QtRTO(9,1)],'k','LineWidth',
1);hold on;plot(t(2:12:end),refQ9t(1:12:end),'--*','color',[0.3010 0.4450
0.9330]);hold                on;plot(t(2:end),Qt(9,:),'r');xlabel('Tiempo
(h)');ylabel('Caudal             (l/s)');grid;set(gca,'GridLineStyle','--
');title('Caudal        Q_9');legend({'Referencia        RTO','Referencia
MPC','Simulación'});

figure();plot(0:1:96,[repmat(QtRTO(11,:),1,4),QtRTO(11,1)],'k','LineWidth
',1);hold     on;plot(t(2:12:end),refQ11t(1:12:end),'--*','color',[0.3010
0.4450    0.9330]);hold    on;plot(t(2:end),Qt(11,:),'r');xlabel('Tiempo
(h)');ylabel('Caudal             (l/s)');grid;set(gca,'GridLineStyle','--
');title('Caudal      Q_{11}');legend({'Referencia      RTO','Referencia
MPC','Simulación'})

figure();plot(t(2:12:end),refH4t(1:12:end),'--*','color',[0.3010   0.4450
0.9330]);hold                on;plot(t(2:end),Ht(4,:),'r');xlabel('Tiempo
(h)');ylabel('Altura               (m)');grid;set(gca,'GridLineStyle','--
');title('Altura H_{4}');legend({'Referencia','Simulación'})

figure();plot(0:1:96,[repmat(HxtRTO(1:24),1,4),HxtRTO(1)],'k','LineWidth'
,1);hold      on;plot(t(2:12:end),Hxt_MPC(1:12:end),'--*','color',[0.3010
0.4450       0.9330]);hold       on;plot(t(2:end),Hxt,'r');xlabel('Tiempo
(h)');ylabel('Altura               (m)');grid;set(gca,'GridLineStyle','--
');title('Altura del depósito (m)');legend({'Referencia RTO','Referencia
MPC','Simulación'})
```