Identificación y Control Adaptativo
Trabajo Práctico 2

Truls Carlson

19 de diciembre de 2024

Índice

1. Enunciado del Problema

Continuamos con el tanque dado del Trabajo Práctico 1. Ahora vamos a incluir la dinámica del sensor y la válvula, entonces excitaremos la planta con entradas diferentes y analizaremos sus respuestas.

1.1. Dinámica de la válvula

La válvula de descarga presenta una dinámica mecánica de primer orden con una constante de tiempo de 10 segundos, como se describió en el TP1.

La función de transferencia de un sistema de primer orden con una constante de tiempo $\hat{\tau}$, se puede escribir como,

$$H(s) = \frac{1}{\hat{\tau}s + 1} \tag{1}$$

Pero, si usamos una función de transferencia de esta manera, suponemos que la condición inicial es igual a 0. Por eso, para iniciar la planta en un estado de equilibrio, utilizamos una representación en el espacio de estados. Se utilizó la función *tf2ss* en Matlab para obtener las siguientes matrices: $A = -\frac{1}{\hat{\tau}}$, $B = 1$, $C = \frac{1}{\hat{\tau}}$, $D = 0$. Entonces expresamos la apertura de válvula $v(t)$ con la entrada k como un sistema de representación en el espacio de estados,

$$\begin{aligned} \dot{v} &= Av + Bk, \quad y = Cv + Dk \\ \dot{v} &= -\frac{1}{\hat{\tau}}v + k, \quad y = \frac{1}{\hat{\tau}}v \end{aligned} \tag{2}$$

con la solución general,

$$\begin{aligned} v(t) &= e^{-\frac{1}{\hat{\tau}}t}v(0) + \int_0^t e^{-\frac{1}{\hat{\tau}}(t-\tau)}k(\tau)d\tau \quad \text{suponiendo } k(\tau) \text{ constante} \\ &\vdots \\ v(t) &= e^{-0,1t}(v_0 - 10k) + 10k, \quad y = 0,1v \end{aligned} \tag{3}$$

Caundo,

$$\lim_{t\to\infty} v(t) = 10k \Rightarrow y = k \tag{4}$$

Luego, para comenzar la planta en equilibrio ponemos $v_0 = 10k$.

1.2. Dinámica del sensor

El sensor de nivel tiene una dinámica mecánica similar a un primer orden con una constante de tiempo de 10 segundos, como se describe en el TP1. Por lo tanto podemos expresarlo igual como la válvula describió en sección 1.1. Pero, en este caso, la entrada sería la altura h. Por eso, la comenzamos en $10h$.

1.3. Implementacíon de las dinámicas

Para simular la dinámica de la válvula y del sensor, usé bloques de *State-Space*, como se muestra en la figura 1.

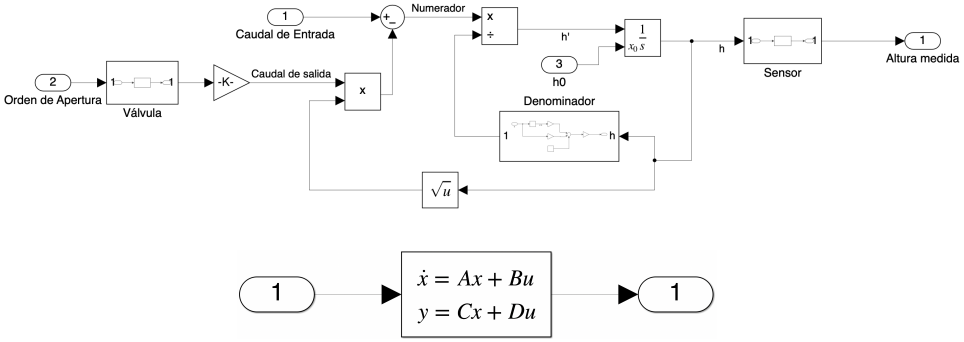

Figura 1: Implementacíon en Simulink

2. Ejercicios

2.1. 1) Excitar a la planta con un impulso y graficar su respuesta impulsional

El impulso de apertura puede representarse matemáticamente como,

$$k(t) = \begin{cases} 1, & \text{si } t = T. \\ k_0, & \text{de lo contrario.} \end{cases} \tag{5}$$

donde k_0 es la apertura de la válvula que mantiene el correspondiente altura h_0.

Dado que Simulink no genera impulsos directamente, se utilizaron dos bloques de escalón combinados.

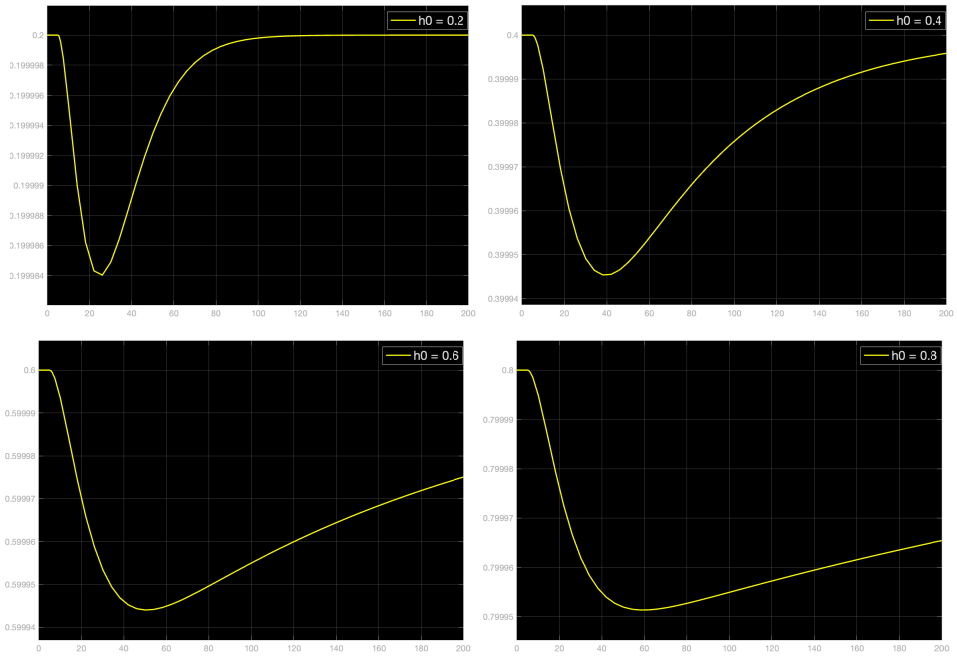

Figura 2: La respuesta impulsiva de la planta a distintos niveles.

Como se puede ver en la figura 2, la respuesta depende de altura inicial. Esto es debido a la dinámica no lineal del tanque conico. En el TP1 obtuve la ecuacion de $\dot{h}$ como,

$$\dot{h} = \frac{F_{in} - k\pi r_o^2\sqrt{2gh}}{\pi\left(\frac{1}{3}h^2 + \frac{2r_0}{\sqrt{3}}h + r_0^2\right)}. \tag{6}$$

Entonces, cuanto mayor sea h, menor será $\dot{h}$.

2.2. 2) Excitar a la planta con un escalón y calcular su respuesta impulsional

El escalón se puede escribir como,

$$k(t) = \begin{cases} 1, & \text{si } t \geq T. \\ k_0, & \text{de lo contrario.} \end{cases} \tag{7}$$

Para simularlo usé un bloque de escalón.

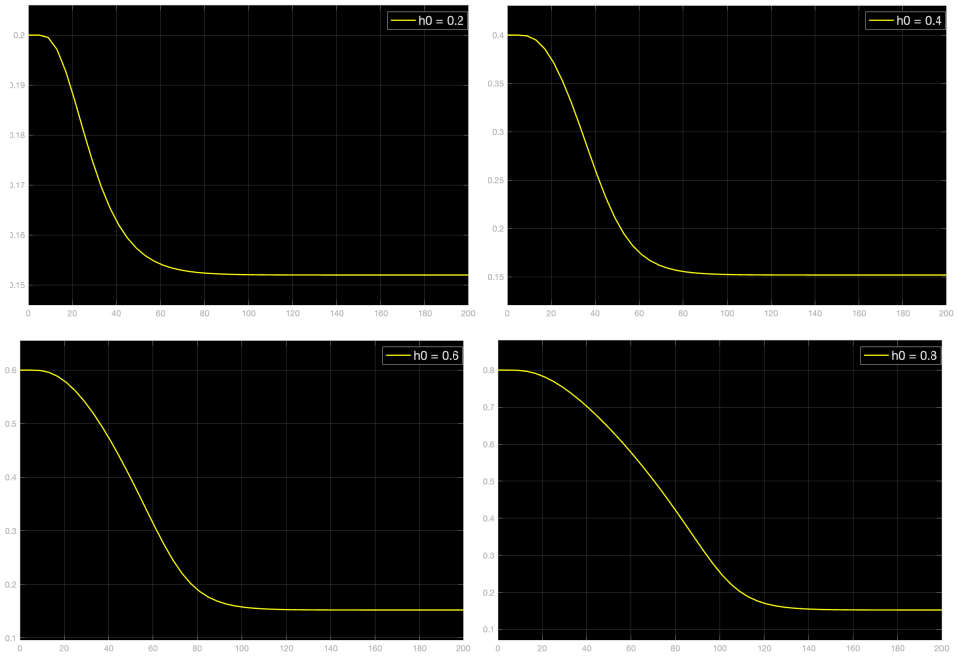

Figura 3: La respuesta de un escalón de la planta a distintos niveles.

Está claro que una altura inicial más alta resultará en más tiempo para descargar el tanque. Con $k = 1$, se puede encontrar el nivel que el tanque se aproxima como,

$$\begin{aligned} F_{in} - F_{out} &= 0 \\ \tilde{h} &= \frac{F_{in}^2}{k^2 \pi^2 r_o^4 2g} \\ &\vdots \\ \tilde{h} &\approx 0{,}152 \end{aligned} \tag{8}$$

2.3. 3) Excitar a la planta con ruido blanco y calcular su respuesta impulsional

Utilicé ruido blanco gaussiano aditivo para simular una entrada de ruido blanco. En simulink, usé el bloque *Random Number* con $\sigma = 0{,}001$ y lo sumé con k_0.

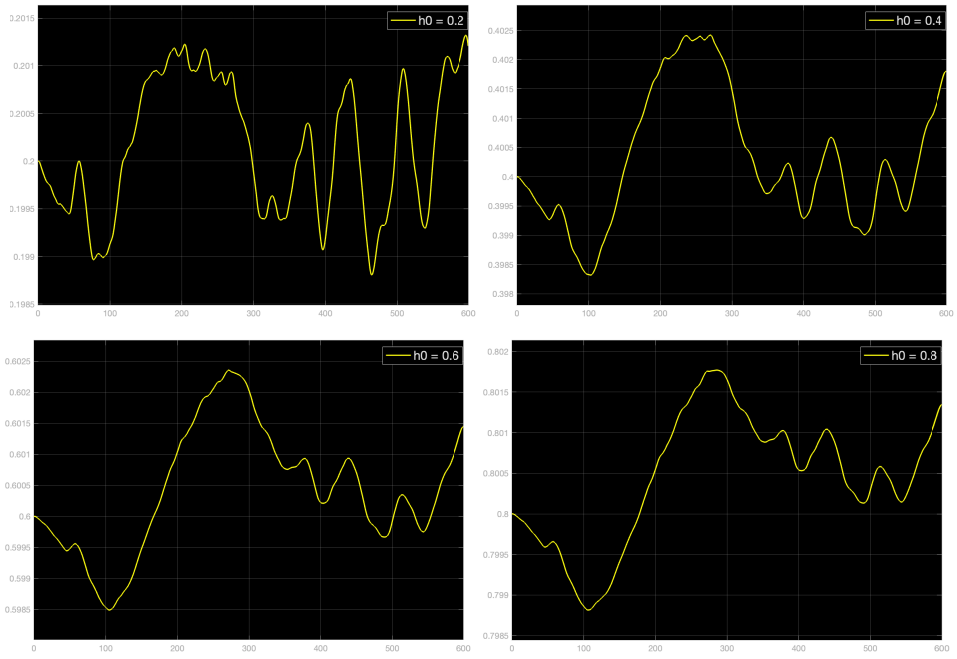

Figura 4: La respuesta de ruido blanco de la planta a distintos niveles.

Como expliqué en 2.1, una altura mayor resultará a una dinámica más lenta. Por eso la planta con la altura inicial de $h_0 = 0{,}2$ se afecta mucho más de una entrada con ruido blanco, que la planta con la altura inicial de $h_0 = 0{,}8$. Se observa claramente en figura 4.

2.4. 4) Excitar a la planta con senoides de diferentes frecuencias y graficar el diagrama de bode

Una senoide puede expresarse en general como,

$$k(t) = Asin(2\pi ft) \tag{9}$$

Elegí una amplitud de $A = 0{,}1$ para mantener $0 < k < 1$.

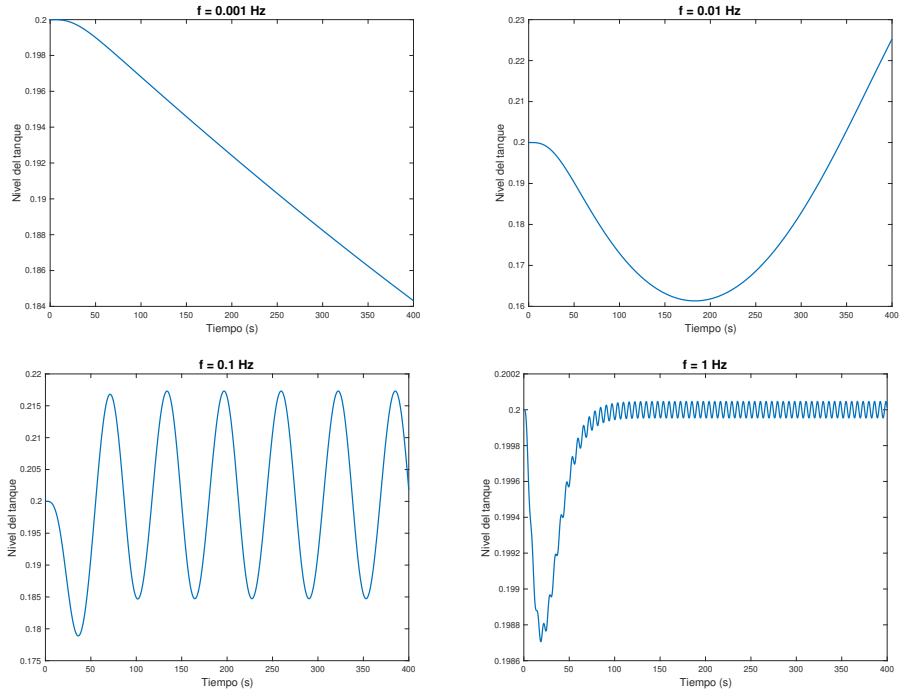

Figura 5: Las respuestas de senoides de diferentes frecuencias de la planta con $h_0 = 0{,}2$.

Para hacer los diagramas de bode, usé la herramienta *Linearization Manager*[1] en Simulink.

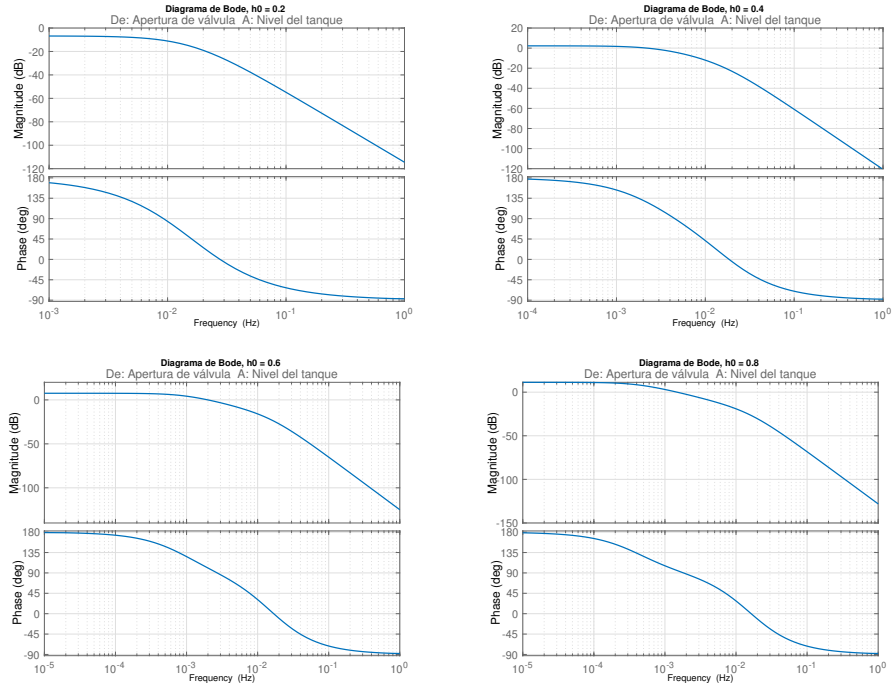

Figura 6: Diagramas de Bode para distintos niveles.

Como se puede ver en la figura 6, el sistema permite el paso de las frecuencias más bajas y atenúa las frecuencias más altas. Por eso, el sistema funciona como un filtro pasa bajo.

2.5. 5) Excitar a la planta con ruido blanco y calcular su respuesta frecuencial

El ancho de banda de ruido blanco, es teóricamente infinito [2]. Por eso, la respuesta frecuencial sería la misma que se describe en la sección 2.4. Un diagrama de Bode es un gráfico de la respuesta frecuencial.

2.6. 6) Implementar una rutina de mínimos cuadrados

El método de mínimos cuadrados se usa para la identificación de sistemas. Conociendo la entrada y la salida de un sistema, es posible estimar sus parámetros mediante este método. Minimizamos una función de costes que describe la diferencia entre las predicciones y los datos reales. Desde [3] se considera la siguiente planta real,

$$y_k = \sum_{i=1}^{n} a_i y_{k-i} + \sum_{i=0}^{m} b_i u_{k-i-1} + \epsilon_k = \boldsymbol{x}_k^T \boldsymbol{\theta}_k + \epsilon_k \tag{10}$$

donde y es el nivel del tanque, u la entrada y ϵ la perturbación o incertidumbre. El modelo de la planta se puede escribir como,

$$\hat{y}_k = \sum_{i=1}^{n} \hat{a}_i y_{k-i} + \sum_{i=0}^{m} \hat{b}_i u_{k-i-1} = \boldsymbol{x}_k^T \hat{\boldsymbol{\theta}}_k \tag{11}$$

donde

$$\hat{\boldsymbol{\theta}}_k = \begin{bmatrix} \hat{a}_1 \\ \vdots \\ \hat{a}_n \\ \hat{b}_0 \\ \vdots \\ \hat{b}_m \end{bmatrix} \quad \boldsymbol{x}_k = \begin{bmatrix} y_{k-1} \\ \vdots \\ y_{k-n} \\ u_{k-1} \\ \vdots \\ u_{k-m-1} \end{bmatrix} \tag{12}$$

Para cada instante k habrá un error o diferencia entre la salida de la planta real y del modelo. Lo escribimos como,

$$e_k = y_k - \hat{y}_k \tag{13}$$

tomando todas las muestras,

$$\boldsymbol{E}_k = \boldsymbol{Y}_k - \boldsymbol{\Phi}_k \hat{\boldsymbol{\theta}}_k \tag{14}$$

con

$$\boldsymbol{E}_k = \begin{bmatrix} e_k \\ \vdots \\ e_1 \end{bmatrix} \quad \boldsymbol{Y}_k = \begin{bmatrix} y_k \\ \vdots \\ y_1 \end{bmatrix} \quad \boldsymbol{\Phi}_k = \begin{bmatrix} \boldsymbol{x}_k^T \\ \vdots \\ \boldsymbol{x}_0^T \end{bmatrix} \tag{15}$$

La función de costes la escribimos como,

$$J_k = \boldsymbol{E}_k^T \boldsymbol{E}_k = \boldsymbol{Y}_k^T \boldsymbol{Y}_k - 2\boldsymbol{Y}_k^T \boldsymbol{\Phi}_k \hat{\boldsymbol{\theta}}_k + \hat{\boldsymbol{\theta}}_k^T \boldsymbol{\Phi}_k^T \boldsymbol{\Phi}_k \hat{\boldsymbol{\theta}}_k \tag{16}$$

Para minimizarla al respecto a $\hat{\boldsymbol{\theta}}_k$ encontramos el mínimo como,

$$\begin{aligned} \frac{\partial J_k}{\partial \hat{\boldsymbol{\theta}}_k} &= 0 \\ -2\boldsymbol{Y}_k^T \boldsymbol{\Phi}_k + 2\boldsymbol{\Phi}_k^T \boldsymbol{\Phi}_k \hat{\boldsymbol{\theta}}_k &= 0 \\ \hat{\boldsymbol{\theta}}_k^* &= [\boldsymbol{\Phi}_k^T \boldsymbol{\Phi}_k]^{-1} \boldsymbol{\Phi}_k^T \boldsymbol{Y}_k \end{aligned} \tag{17}$$

2.6.1. Método directo

Con el método directo, calculamos los parámetros directamente a partir de las matrices construidas $\boldsymbol{\Phi}_k$ y $\boldsymbol{Y}_k$. Usé Matlab para implementar el método directo,

```
y = out.output_simout; % Output data
u = out.input_simout;  % Input data

% Subtract the initial output and input
y = y-y(1);
u = u-u(1);

n = 1; % Output order
m = 1; % Input order

N = length(y); % Number of samples

% Preallocate Phi and Y
Phi = zeros(N-1, n + m);
Y = zeros(N-1, 1);

% Construct Phi and Y matrices
for k = 2:N
    phi_y = y(k-1);
    phi_u = u(k-1);
    Phi(k-1, :) = [phi_y, phi_u];
    Y(k-1) = y(k);
end

theta_hat = (Phi' * Phi) \ (Phi' * Y);
```

Listing 1: Código del método directo.

En la siguiente cuadro se puede ver los parámetros al cambiar el nivel,

Altura h_0 (m)	Parámetros $\hat{\boldsymbol{\theta}}$
0,2	$\begin{bmatrix} 0{,}9983 \\ -0{,}0012 \end{bmatrix}$
0,4	$\begin{bmatrix} 0{,}9995 \\ -0{,}0011 \end{bmatrix}$
0,6	$\begin{bmatrix} 0{,}9997 \\ -0{,}0009 \end{bmatrix}$
0,8	$\begin{bmatrix} 0{,}9998 \\ -0{,}0007 \end{bmatrix}$

Cuadro 1: Parámetros estimados para diferentes niveles.

2.6.2. Método recursivo

Con el método recursivo, actualizamos la estimación de los parámetros en cada iteración. Como describió en [3], se puede encontrar el algoritmo recursivo desde 17,

$$\begin{aligned}\hat{\boldsymbol{\theta}}_k &= \hat{\boldsymbol{\theta}}_{k-1} - \boldsymbol{P}_k \boldsymbol{x}_k [\hat{y}_k - y_k] \\ \boldsymbol{P}_k &= \boldsymbol{P}_{k-1} - \frac{\boldsymbol{P}_{k-1} \boldsymbol{x}_k \boldsymbol{x}_k^T \boldsymbol{P}_{k-1}}{1 + \boldsymbol{x}_k^T \boldsymbol{P}_{k-1} \boldsymbol{x}_k}\end{aligned} \tag{18}$$

Usé Matlab para implementar el algoritmo,

```
% Initialize parameters and P matrix
theta_hat = zeros(n + m, 1);
P = 1e6 * eye(n + m);

for k = 2:N
    x_k = [y(k-1); u(k-1)];

    y_hat = x_k' * theta_hat;

    theta_hat = theta_hat - P * x_k * (y_hat - y(k));

    P = P - (P * (x_k * x_k') * P) / (1 + x_k' * P * x_k);
end
```

Listing 2: Código del método recursivo.

2.6.3. Resultados

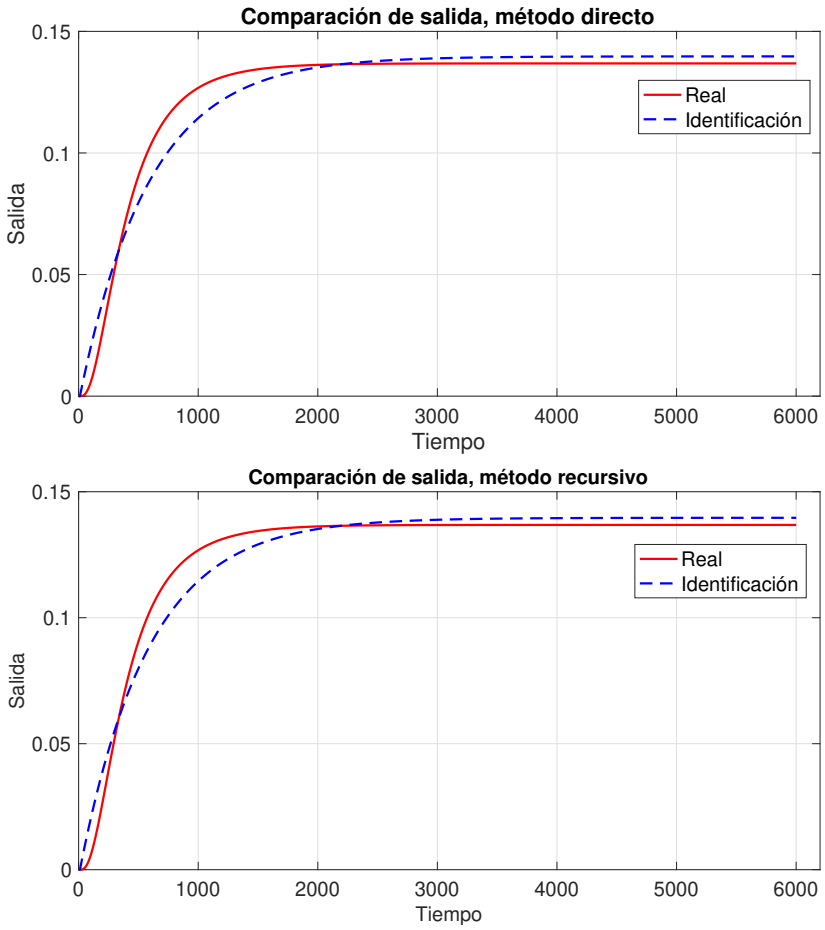

Figura 7: Comparación entre la salida real y la salida estimida con $h_0 = 0{,}8$.

Ambos métodos generaron parámetros idénticos, como se observa en el cuadro 1.

Referencias

[1] Linearization manager. https://la.mathworks.com/help/slcontrol/ug/linearizationmanager-app.html. Accedido: 25-11-2024.

[2] White noise. https://en.wikipedia.org/wiki/White_noise. Accedido: 01-12-2024.

[3] Universidad de Buenos Aires Facultad de Ingeniería. *Identificación por Mínimos Cuadrados*. 2024.