COMPARACIÓN DE CONTROLADORES PID DE ORDEN FRACCIONARIO

FRACTIONAL ORDER PID CONTROLLERS COMPARISON

Uri Abdiel Vela Ortiz
Tecnológico Nacional de México / IT de la Laguna, México
abdiel_vel@outlook.com

Juan Sifuentes Mijares
Tecnológico Nacional de México / IT de la Laguna, México
jsifuentesm@correo.itlalaguna.edu.mx

Jesús Antonio Botello Triana
Tecnológico Nacional de México / IT de la Laguna, México
jabotellot@correo.itlalaguna.edu.mx

Recepción: 25/octubre/2021 **Aceptación:** 19/diciembre/2021

Resumen

Muchos sistemas dinámicos pueden ser mejor representados usando modelos no enteros, los cuales se basan en el cálculo fraccionario. Aunque durante varias décadas el cálculo fraccionario no fue desarrollado ni utilizado por carecer de una interpretación física y la complejidad en los cálculos que se requieren, en la actualidad ya se cuenta con herramientas computacionales lo suficientemente robustas para la resolución de este tipo de problemas. En este documento se presentan algunas definiciones básicas de cálculo fraccionario, así como algunas herramientas para el diseño de un controlador PID de orden fraccionario, en este caso desarrolladas para Matlab. Aunque los sistemas puedan representarse mediante un modelo de orden no entero, no es necesario que el sistema sea de este orden para poder diseñar un controlador de orden fraccionario.

Palabras Clave: Controlador difuso PID de orden fraccionario, controlador PID de orden fraccionario, sistema péndulo invertido de primer orden.

Abstract

Many dynamic systems can be represented in a better way using non integer models, which are based on fractional calculus. Although fractional calculus wasn't

developed or used for many decades due to its lacks of physical interpretations and the math complexity that was necessary, nowadays we have the necessary computational tools to solve these problems.

In this document, we represent some basic fractional calculus definitions, as well as some fractional order PID controller design tools which, in this case, were developed with MATLAB. Although the systems can be represented by a non integer order model, the system doesn't need to be a non integer order system in order to design its fractional order controller.

Keywords: *First-order inverted pendulum system, fractional order PID controller, fuzzy fractional order PID controller.*

1. Introducción

El sistema carro péndulo ha sido muy estudiado en el área de control debido a los distintos retos que éste puede presentar, como el hecho de que el sistema sea no lineal, subactuado, inestable, entre otros aspectos, por lo que su estudio puede proveer una primera aproximación a problemas que sistemas de mayor complejidad pueden plantear.

El cálculo fraccionario se ha venido desarrollando casi al mismo tiempo que el cálculo tradicional que conocemos por Newton y Leibniz [Chen, 2009], aunque menos utilizado debido a que existen distintas definiciones de la derivada fraccionaria [Mathai, 2017], y a la complejidad que pueden llegar a tener los cálculos de éstas, aplicadas a funciones sencillas en comparación con la derivada de orden entero, que es la que conocemos de los primeros cursos de cálculo que solemos llevar.

En cuanto a los controladores de orden fraccionario, desde hace ya varios años se ha estudiado en las áreas de ciencia e ingeniería, por ejemplo, los controladores PID de orden fraccionario, los cuales, a diferencia de los controladores de orden entero, agregan dos parámetros más de sintonización: el operador derivativo fraccionario y el operador integral fraccionario, los cuales permiten un mayor rango al controlar una planta debido a que los polos de sistema pueden encontrarse en el lado derecho del semiplano imaginario, siempre y cuando cumplan ciertas

condiciones [Tepljakov, 2017], pero a su vez aumentando la dificultad al sintonizar los parámetros del controlador.

En este documento se presentará una comparación entre controladores PID de orden fraccionario diseñados mediante herramientas de Matlab las cuales realizarán la sintonización de los parámetros del controlador de acuerdo a ciertos criterios de diseño, algoritmos de optimización y un controlador difuso.

Estado del arte

En [Bingi, 2020] se presentan tanto una introducción general al cálculo fraccionario como herramientas basadas en Scilab para su implementación, presenta diferentes métodos de aproximación y ejemplos de controladores PID de orden fraccionario. Por otro lado [Chen, 2009] explica brevemente de que tratan los sistemas de orden fraccionario, pero se enfoca en presentarnos un toolbox de Matlab llamado FOTF por sus siglas en inglés (Fractional Order Transfer Function), el cual nos permite realizar los cálculos de la simulación de un sistema fraccionario mediante una serie de pasos y código basados en tener la función de transferencia del sistema en orden fraccionario, así como la estabilidad del sistema y grafica de la simulación.

Muy parecido al anterior [Tepljakov, 2017], nos presenta otra herramienta de Matlab llamada FOMCOM en donde también se nos introduce al calculo fraccionario de manera breve, pero nos da una explicación más amplia de este toolbox, donde nos muestran las distintas opciones, ventanas y resultados que podemos obtener, así como cualquier parámetro que podamos modificar para mejorar el desempeño del controlador que se diseñe mediante la herramienta. Mediante FOMCOM es posible obtener un análisis de estabilidad, diagramas de bode, respuesta de la simulación, función de transferencia y realizar el diseño del controlador. Y en [Wang, 2016] se presenta un controlador de orden fraccionario aplicado a un carro péndulo, donde se nos da también una introducción y explicación del calculo fraccionario, la obtención de la planta con base en los parámetros en el robot que utilizaron y los controladores de orden fraccionario y entero que se obtuvieron, así como la comparación de los resultados.

2. Métodos

Cálculo Fraccionario

Habitualmente en el ámbito de las matemáticas y la ingeniería en general se acostumbra a que los operadores sean descritos por expresiones de orden entero como $\frac{d}{dt}$, D_t o simplemente D, sin embargo, con el desarrollo de del calculo fraccionario se pueden tener expresiones como $\frac{d}{dt^\gamma}$, D_t^γ o D^γ, en donde $\gamma \in R$.

Existen varias definiciones de la derivada en el cálculo fraccionario siendo tres de estas algunas de las más utilizadas [Duarte, 2011], Riemann-Liouville, Grünwald-Letnikov y la de Caputo. Puede usarse como notación (GL, RL o C) al lado izquierdo del operador para cada definición, la cual se puede omitir si no hay modo de confusión.

La definición de derivada de Grünwald-Letnikov es tal como en ecuación1.

$$GL_aD_t^\gamma\, f(t) = \lim_{h\to 0} \frac{1}{h^\alpha} \sum_{j=0}^{[(h-a)/h]} (-1)^j \binom{\alpha}{j} f(t-jh) \tag{1}$$

La definición de la integral de Riemann-Liouville está dada por ecuación 2.

$${}_aD_t^{-\alpha}\, f(t) = \frac{1}{\Gamma(\alpha)} \int_a^t (t-\tau)^{\alpha-1} f(\tau)\, d\tau \tag{2}$$

Y la definición de su derivada se muestra en ecuación 3.

$${}_aD_t^{\beta}\, f(t) = \frac{d^n}{dt^n}\left[{}_aD_t^{-(n-\beta)} f(t)\right] = \frac{1}{\Gamma(1-\beta)} \frac{d^n}{dt^n}\left[\int_a^t (-\tau)^{n-\beta-1} f(\tau)\, d\tau\right] \tag{3}$$

La derivada de Caputo se define en ecuación 4.

$${}_aD_t^{\alpha}\, f(t) = \frac{1}{\Gamma(-\gamma)} \int_a^t (t-\tau)^{-\gamma} f^{(m+1)}(\tau)\, d\tau \tag{4}$$

En donde $\alpha = m + \gamma$, $0 < \gamma \leq 1$ y m es un entero positivo. Y la definición de su integral se muestra en ecuación 5. Esta definición fue desarrollada para lograr un enfoque más practico y mayormente aplicable de lo que pueden ser otras definiciones como la de (RL)

$$ {}_aD_t^{\gamma} f(t) = \frac{1}{\Gamma(-\gamma)} \int_a^t (t-\tau)^{-1-\gamma} f(\tau)\, d\tau \tag{5} $$

Sistema Carro péndulo

El sistema carro péndulo es el sistema sobre el cual se realizaron los diseños de los controladores PID fraccionarios. El sistema carro péndulo se muestra en figura 1, cuyos parametros son M-masa del carro, m-masa del péndulo, l-longitud del péndulo, b-coeficiente de fricción, I-momento de inercia, X-la posición del carro,φ-ángulo del péndulo respecto a la vertical y u-la fuerza de entrada. En la tabla 1 del carro péndulo se muestran los valores de los parámetros de un sistema disponible en el Tecnológico de la Laguna y los cuales serán tomados para realizar las pruebas en simulación.

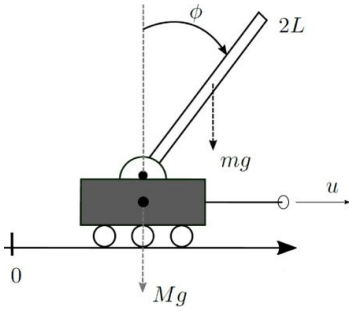

Figura 1 Sistema carro péndulo.

Tabla 1 Parámetros del carro péndulo.

Parámetro	Valor
M	2.27 kg
m	0.26 kg
l	0.295 m
I	0.0053 kg/m^2
b	0.1

El modelo matemático del carro péndulo puede expresarse mediante las ecuaciones mostradas en ecuación 6, [Wang, 2016].

$$ \begin{cases} (I + Ml^2)\varphi - ml\ddot{\varphi} = mgl\ddot{x} \\ (M+m)\ddot{x} + b\dot{x} - mgl\varphi = u \end{cases} \tag{6} $$

La función de transferencia del ángulo del péndulo y la fuerza externa puede expresarse mediante ecuación 7.

$$P(s) = \frac{\varphi(s)}{F(s)} = \frac{\frac{ml}{q}s^2}{s^4 + \frac{b(I+ml^2)}{q}s^3 - \frac{(M+m)mgl}{q} - \frac{bmgl}{q}s} \quad (7)$$

Donde, $q = [(M+m)(l+ml^2) - m^2l^2]$.

Y sustituyendo los valores de la tabla 1 en la ecuación 7 obtenemos la ecuación 8.

$$P(s) = \frac{0.09615s^2}{s^3 + 0.03501s^2 - 2.386s - 0.09432} \quad (8)$$

Controlador PID de orden fraccionario

Uno de los controladores más utilizados es el PID en el cual es necesario realizar la sintonización de los tres parámetros, la parte proporcional, la parte derivativa y la parte integral. Gracias al desarrollo del cálculo fraccionario es que se logra desarrollar un controlador PID de orden fraccionario. El controlador PID fraccionario es una generalización del controlador PID [Lachhab, 2013]. Esta generalización se puede representar como en ecuación 9.

$$PI^{\lambda}D^{\mu} = k_p + \frac{k_i}{s^{\lambda}} + k_d s^{\mu} \quad (9)$$

FOTF-Toolbox

En la actualidad existen varias herramientas para Matlab disponibles para el cálculo y control de orden fraccionario siendo FOTF una de las más utilizadas. FOTF es una herramienta de control para sistemas de orden fraccionario desarrollado por el profesor Dingyü Xue de China. Una herramienta muy útil para el diseño de orden fraccionario que durante años se ha ido mejorando para lograr un mejor desempeño mediante algoritmos de alta precisión [Xue, 2010].

El carro péndulo es un sistema no lineal el cual ha sido muy estudiado. En este documento se presenta una comparación de tres métodos de sintonización de controladores PID de orden fraccionario con un controlador PID clásico o de orden entero, aplicado al sistema carro péndulo.

La planta de este sistema la podemos ver en ecuación 8 y se observa que el sistema es inestable. De acuerdo con la teoría que se encuentra en el capítulo 7 de [Ogata, 2010], se puede diseñar un compensador para reducir el orden del sistema. Para este caso se elige un compensador como en la ecuación 10, que nos permita eliminar el polo que se encuentra del lado derecho del semiplano y los ceros del sistema.

$$H(s) = \frac{(s - 1.5264)}{s^2} \tag{10}$$

De este modo la planta resultante resulta en (ecuación 11)

$$P_2(s) = \frac{0.09615s^2}{(s + 1.5264)(s + 0.0395)} \tag{11}$$

El controlador PID

Los parámetros del controlador PID se sintonizaron mediante la herramienta optimpid de Matlab, de modo que el controlador clásico se muestra en ecuación 12.

$$C_1(s) = 26.1619 + \frac{28}{s} + 32.6547s \tag{12}$$

El controlador PID de orden fraccionario

Primer algoritmo

Ahora se sintonizarán los parámetros del controlador mediante algunos algoritmos de la herramienta FOTF [Xue, 2010]. Para nuestro primer controlador haremos uso de algunos parámetros de diseño como el margen de ganancia, margen de fase, constante de tiempo, frecuencia de corte, etc. [Ogata, 2010]. Nuestra planta de orden entero puede tener la forma de ecuación 13.

$$G(s) = \frac{K}{Ts + 1} e^{-Ls} \tag{13}$$

Y los parámetros utilizados son: $T1 = 1.5264$, $T2 = 0.0395$, $K = 0.09615$, $\omega c = 1\ rad/s$ y $\varphi m = 60°$.

En [Xue, 2010] se encuentra una función de Matlab la cual se usa para resolver este problema y obtener los parámetros del PID de orden fraccionario. Como vemos, este método requiere de parámetros de diseño para sintonizar el PID de orden

fraccionario. Las funciones utilizadas so fmincon y c9mfpid los cuales de acuerdo con los valores dados realiza varias iteraciones para tratar de obtener los valores más pequeños, pero óptimos para que funcione el PID de orden fraccionario. En la figura 2 se muestra un ejemplo del código para su implementación y simulación.

```
1
2 -   G=0.09615/(s^2+1.5659+.0603);%Función de transferencia de la planta
3 -   wcg=1; phi=60; A=-20; B=-20; wt=10; ws=.001;%parámetros
4 -   k=0.09615; T=1.5264; L=1; x0=rand(5,1);%Parametros
5 -   x=fmincon(@c9mfpid_opt,rand(5,1),[],[],[],[],[],...
6         [],@c9mfpid_con,'',wcg,phi,ws,B,wt,A,k,T,L);
7 -   t=0:0.01:20;
8 -   Gc=fopid(x);
9 -   step(feedback(G*Gc,1),t)
```

Figura 2 Función de sintonización usando algoritmo 9.4 de [Xue, 2010].

Segundo algoritmo

Esta segunda función utiliza un algoritmo de optimización numérica para poder calcular los valores de los parámetros del controlador [Xue, 2010].

Aunque más complejo en su ejecución, esta función no necesita de parámetros de diseño como el anterior, solamente necesitamos indicarle los valores entre los cuales se quiere que estén las ganancias, un ejemplo en el código se muestra en la figura 3.

```
21 -  s=fotf('s');
22 -  G=K/(s^2+1.5659+.0603);%Función de transferencia de la planta
23 -  xm=[0;0;0;0;0];%zeros(5,1);      %cota baja de los parámetros del PID fraccionario
24 -  xM=[60; 50; 40; 3; 3]; %Cota alta de los parámetros del PID fraccionario
25 -  x0=rand(5,1)'; %condiciones iniciales
26 -  t=0:0.01:20;
27 -  x=fminsearchbnd(@fpidfun,x0,xm,xM,[],G,t,1); %busqueda de parámetros optimizados del PID
28 -  Gc=fopid(x);%FDT del controlador PID fraccionario
29 -  x %vector que contiene los valores de los parámetros del PID fraccionario
30 -  figure(1)
31 -  bode(G*Gc)
32 -  figure(2)
33 -  step(feedback(G*Gc,1),t)
```

Figura 3 Algoritmo de Optimización Numérica.

Al igual que con el algoritmo anterior, mediante una serie de iteraciones calculará los valores de los 5 parámetros hasta obtener valores que nos regresará también en un vector x de 5x1 que también contiene los parámetros en el mismo orden.

Controlador PID difuso de orden fraccionario

El paquete de herramientas FOTF también cuenta con bloques de simulink, para este controlador se utiliza el bloque mostrado en la Figura 4 Modelo de simulink de un controlador PID difuso el cual tiene el bloque de un controlador PID de orden fraccionario el cual va a estar recibiendo actualizados. El bloque S-function es un bloque discreto cuyas salidas son los parámetros del controlador. Algunas características de este diagrama son:

- El modelo de la planta no tiene que ser de orden entero, se puede reemplazar el bloque de función de transferencia por uno de orden fraccionario.
- También se le puede agregar un retraso o delay a la señal, en caso de ser necesario.
- Como ya se mencionó, la función del bloque S-function es el de actualizar el valor de los parámetros del controlador en el espacio de trabajo de Matlab, además, no es necesario conectarlo con otro bloque. El código empleado junto con el sistema de simulink se muestra en la figura 5

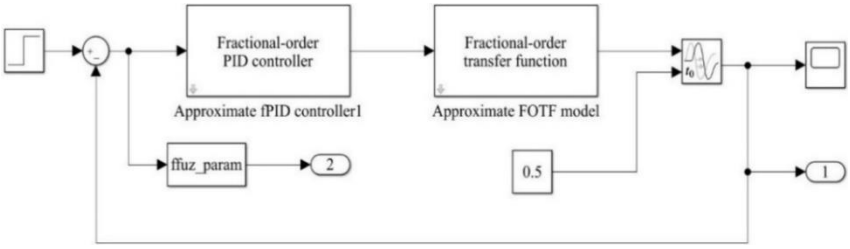

Figura 4 Modelo de simulink de un controlador PID difuso.

```
10 -    s=fotf('s');
11 -    G=0.09615/(s^2+1.5659*s +.0603);%Función de transferencia de la planta
12 -    Kp=.0014; Ki=.331; Kd=1.1907;
13 -    lam=.89; mu0=.38;
14 -    T=0.001; fuz=readfis('voffuzzy');
15 -    K0=[Kp,Kd,Ki,lam,mu0]';
16 -    [t0,~,y0]=sim('c9mvofuz');
17 -    plot(t0,y0(:,1))%gráfica de la respuesta
18 -    figure(2)
19 -    plot(t0,y0(:,2:6))%Grafica de los parámetros
```

Figura 5 Código para iniciar el controlador PID difuso de orden fraccionario.

En este código se colocan valores a los parámetros del controlador como condiciones iníciales. Estas condiciones iníciales podrían ser los valores que se

calcularon mediante un algoritmo de optimización, o podrían ser completamente arbitrarios, lo que nos indicaría que las condiciones iníciales no son tan importantes en algunos sistemas de control.

3. Resultados

Una vez visto los diferentes métodos o algoritmos para diseñar los controladores PID de orden fraccionario procederemos con los resultados de los mismos. Primeramente, el Controlador PID clásico, como ya vimos en la ecuación 12., fue sintonizado mediante la herramienta de Matlab optimPID. Cuya respuesta ante un escalón unitario se muestra en la figura 6

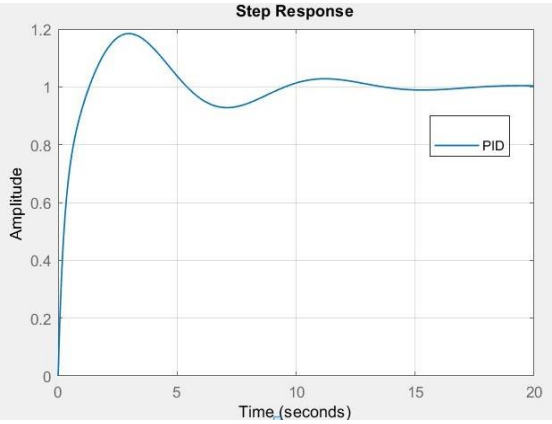

Figura 6 Respuesta del sistema con PID clásico ante un escalón unitario.

La primera función que utilizamos para sintonizar los parámetros del controlado nos regresará un vector x de 5x1 el cual contiene los parámetros kp, ki, kd, λ y μ en ese orden. Tomando todos los valores de este vector podemos ya tener nuestra Función del controlador (ecuación 14).

$$C_2(s) = 18.6092 + \frac{15.76}{s^{1.3203}} + 14.22s^{0.6268} \tag{14}$$

La figura 7 muestra la respuesta del sistema con el controlador PID de orden fraccionario ante un escalón unitario, cuyo controlador resulta como en ecuación 15.

$$C_3 = 18.609 + \frac{15.76}{s^{1.32}} + 14.22s^{0.626} \tag{15}$$

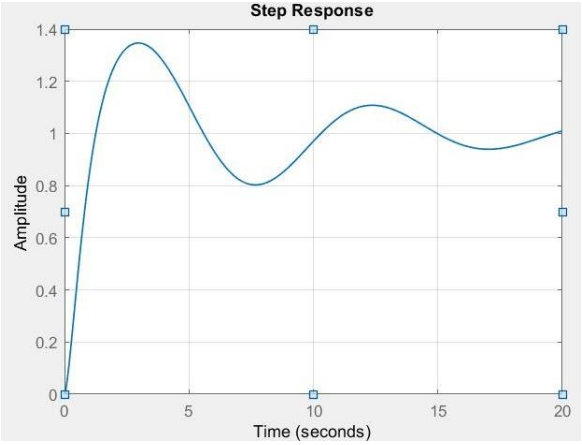

Figura 7 Respuesta del sistema con controlador PID fraccionario ante un escalón unitario.

El algoritmo utilizado, es un algoritmo de optimización que se basa en los parámetros de diseño que se requieren para el sistema, buscando así satisfacer estas necesidades de diseño, es por esto que podría trabajar mejor o peor dependiendo de qué es lo que se esté buscando.

La figura 8 muestra la respuesta del sistema con el controlador PID, pero en este caso diseñado con los algoritmos de optimización, los cuales lo que se busca es reducir los sobreimpulsos, oscilaciones y tiempo de establecimiento, a diferencia del anterior no necesita los parámetros de diseño, pero esto hace que los cálculos requeridos sean más, y por lo tanto tardará un poco más en realizar los cálculos. Los valores obtenidos se muestran en la ecuación 16.

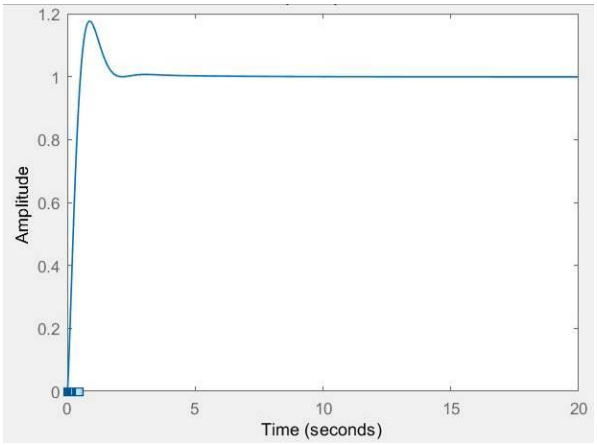

Figura 8 Respuesta del sistema con PID de orden fraccionario.

$$C_4 = 35.452 + \frac{48.86}{s^{1.0094}} + 36.17s^{1.4714} \tag{16}$$

Por último, se muestra la respuesta de un controlador PID de orden fraccionario difuso en la figura 9. Esto con la ayuda de un diagrama de bloques de Simulink el cual se encuentra en la herramienta o toolbox FOTF. Aunque el bloque se usa mediante código es necesario conocer que los bloques que contiene pueden cambiarse, hablando específicamente del bloque de Función de Transferencia de Matlab, el cual es de orden entero y se podría cambiar por uno de orden fraccionario.

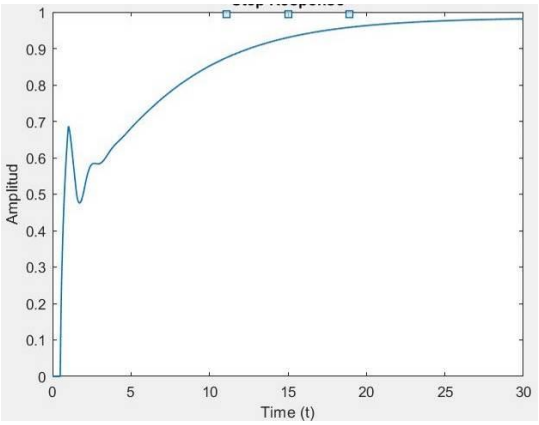

Figura 9 Respuesta de escalón unitario, sistema con controlador PID fraccionario difuso.

También en la figura 10 se muestran la variación de los parámetros del controlador PID difuso de orden fraccionario.

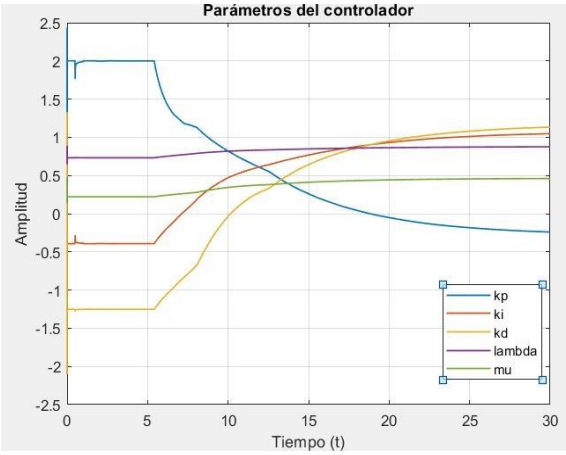

Figura 10 Variación de los parámetros del controlador PID difuso de orden fraccionario.

4. Discusión

Sabemos que el PID es uno de los controladores más utilizados en la industria, y el hecho de que también se pueda aplicar la teoría del cálculo fraccionario abre muchas posibilidades de desarrollo del tema. Y como ya se vio existen muchos métodos de aproximación para estimar o resolver un sistema de ecuaciones diferenciales de orden No entero. Es importante notar que a lo largo de estas últimas dos o tres décadas las herramientas para utilizar y aplicar el cálculo y controladores de orden fraccionario se han incrementado. En este documento solo se utilizó la herramienta FOTF pero existen más herramientas igual de útiles.

Al ver las respuestas del sistema con distintos controladores y hacer una comparación directa, en la figura 11 se muestran los controladores PID de orden entero y fraccionario, y en la figura 12 el controlador difuso.

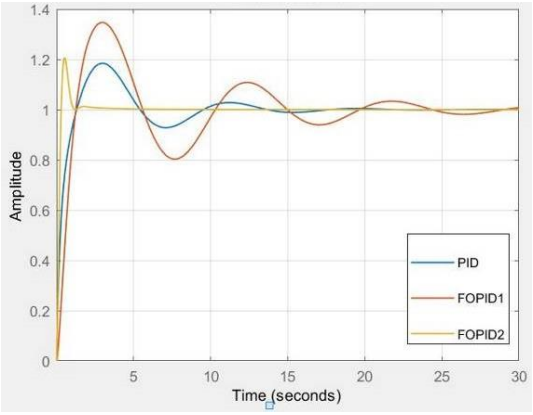

Figura 11 Comparación de los controladores PID clásico y de orden fraccionario.

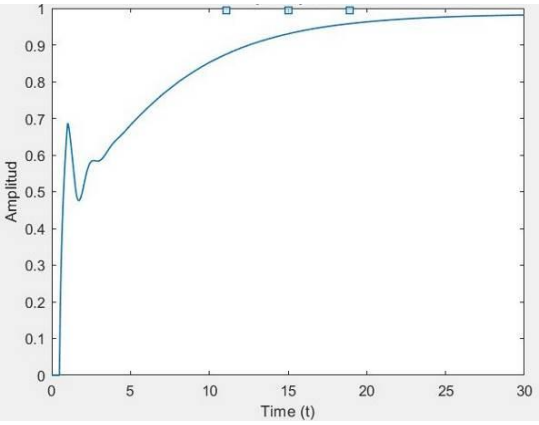

Figura 12 Respuesta del sistema con un controlador PID difuso de orden fraccionario.

5. Conclusiones

Definitivamente se puede ver que un controlador de orden fraccionario supera el desempeño de uno clásico, esto al menos con el que fue diseñado con algoritmos de optimización numérica, y es posible que, debido a que los parámetros de diseño que se eligieron no fueron los más apropiados, el controlador PID de orden fraccionario no logró mejorar el desempeño del PID clásico, por lo que no se debe de asumir que no se mejorará el desempeño del sistema mediante este método de diseño.

Por otra parte, es notable la diferencia que existe entre el controlador diseñado con algoritmos de optimización numérica, ya que no requiere de parámetros de diseño para el controlador, sino que lo que trata de reducir son los sobreimpulsos, oscilaciones y tiempo de establecimiento a lo mínimo posible. Por su parte, el controlador difuso es un poco más lento en la respuesta, pero sin sobreimpulsos.

En conclusión, al menos en la parte de diseño de un controlador y la parte de simulación de un sistema de orden fraccionario no es tan complicada de realizar gracias a las herramientas que se han desarrollado. Y en cuanto a los controladores, el PID clásico sigue mostrando muy buenos resultados de desempeño, pero está más que comprobado que, en general, los controladores PID de orden fraccionario superan el desempeño. Y entre los controladores de orden fraccionario el de mejor desempeño sin duda fue el que fue hecho con algoritmos de optimización.

6. Bibliografía y Referencias

[1] Bingi, K., Ibrahim, R., Noh Karsiti, M., Miya Hassan, S., & Harindan, V. R. (2020). *Fractional-order Systems and PID Controllers.* Springer.

[2] Chen, Y., Petrás, I., & Xue, D. (2009). Fractional Order Control - A Tutorial. *American Control CONFERENCE.* St. Louis, MO, USA.

[3] Duarte Ortigueira, M. (2011). *Fractional Calculus for Scientists and Engineers.* Springer.

[4] Lachhab, N., Svaricek, F., Wobbe, F., & Rabba, H. (2013). Fractional Order PID Controller (FOPID)-Toolbox. *European Control Conference (ECC).* Zürich,Switzerland.

[5] Mathai, A. M., & Haubold, H. J. (2017). Fractional and Multivariable Calculus: model Building and Optimization Problems. Springer.

[6] Ogata, K. (2010). Ingeniería de Control Moderna (5ta ed.). Pearson.

[7] Podlubny, I. (1999). Fractional differential equations: an introduction to fractional derivatives, fractional differential equations, to methods of their solution and some of their applications. San Diego: Academic Press.

[8] Tepljakov, A. (2017). Fractional-order Modeling and Control of Dynamic Systems. Springer.

[9] Wang, C., Yin, G., Liu, C., & Fu, W. (2016). Design and Simulation of Inverted pendulum system based on fractional PID controller. 11th Conference onIndustrial Electronics and Applications (ICIEA).

[10] Xue, D. (2010). Fractional-order Control Systems: Fundamentals and numerical implementations. Waltern de Gruyter.