Identificación de sistemas

Josep María Barberá Civera (17048)
Electrónica y Automática

8 de abril de 2022

Sistema modelado

El sistema del que se dispone para la realización de la práctica corresponde a un motor-generador. La señal de excitación del motor es analógica de un valor máximo de 24 voltios de continua (DC). El motor actúa directamente sobre un generador (G) al que se le pueden acoplar distintas cargas mediante la conexión de lámparas incandescentes. La velocidad del motor se mide a través del sensor taco-generatriz (TG) que proporciona una relación de 4.3V/1000 rpm. La posición del motor se puede medir a través del *encoder* incremental de tres canales (A, B, Z). Puede verse una imagen[1] de dicho motor en la figura 1.

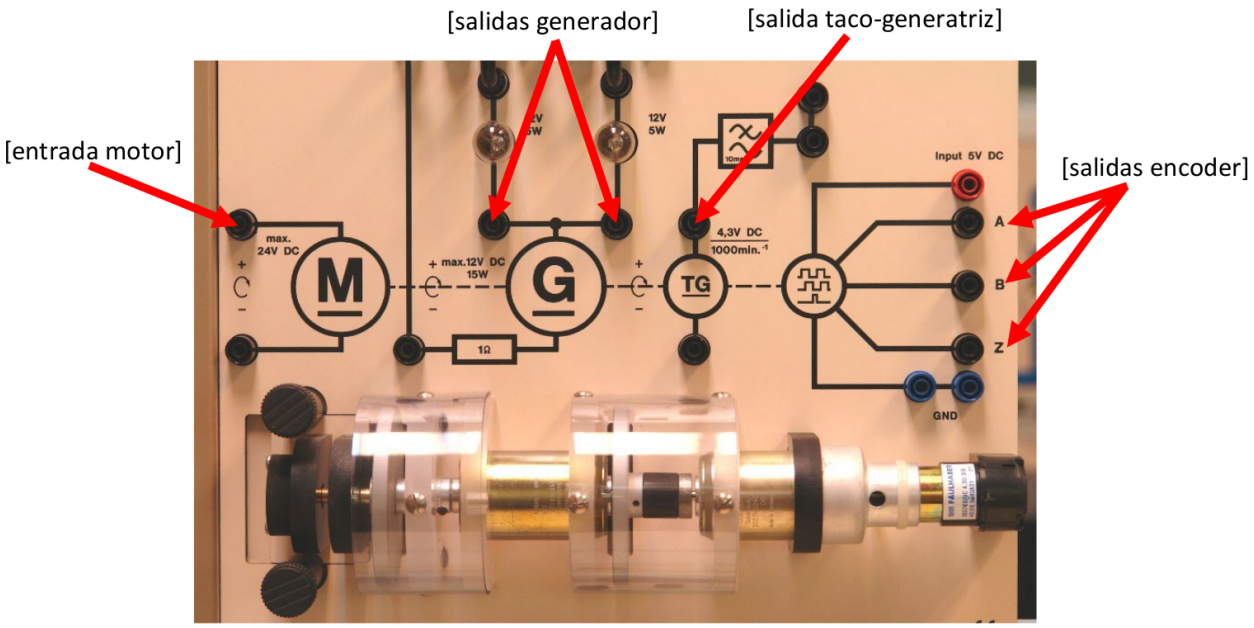

Figura 1: Motor generador. Sistema a modelar mediante una tarjeta de adquisición de datos y el uso de *Simulink*.

[1]Imagen y descripción cortesía del Departamento de Ingeniería de Sistemas de la ETSII-UPM (DISAM).

Obtención del modelo

Mediante la aplicación de Matlab llamada *System Identification* importaremos los datos obtenidos previamente del sistema motor. Puede verse dicha aplicación y su entorno en la figura 2. Básicamente, podemos importar los datos para estimar el modelo, además, podemos realizar un preprocesado de ellos, y seleccionando cómodamente podemos generar gráficos que nos ayuden a comprobar los resultados obtenidos y su similitud con el sistema captado. Puede verse como ejemplo el gráfico que se obtiene de seleccionar la casilla *Time plot* debajo de los datos recién importados (ver Fig. 3).

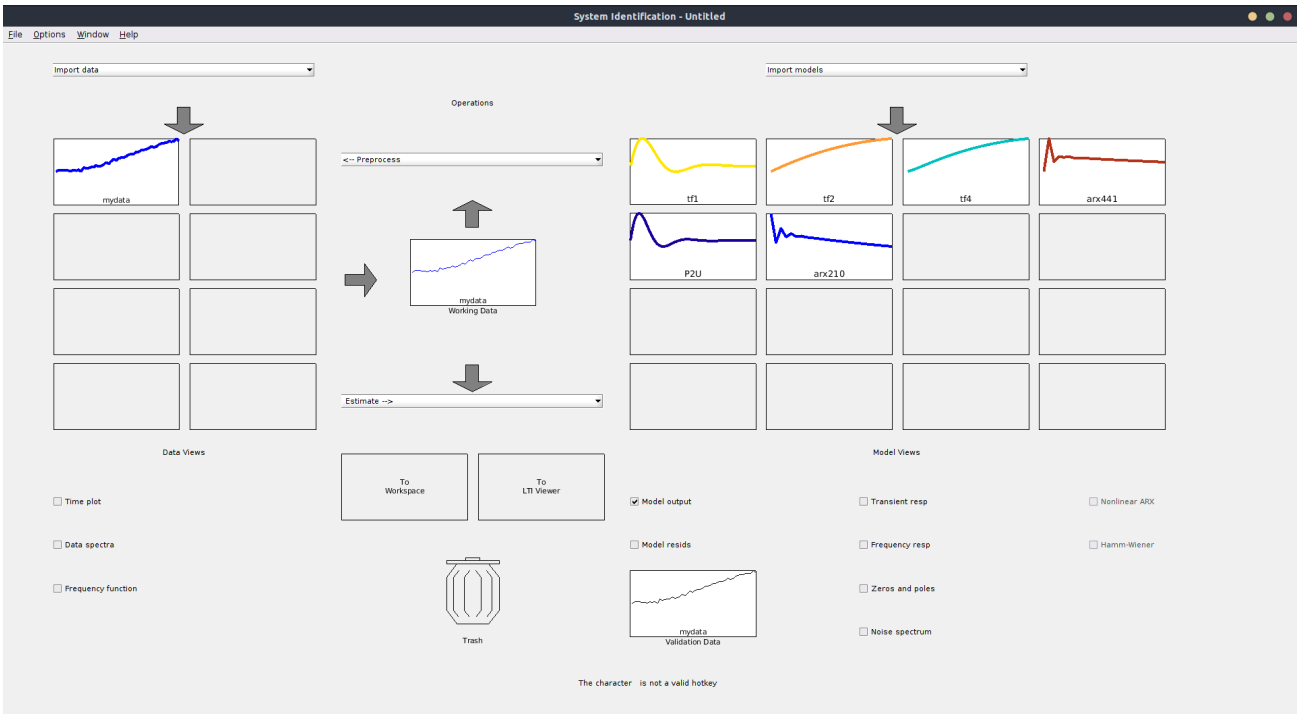

Figura 2: Uso de *System Identification* para el modelado del sistema motor-generador.

En la pestaña de *Estimate* podemos seleccionar qué método deseamos emplear. En nuestro caso hemos empleado los tres siguientes:

- Transfer Function Models
- Process Models
- Polynomial Models

Puede verse un versión más organizada de estos tres métodos en la figura 4. Puede apreciarse unos datos de entrada filtrados que reciben el nombre de *mydatafiltered* y cómo han sido empleado como *Working Data*. Debe decirse respecto a esto, que el proceder general suele ser: realizar la estimación/modelado mediante los datos sin filtrar eligiendo el método deseado. Luego se realiza el filtrado y se validan los modelos con dichos datos (arrastrando la casilla de los datos filtrados a la casilla de *Validation Data*). Si la diferencia en la salida de dichos modelos comparando sus índices de error (FIT[2]) es menor, es decir, los modelos mejoran al filtrar puede procederse a reestimar los modelos con

[2] $FIT = \left(1 - \frac{norm(out - \widehat{out}}{norm(out - med(out))}\right) \cdot 100$

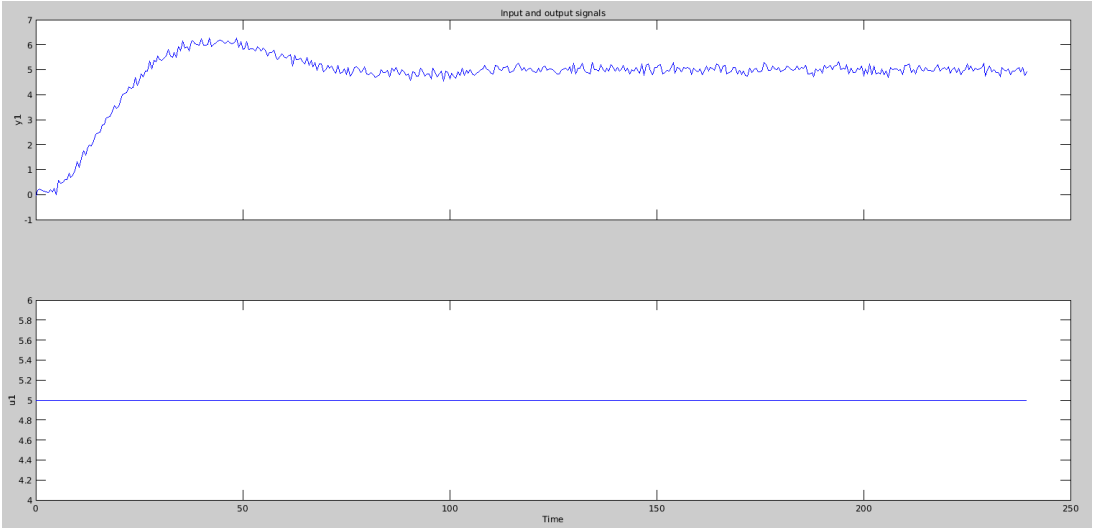

Figura 3: Gráfico obtenido mediante la casilla *Time plot* tras importar los datos a la app de *System Identification*.

los datos filtrados (empleando *mydatafiltered* como *Working Data*). De esta forma, en la primera fila de la vista de modelos (*Model Views*) estan los modelos donde se ha empleado el método de *Transfer Function Models*, algunos mediante los datos filtrados y otros sin filtrar (se indica en el nombre). De estos cuatro, los dos primeros son funciones continuas y los dos últimos son funciones discretas. En este caso se ha podido comprobar como el filtrado de datos no mejoraba significativamente (ver Fig. 5).

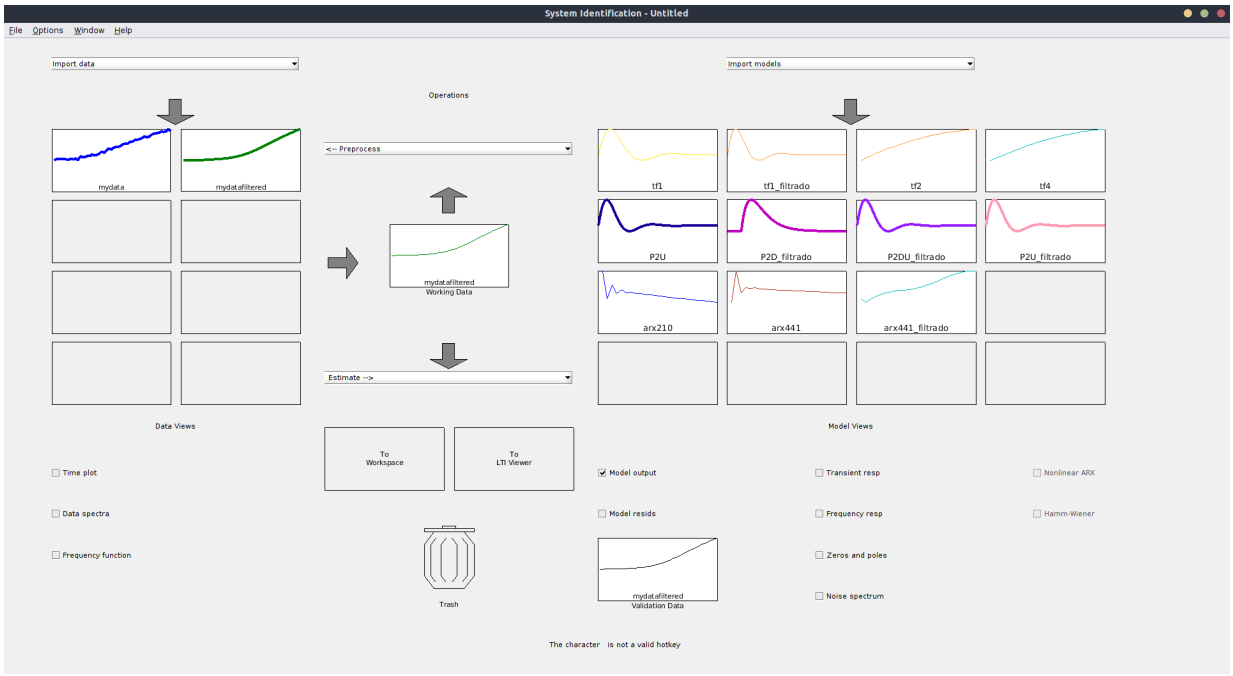

Figura 4: Simulaciones mediante *System Identification* empleando tres métodos diferentes.

La segunda fila es para los modelos obtenidos mediante *Process Models*. También se han estimado cuatro y todos ellos continuos. En casi todos se ha exigido dos polos complejos subamortiugados. Y en algunos además se ha permitido un *delay* a la hora de estimar el modelo. Una comparativa de las cuatro simulaciones puede verse en la figura 6. Puede comprobarse, como el caso anterior, que el

preprocesado de los datos no afecta de forma significativa a la estimación de los datos, pues los valor de FIT obtenidos en ambos casos son prácticamente idénticos.

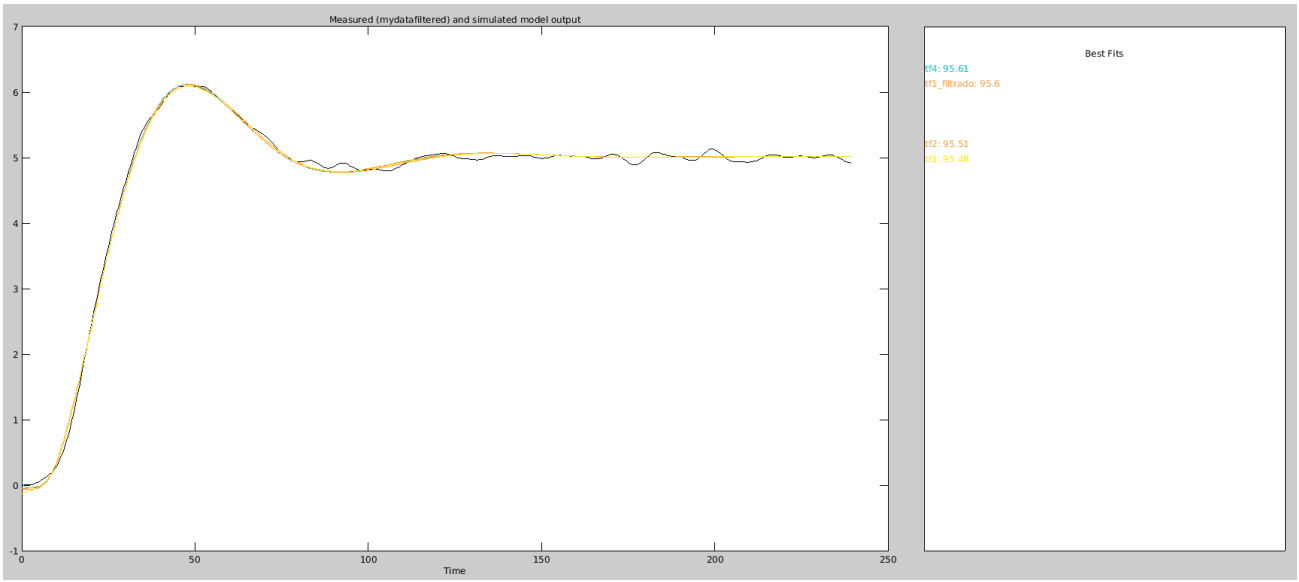

Figura 5: Comparativa de simulaciones obtenidas mediante el método de *Transfer Function Models*.

Finalmente la tercera fila en la sección de *Model Views* es la correspondiente al método *Polynomial Models*. Esta vez, todos ellos corresponden a modelos discretos. En este caso debe especificarse el orden del numerador y denominador para realizar la estimación. Por defecto, dicha configuración está en [4 4 1] y se ha querido emplearla para comparar con una estimación más realista (conocida la salida del sistema analizado) que supone en nuestro caso unos valores iniciales de [2 2 0]. Puede comprobarse en la figura 7 cómo el filtrado de los datos es *clave* en este método de estimación pues pasamos de valores de un 50 % a valores casi el doble, de un 90 % de FIT.

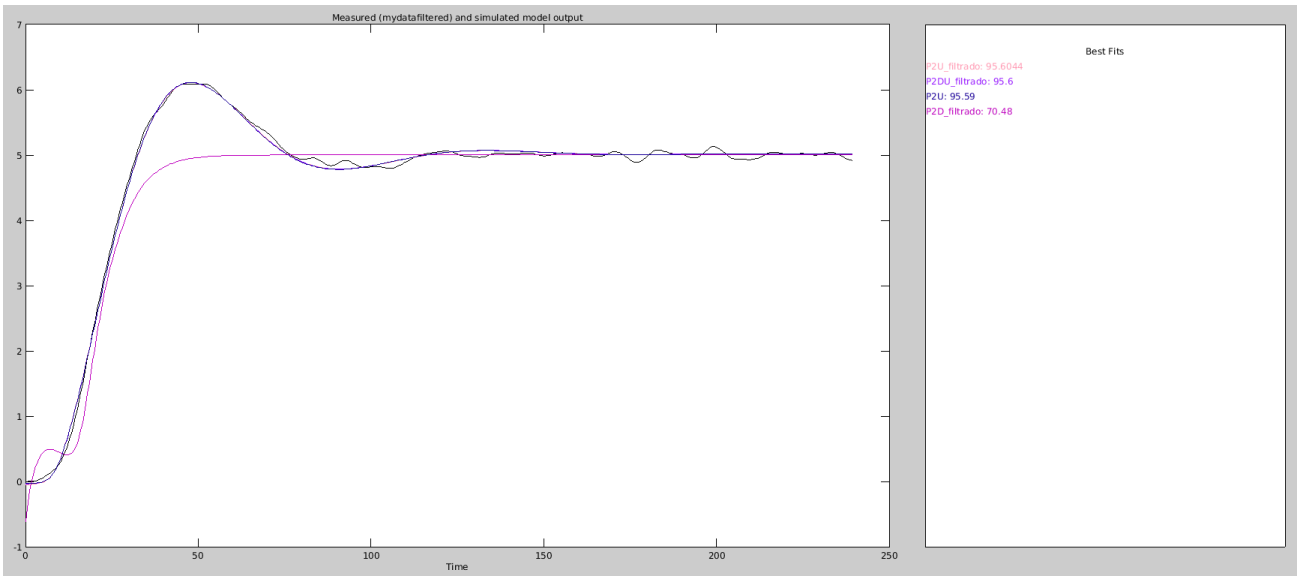

Figura 6: Comparativa de simulaciones obtenidas mediante el método de *Process Models*.

Sobre el preprocesado de datos de las muchas opciones posibles en nuestro caso hemos elegido *Filter*. En la pestaña que se abre debemos configurar la frecuencia en hercios, el rango de filtrado como *Stop band* y el valor del rango de filtrado, que en nuestro caso ha sido desde 0,1 hasta 1 Hz. Puede

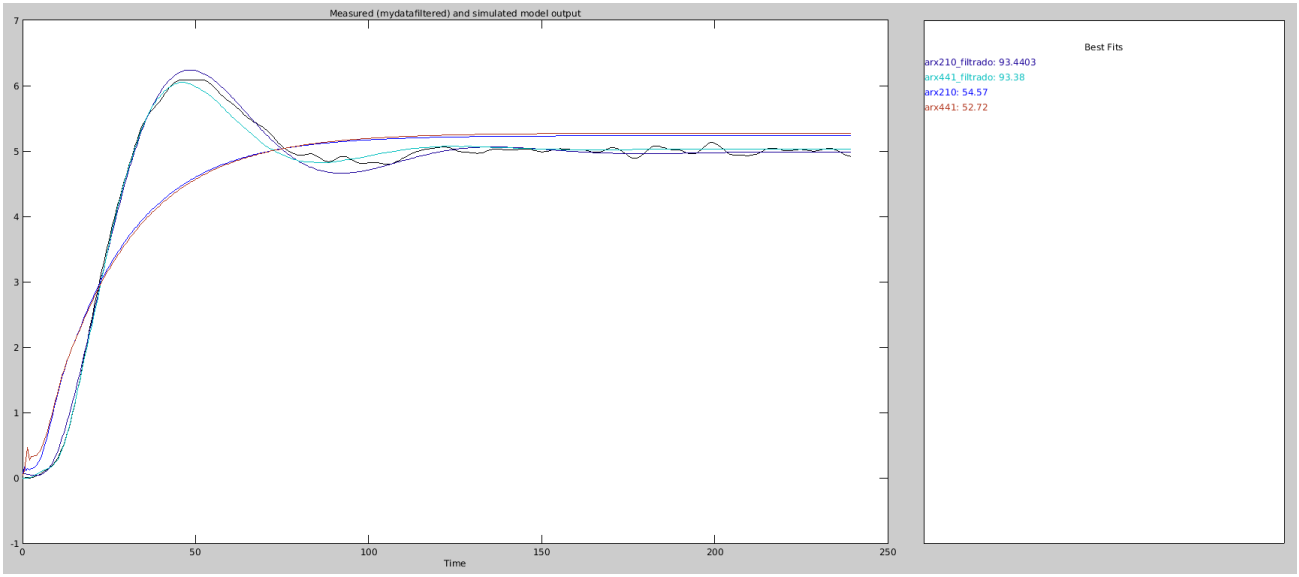

Figura 7: Comparativa de simulaciones obtenidas mediante el método de *Polynomial Models*.

verse una imagen de dicho filtro en la figura 8. Como mejora podría haberse tomado un rango más amplio de frecuencias para filtrar hasta los 0,05 Hz, pero posiblemente las estimaciones no mejoraran de forma apreciable.

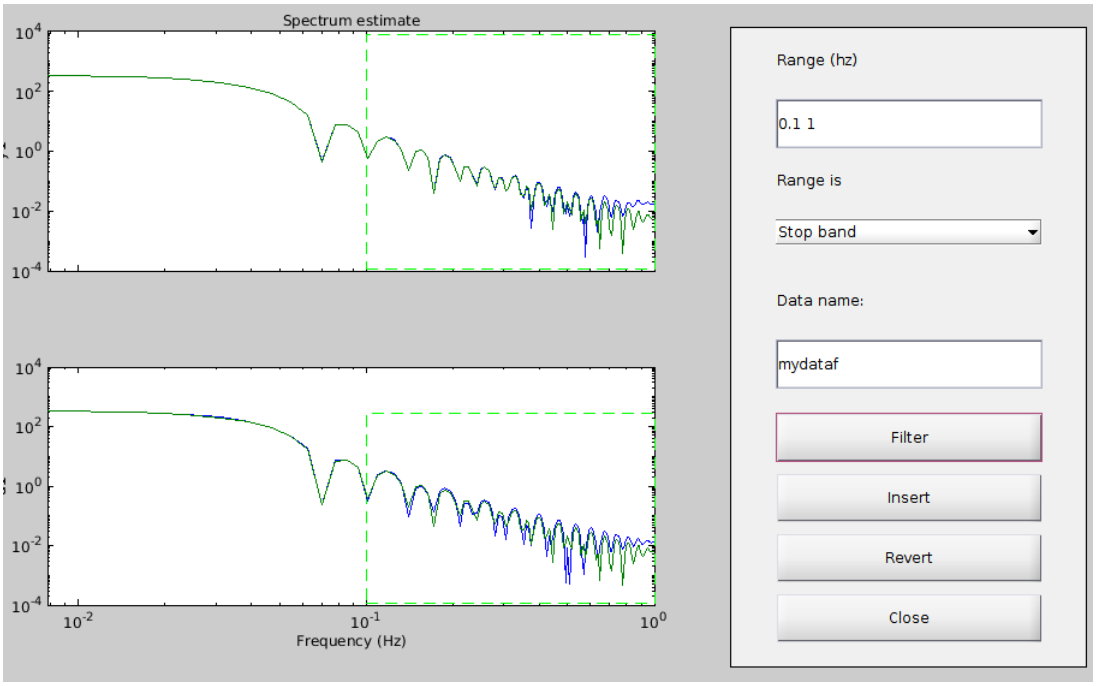

Figura 8: Filtrado mediante *Filter* de la banda de frecuencias entre los 0,1 y 1 Hz.

Simulación del modelo

Finalmente, debemos comparar la respuesta del sistema frente a la respuesta del modelo estimado, para cada uno de los métodos y según si los datos han sido filtrado o no. Los métodos se han nombrado, para agrupar los modelos, de la siguiente forma:

- 'Gz' y 'Gz' para los modelos continuo y discreto respectivamente obtenidos mediante *Transfer Function Models*.
- 'Proceso' para el modelo continuo obtenido mediante *Process Models*.
- 'ARX' para el modelo obtenido mediante *Polynomial Models*.

Los diferentes gráficos que muestran esto son:

- la figura 10 para el modelo *Gs* (continuo) y la figura 11 para el modelo *Gz* (discreto).
- la figura 12 para el modelo *Proceso*.
- la figura 13 para el modelo *ARX*.

Notar el retardo que muestra el modelo estimado frente a la salida del sistema cuando esta última se ha filtrado. Puede apreciarse como el sistema sufre un retraso (*delay*) debido al proceso de filtrado, puede comprobarse este efecto de forma más pronunciada en la señal de referencia (escalón de 5 voltios) en la figura 9.

Puede consultarse el enunciado de dicha práctica y la solución implementada en Matlab en este enlace.

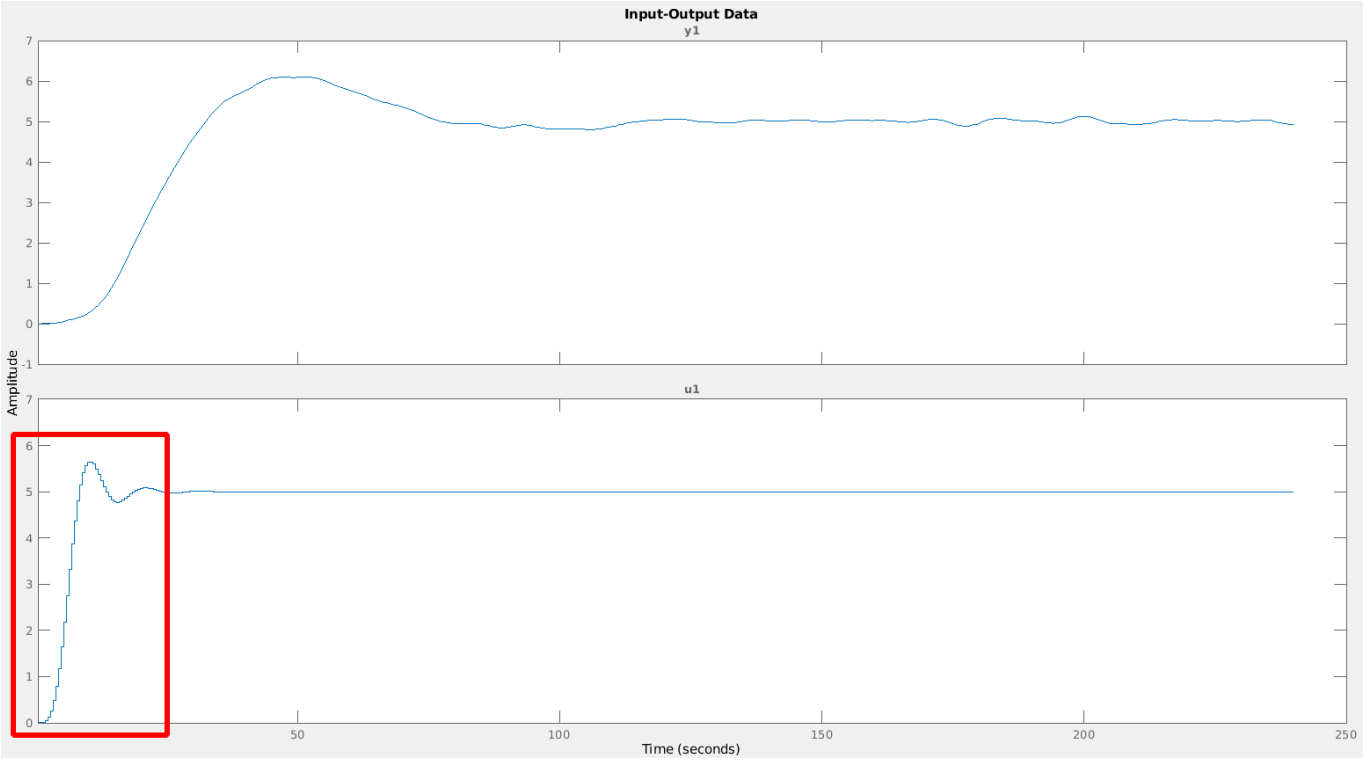

Figura 9: Respuesta del sistema filtrada. Puede observarse en un recuadro en rojo el retraso que sufre la entrada escalón al ser sometida al filtrado.

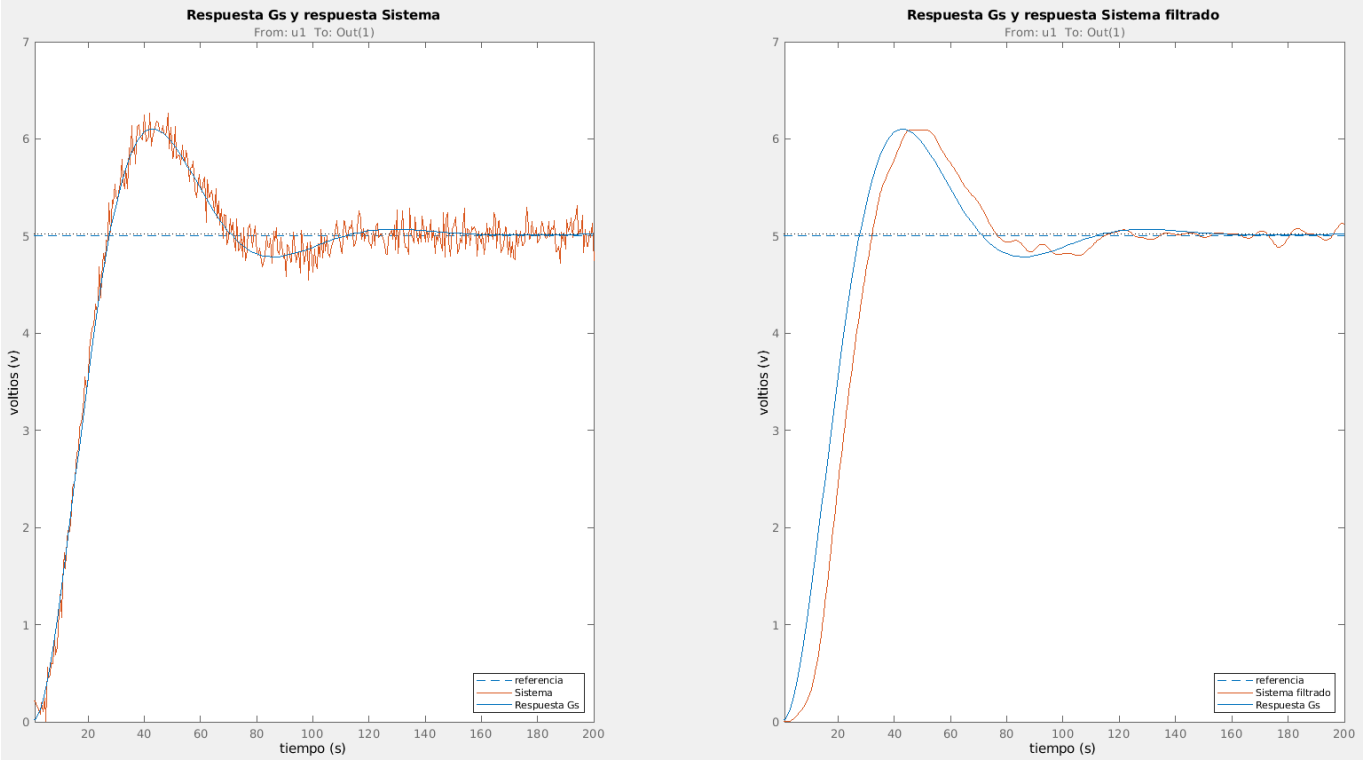

Figura 10: Respuesta del modelo estimado *Gs* frente a la respuesta del sistema sin filtrar (izqda.) y filtrada (dcha.).

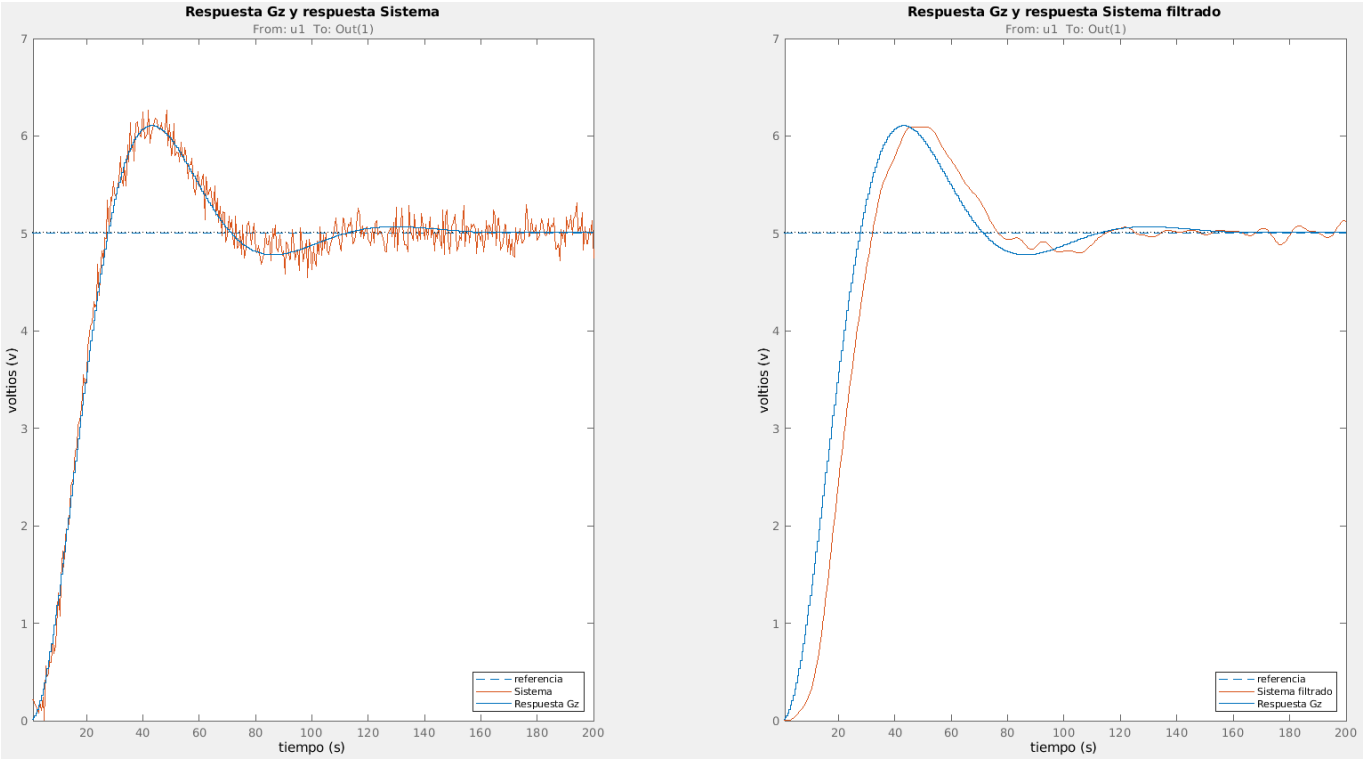

Figura 11: Respuesta del modelo estimado *Gz* frente a la respuesta del sistema sin filtrar (izqda.) y filtrada (dcha.).

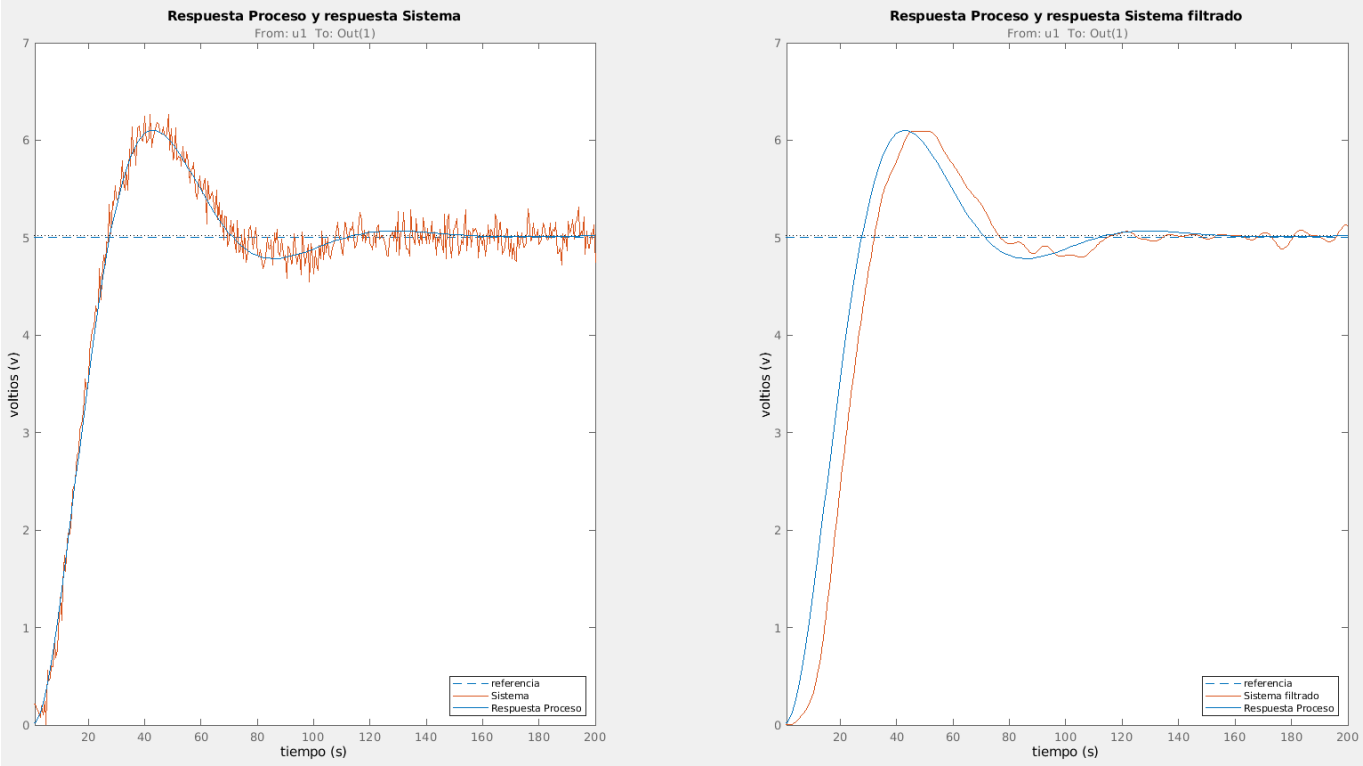

Figura 12: Respuesta del modelo estimado *Proceso* frente a la respuesta del sistema sin filtrar (izqda.) y filtrada (dcha.).

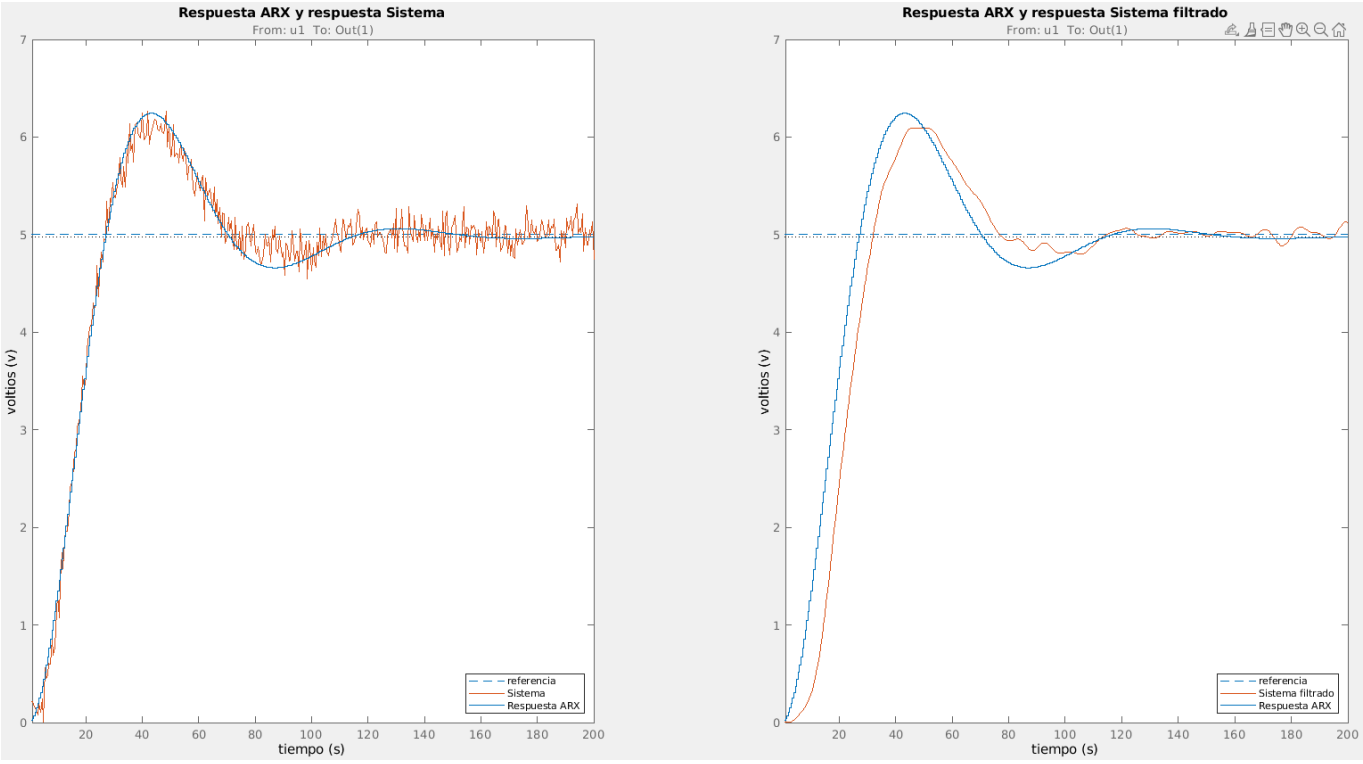

Figura 13: Respuesta del modelo estimado *ARX* frente a la respuesta del sistema sin filtrar (izqda.) y filtrada (dcha.).