PROGRAMACIÓN DE INTERFAZ GRÁFICA EN APP DESIGNER DEL MATLAB PARA REPRESENTAR LA SERIE DE FOURIER EN CURSO INTRODUCTORIO DE TELECOMUNICACIONES

Pedro HUAMANÍ NAVARRETE[1]
Universidad Ricardo Palma
phuamani@urp.edu.pe

RESUMEN

Este artículo describe el procedimiento para la implementación de una interfaz gráfica de usuario utilizando el entorno de desarrollo App Designer del software Matlab y sus Toolbox Signal Processing y Symbolic Math, que permitieron obtener los coeficientes de la Serie de Fourier para siete señales periódicas seguido de una representación gráfica en el dominio del tiempo y la frecuencia a través de impulsos. Asimismo, dicha interfaz fue implementada utilizando variados componentes como Drop Down, Button, Knob, Axes, entre otros más, los cuales permitieron la elección del número de componentes sinusoidales utilizados en la descomposición, así como la elección del tipo de señal, su amplitud, frecuencia, nivel DC y tiempo de duración de la misma. Por lo tanto, con esta implementación se complementa el proceso de enseñanza-aprendizaje desarrollada de forma remota para un tema del curso de Telecomunicaciones I, del VI semestre académico de la Carrera de Ingeniería Electrónica de la Universidad Ricardo Palma. Finalmente, se presentan los resultados gráficos para el caso de cuatro elecciones de señales periódicas, con diferentes amplitudes, frecuencias y niveles DC; así como los resultados numéricos en cuanto al error medio cuadrático, comprobándose un menor valor para la señal triangular a pesar de utilizar un número reducido de componentes sinusoidales.

PALABRAS CLAVES

Serie de Fourier, App Designer, Toolbox Symbolic Math, Toolbox Signal Processing.

PROGRAMMING OF GRAPHICAL INTERFACE IN APP DESIGNER OF MATLAB TO REPRESENT THE FOURIER SERIES IN INTRODUCTORY TELECOMMUNICATIONS COURSE

ABSTRACT

This article describes the procedure for the implementation of a graphical user interface using the App Designer development environment of the Matlab software and its Toolbox Signal Processing and Symbolic Math, which allowed obtaining the Fourier Series coefficients for seven periodic signals followed by a graphical representation in the domain of time and frequency through pulses. Likewise, this

1 Facultad de Ingeniería, Investigador CONCYTEC - IEEE Member. phuamani@urp.edu.pe

interface was implemented using various components such as Drop Down, Button, Knob, Axes, among others, which allowed the choice of the number of sinusoidal components used in the decomposition, as well as the choice of the type of signal, its amplitude, frequency, DC level and duration time of the same. Therefore, this implementation complements the teaching-learning process developed remotely for a topic in the Telecommunications I course of the 6th academic semester of the Electronic Engineering Degree at Ricardo Palma University. Finally, the graphical results are presented for the case of four choices of periodic signals, with different amplitudes, frequencies and DC levels; as well as the numerical results in terms of the mean square error, showing a lower value for the triangular signal despite using a reduced number of sinusoidal components.

KEYWORDS
Fourier series, App Designer, Toolbox Symbolic Math, Toolbox Signal Processing.

Recibido: 30/04/2021
Aprobado: 02/08/2021

INTRODUCCIÓN

El progreso de las herramientas computacionales ha permitido que cada vez sean más utilizadas en el proceso de enseñanza-aprendizaje, principalmente en esta época que venimos experimentando una educación a distancia debido al estado de emergencia sanitaria decretada por el gobierno peruano desde el pasado marzo del 2020, a causa de la pandemia del COVID-19. Por esta razón, las clases a nivel de pregrado y posgrado en la Universidad Ricardo Palma y otras instituciones más de nuestro país, vienen realizándose de forma remota utilizando las plataformas adecuadas para la interacción síncrona y/o no síncrona con los estudiantes, y así transmitir los conocimientos y realizar las evaluaciones según el aprendizaje alcanzado.

Es así que, para complementar el dictado de las clases de teoría de un tema del curso de Telecomunicaciones I, del VI semestre académico de la Carrera de Ingeniería Electrónica de la Universidad Ricardo Palma, se optó por utilizar el software de Computación Científica Matlab, los Toolbox Signal Processing y Symbolic Math, y el entorno de desarrollo App Designer, para diseñar una Interfaz Gráfica de Usuario que permita la representación de la Serie de Fourier a través de la elección de un grupo de señales periódicas con características particulares de amplitud, frecuencia y nivel de continua (DC) u offset, y mostrando a su vez un gráfico temporal y frecuencial según la cantidad de componentes sinusoidales elegidas.

De esta manera, se permite que el estudiante asimile de manera más interactiva el concepto matemático del análisis de Fourier tal como en [1], [2] y [3], así como también lo motive a mejorar dicha interfaz añadiendo más propiedades o tipos de señales periódicas.

METODOLOGÍA DESARROLLADA

La metodología desarrollada en este artículo está constituida de tres etapas: i) Diseño de la interfaz gráfica de usuario utilizando el App Designer del Matlab, donde se describen los componentes utilizados de este entorno de desarrollo. ii) Selección de las Señales Periódicas más comunes y utilizadas en el curso de Telecomunicaciones I, utilizando algunos comandos del Toolbox Signal Processing, y iii) Desarrollo de la Programación para la obtención de los coeficientes de la Serie de Fourier con apoyo del Toolbox Symbolic Math.

DISEÑO DE LA INTERFAZ GRÁFICA DE USUARIO

Para ello se utilizó el App Designer del Matlab, el cual es un entorno de desarrollo interactivo que permite diseñar una aplicación y programar su comportamiento; además, proporciona una versión totalmente integrada del editor de MATLAB® y un gran conjunto de componentes interactivos de la Interfaz de Usuario [4]; entre ellos, botones, casillas de verificación, árboles y listas de desplegables, así como también elementos de control como medidores, indicadores luminosos, controles y conmutadores, componentes de contenedor como pestañas y paneles, diseños de mallas, entre otros más [5]. A continuación, en la figura 1, se muestra una captura de pantalla del entorno App Designer donde se observa a la izquierda la Ventana Biblioteca de Componentes (Component Library), al lado derecho las ventanas Explorador de Componentes (Component Browser) y Propiedades de los Componentes (Component Properties), y en la parte central la ventana de Diseño con la opción "Design View" activada; asimismo, a su lado, se encuentra la opción "Code View" que permite visualizar el código de programación creado para la aplicación en blanco.

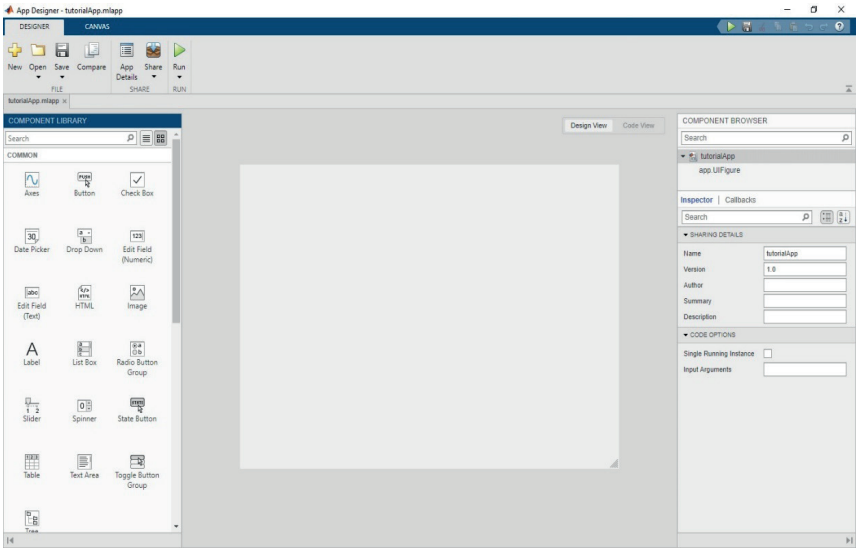


Figura 1. Captura de Pantalla del entorno desarrollador App Designer del Matlab.

De esta manera, para el diseño de la interfaz gráfica se optó por utilizar los siguientes componentes: un componente Drop Down (muestra la lista de señales periódicas utilizadas), un componente Spinner (para elegir el número de componentes sinusoidales), un componente Lamp (indicación si la aplicación se encuentra ejecutándose), un componente Button (para calcular los coeficientes de la Serie de Fourier y representarlos gráficamente en tiempo y frecuencia), cuatro componentes Knob (perillas de Amplitud, Nivel DC, Frecuencia y Tiempo), seis componentes EditField (visualización de los valores numéricos de las perillas, del error medio cuadrático y del tiempo aproximado de ejecución de los cálculos), seis componentes Label (asignación del nombre de las perillas, del error medio cuadrático y del tiempo aproximado de ejecución de los cálculos), dos componentes Axes (uno para la representación gráfica temporal y el otro para la frecuencial), un componente Image (para el logo de la Universidad Ricardo Palma) y tres componentes Panel (para agrupar algunos de los componentes anteriormente citados). Seguidamente, en la figura 2, se muestra una captura de pantalla del diseño de la interfaz gráfica indicando las opciones por defecto que presenta cada componente utilizado.

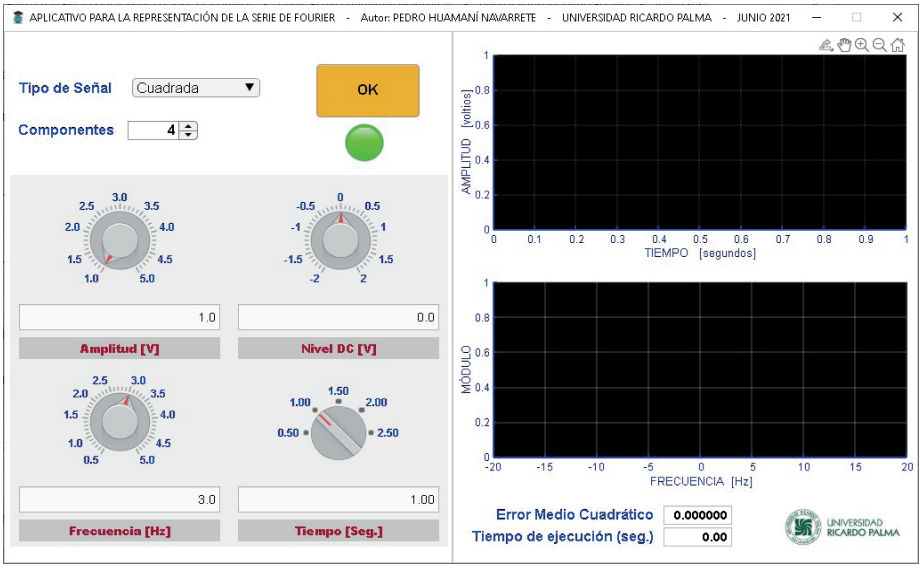


Figura 2. Captura de Pantalla del diseño de la Interfaz Gráfica, utilizando el App Designer del Matlab.

Asimismo, el componente Drop Down contiene los nombres de 5 tipos de señales periódicas (Cuadrada, Triangular, Diente Sierra Right, Diente Sierra Left y Pulso), más 2 con el nombre de arbitrarias que más adelante serán comentadas. De igual forma, para el componente Spinner se determinó el uso de hasta 20 componentes sinusoidales, pero por defecto se indicó la cantidad de 4. Y, como también, por defecto se estableció la perilla de Amplitud en 1

voltio (1 a 5 voltios), la de Nivel DC en 0 voltios (-2 a 2 voltios), la de Frecuencia en 3 Hertz (0.5 a 5 Hertz), y la de tiempo en 1 segundo (0.5 a 2.5 segundos). Por otro lado, los gráficos se establecieron por defecto de 0 a 1 segundo para el caso temporal, y de -20 a 20 Hz para el caso frecuencial. En este último caso, se estableció dicho dominio por la necesidad de mostrar el resultado teórico del módulo de la Serie de Fourier.

SELECCIÓN DE LAS SEÑALES PERIÓDICAS

Teniendo en cuenta que existen infinitas señales periódicas, se determinó utilizar solamente un grupo de 7 las cuales son las más comunes y utilizadas en los ejercicios de aplicación de la asignatura Telecomunicaciones I. Dicha elección también se sustenta por la flexibilidad que brinda el propio entorno de Matlab y su Toolbox Signal Processing, para la creación de ese grupo de señales periódicas [6], [7]. A continuación, se describe la función que contiene el código de programación en Matlab para generar cada señal periódica.

- **Función "GraficaSenal".** Para la representación vectorial y gráfica de las 7 señales periódicas seleccionadas en este artículo, se desarrolló la función denominada "GraficaSenal" en el editor del App Designer del Matlab utilizando la sentencia SWITCH y los comandos SQUARE, SAWTOOTH, SIN y combinaciones de ellos. Seguidamente, la figura 3 muestra una captura de pantalla de la función desarrollada en el editor, donde se observa que se establecieron 7 parámetros de entrada y 1 de salida. En cuanto a los parámetros de entrada se definieron *APP* (puntero a los datos de la aplicación), *A* (amplitud de la señal periódica seleccionada), *F* (frecuencia de la señal periódica seleccionada), *DC* (nivel de continua para la señal periódica seleccionada), *T* (tiempo para la representación de la señal periódica seleccionada), *Fs* (frecuencia de muestreo para la señal periódica seleccionada) y *SENAL* (valor del 1 al 7 para la señal periódica seleccionada). Por otro lado, en cuanto al parámetro de salida, se definió la variable tipo arreglo denominada "resultado" la cual contiene dos variables tipo vector a la vez; la primera es un vector referente al tiempo en función a la frecuencia de muestreo y al valor seleccionado por el componente perilla para la variable "Tiempo ", mientras que la segunda es también un vector que contiene la señal periódica seleccionada desde el componente Drop Down y con las características elegidas de los cuatro componentes tipo perilla.

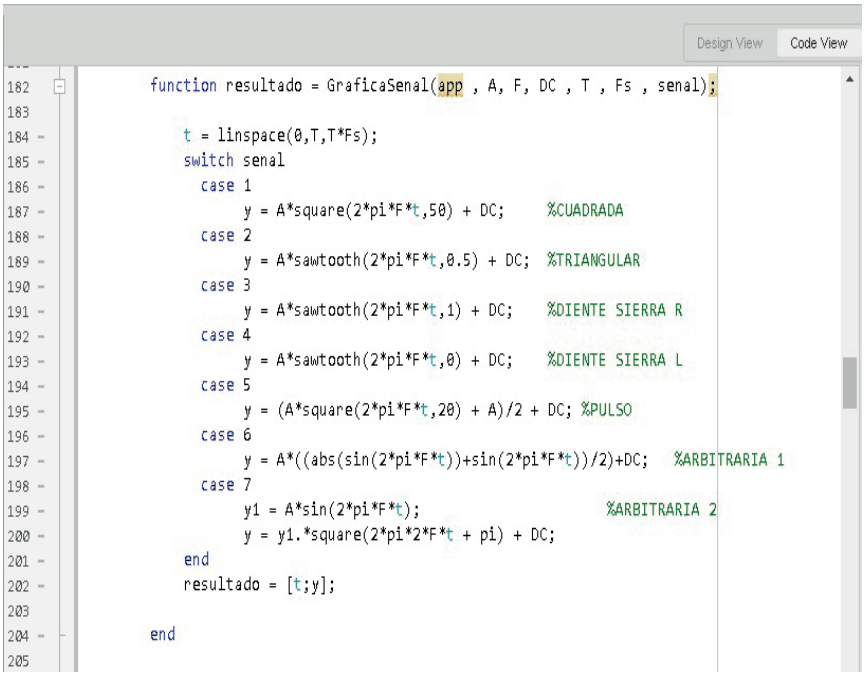


Figura 3. Programación de la Función "GraficaSenal" en el editor del App Designer del Matlab.

- **Representación gráfica temporal de las 7 señales periódicas.** Para la representación gráfica temporal del grupo de señales periódicas seleccionadas, se utilizaron algunas funciones del Toolbox Signal Processing del Matlab [6]. Así tenemos; para la señal cuadrada se empleó la función SQUARE, para la señal triangular SAWTOOTH, para las señales diente de sierra hacia la derecha e izquierda también se utilizó SAWTOOTH, pero complementado con un parámetro que determinó la inclinación, para la señal pulso también se empleó la función SQUARE, mientras que para las señales arbitrarias 1 y 2 se utilizaron combinaciones de las funciones anteriormente mencionadas.

A continuación, en las figuras 4 y 5 se muestran las representaciones gráficas de las 7 señales periódicas utilizadas. Respecto a las señales periódicas Arbitrarias 1 y 2, estas fueron implementadas de la siguiente manera. Para el caso de la primera señal arbitraria se utilizaron las funciones SIN y ABS para generar la onda rectificada (media onda) con amplitud, frecuencia y nivel DC seleccionado por las perillas respectivas; para ello, se realizó la suma de dos ondas sinusoidales con la misma frecuencia, pero una de ellas con la aplicación de la función absoluto para anular el semiciclo negativo (ver la figura 5.b).

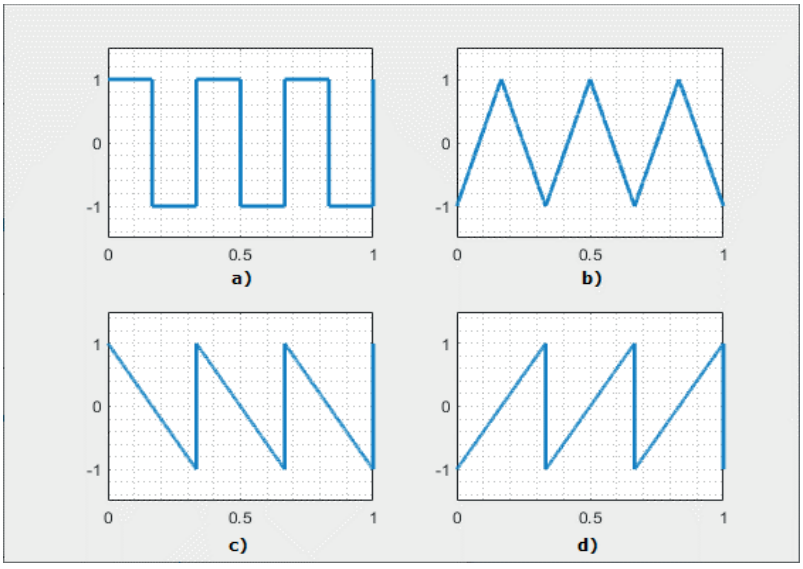


Figura 4. Representación temporal de las señales periódicas utilizadas. a) Cuadrada. b) Triangular. c) Diente de sierra hacia la izquierda. d) Diente de sierra hacia la derecha.

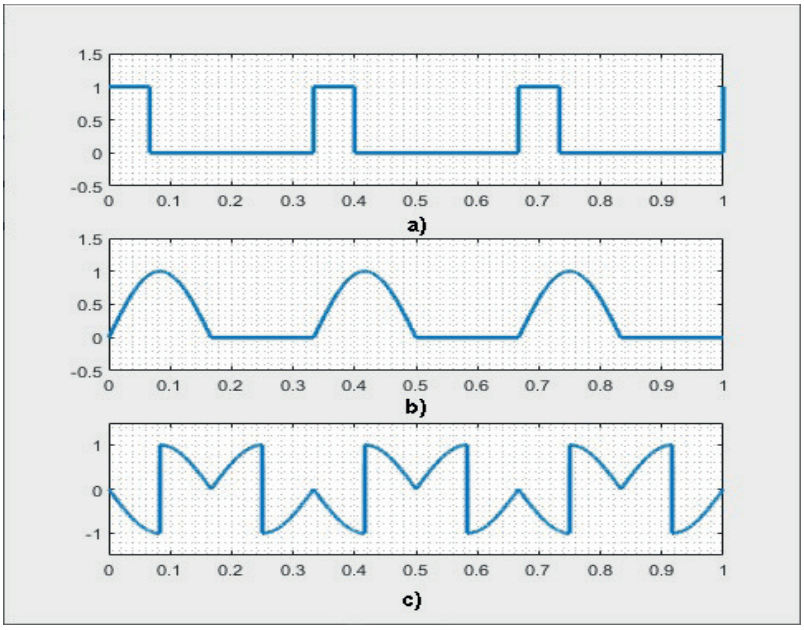


Figura 5. Representación temporal de las señales periódicas utilizadas. a) Pulso. b) Arbitraria 1 y c) Arbitraria 2.

De manera similar, la señal periódica arbitraria 2 fue generada a partir de una onda sinusoidal en la variable "y1" y con frecuencia, amplitud y nivel Off Set en particular; luego, dicha variable se multiplicó con una onda cuadrada desfasada en π radianes y con frecuencia igual al doble de la señal sinusoidal, (ver la figura 5.c). Por lo cual, así como fueron generadas las señales periódicas arbitrarias 1 y 2, también es posible generar cualquier otra señal periódica arbitraria a partir de la combinación de los comandos SQUARE, SAWTOOTH y SIN, y adicionalmente algún artificio matemático necesario que permita establecer las características particulares de la señal de interés.

DESARROLLO DE LA PROGRAMACIÓN

El código de programación se realizó a través del editor del entorno App Designer del Matlab, el cual se visualiza cuando se elige la opción "Code View" hallada sobre la ventana central de Diseño. Asimismo, es importante aclarar que este entorno genera de manera automática un código de programación relacionado a los componentes que se van añadiendo en la interfaz gráfica diseñada. Esta generación se realiza con la característica de programación orientada a objetos, de manera ordenada, formando clases y con propiedades, métodos y funciones específicas; de la misma manera, el código de programación generado en automático no es posible editarlo. Por otro lado, a través de la función CALLBACKS es posible añadir el código de programación de interés utilizando condicionales, lazos repetitivos, funciones propias y hasta comandos de los diferentes Toolbox que tenga instalado el software Matlab.

Por lo tanto, para esta sección, se han establecido tres ítems que corresponden a la obtención de los coeficientes de la Serie de Fourier con apoyo de los comandos SYMS, INT y SUBS del Toolbox Symbolic Math [8], la representación gráfica temporal y la frecuencial utilizando el Toolbox Signal Processing [6]

1. Coeficientes de la Serie de Fourier. Desde el punto de vista trigonométrico, son tres los coeficientes de la Serie de Fourier: a_0, a_n y b_n que se obtienen a partir de las expresiones matemáticas formuladas en (1), (2), (3) y (4) [9], [10].

$$a_0 = \frac{1}{T_0}\int_t^{t+T_0} f(t)dt \quad (1)$$

$$a_n = \frac{2}{T_0}\int_t^{t+T_0} f(t)\cos(2\,\pi\,f_0\,n\,t)\,dt \quad (2)$$

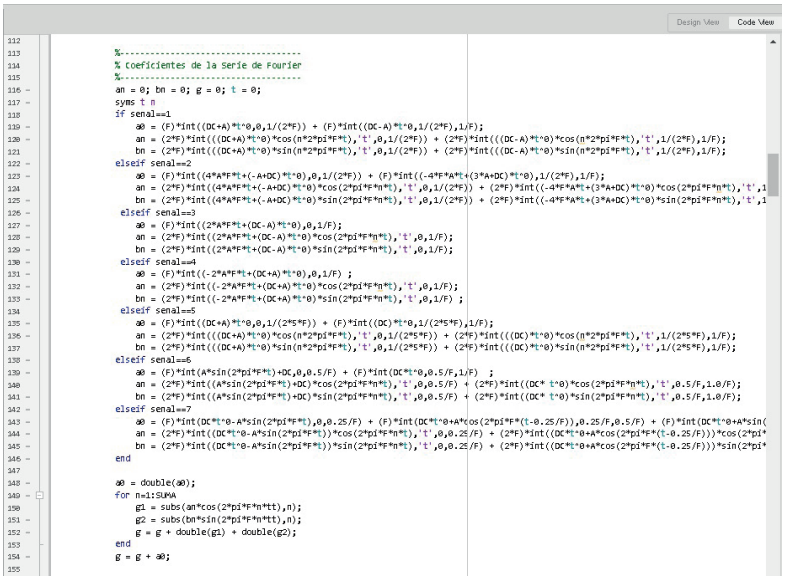
$$b_n = \frac{2}{T_0}\int_t^{t+T_0} f(t)\,\mathrm{sen}(2\,\pi\,f_0\,n\,t)\,dt \quad (3)$$

$$f(t) = a_0 + \sum_{n=1}^{\infty}\left(a_n\cos(2\,\pi\,f_0\,n\,t) + b_n\,\mathrm{sen}(2\,\pi\,f_0\,n\,t)\right) \quad (4)$$

De la expresión (4), "f(t)" representa a la señal periódica con frecuencia fundamental "f_o", y periodo "T_o". Por otro lado, "n" es un número entero que determina la cantidad de componentes sinusoidales que participan en la representación de la Serie de Fourier.

De esta manera, para determinar los tres coeficientes de la Serie de Fourier en el Maltab, se procedió con la creación de las dos variables literales "t" y "n" con apoyo del comando SYMS; la primera, correspondiente al tiempo y la segunda a la cantidad de componentes sinusoidales elegidas a través del componente SPINNER, de la interfaz gráfica diseñada. Luego, se determinaron las integrales para los tres coeficientes con respecto a la variable tiempo "t", utilizando el comando INT del Toolbox Symbolic Math. A su vez, este comando recibió como límites de integración el "0" y el valor del periodo de la señal, que en algunos casos se optó por seccionarlo porque la señal asumía comportamientos diferentes en el tiempo tal como una función constante, lineal o no lineal. Asimismo, los tres coeficientes fueron determinados, pero aún en función de la variable literal "n"; por ello, se tuvo que continuar con el código de programación utilizando el comando SUBS. Este comando permitió evaluar cada uno de los coeficientes con la variable "n" y a través de una suma acumulativa. Finalmente, con la suma acumulativa a través del uso del FOR y la transformación del tipo de variable con el comando DOUBLE, se logró obtener la variable "g" que contiene a la señal periódica generada a partir de los coeficientes de la Serie de Fourier. Seguidamente, en la figura 6, se visualiza toda esta explicación a nivel de código de programación, donde también se observa que en algunas señales periódicas se tuvo que extender el cálculo y evaluación de las integrales simbólicas, mientras que en otros casos este procedimiento fue más rápido y directo.

2. Representación gráfica temporal. La representación gráfica temporal consistió en mostrar dos gráficos de la misma señal periódica, y ambas en función a la variable tiempo, tan igual como se optó por representar en [11]. De esta manera, el primer gráfico correspondió a la representación teórica de la señal periódica; es decir, el resultado del código de programación implementado en la función "GRÁFICA SEÑAL" y que a la vez es visualizado en la figura 3. Y, el segundo gráfico correspondió a la representación aproximada de la señal periódica; es decir, el resultado del código de programación implementado y visualizado en la figura 6.

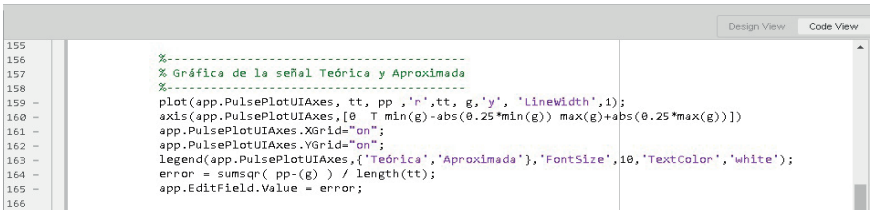


```
%-----------------------------------
% Coeficientes de la Serie de Fourier
%-----------------------------------
an = 0; bn = 0; g = 0; t = 0;
syms t n
if senal==1
    a0 = (F)*int((DC+A)*t^0,0,1/(2*F)) + (F)*int((DC-A)*t^0,1/(2*F),1/F);
    an = (2*F)*int(((DC+A)*t^0)*cos(n*2*pi*F*t),'t',0,1/(2*F)) + (2*F)*int(((DC-A)*t^0)*cos(n*2*pi*F*t),'t',1/(2*F),1/F);
    bn = (2*F)*int(((DC+A)*t^0)*sin(n*2*pi*F*t),'t',0,1/(2*F)) + (2*F)*int(((DC-A)*t^0)*sin(n*2*pi*F*t),'t',1/(2*F),1/F);
elseif senal==2
    a0 = (F)*int((4*A*F*t+(-A+DC)*t^0),0,1/(2*F)) + (F)*int((-4*F*A*t+(3*A+DC)*t^0),1/(2*F),1/F);
    an = (2*F)*int((4*A*F*t+(-A+DC)*t^0)*cos(2*pi*F*n*t),'t',0,1/(2*F)) + (2*F)*int((-4*F*A*t+(3*A+DC)*t^0)*cos(2*pi*F*n*t),'t',1
    bn = (2*F)*int((4*A*F*t+(-A+DC)*t^0)*sin(2*pi*F*n*t),'t',0,1/(2*F)) + (2*F)*int((-4*F*A*t+(3*A+DC)*t^0)*sin(2*pi*F*n*t),'t',1
 elseif senal==3
    a0 = (F)*int((2*A*F*t+(DC-A)*t^0),0,1/F);
    an = (2*F)*int((2*A*F*t+(DC-A)*t^0)*cos(2*pi*F*n*t),'t',0,1/F);
    bn = (2*F)*int((2*A*F*t+(DC-A)*t^0)*sin(2*pi*F*n*t),'t',0,1/F);
 elseif senal==4
    a0 = (F)*int((-2*A*F*t+(DC+A)*t^0),0,1/F) ;
    an = (2*F)*int((-2*A*F*t+(DC+A)*t^0)*cos(2*pi*F*n*t),'t',0,1/F);
    bn = (2*F)*int((-2*A*F*t+(DC+A)*t^0)*sin(2*pi*F*n*t),'t',0,1/F) ;
 elseif senal==5
    a0 = (F)*int((DC+A)*t^0,0,1/(2*5*F)) + (F)*int((DC)*t^0,1/(2*5*F),1/F);
    an = (2*F)*int(((DC+A)*t^0)*cos(n*2*pi*F*t),'t',0,1/(2*5*F)) + (2*F)*int(((DC)*t^0)*cos(n*2*pi*F*t),'t',1/(2*5*F),1/F);
    bn = (2*F)*int(((DC+A)*t^0)*sin(n*2*pi*F*t),'t',0,1/(2*5*F)) + (2*F)*int(((DC)*t^0)*sin(n*2*pi*F*t),'t',1/(2*5*F),1/F);
elseif senal==6
    a0 = (F)*int(A*sin(2*pi*F*t)+DC,0,0.5/F) + (F)*int(DC*t^0,0.5/F,1/F)  ;
    an = (2*F)*int((A*sin(2*pi*F*t)+DC)*cos(2*pi*F*n*t),'t',0,0.5/F) + (2*F)*int((DC* t^0)*cos(2*pi*F*n*t),'t',0.5/F,1.0/F);
    bn = (2*F)*int((A*sin(2*pi*F*t)+DC)*sin(2*pi*F*n*t),'t',0,0.5/F) + (2*F)*int((DC* t^0)*sin(2*pi*F*n*t),'t',0.5/F,1.0/F);
elseif senal==7
    a0 = (F)*int(DC*t^0-A*sin(2*pi*F*t),0,0.25/F) + (F)*int(DC*t^0+A*cos(2*pi*F*(t-0.25/F)),0.25/F,0.5/F) + (F)*int(DC*t^0+A*sin(
    an = (2*F)*int((DC*t^0-A*sin(2*pi*F*t))*cos(2*pi*F*n*t),'t',0,0.25/F) + (2*F)*int((DC*t^0+A*cos(2*pi*F*(t-0.25/F)))*cos(2*pi*
    bn = (2*F)*int((DC*t^0-A*sin(2*pi*F*t))*sin(2*pi*F*n*t),'t',0,0.25/F) + (2*F)*int((DC*t^0+A*cos(2*pi*F*(t-0.25/F)))*sin(2*pi*
end

a0 = double(a0);
for n=1:SUMA
    g1 = subs(an*cos(2*pi*F*n*tt),n);
    g2 = subs(bn*sin(2*pi*F*n*tt),n);
    g = g + double(g1) + double(g2);
end
g = g + a0;
```

Figura 6. Código de programación para el cálculo de los coeficientes de la Serie de Fourier utilizando los comandos SYMS, INT y SUBS del Toolbox Symbolic Math.

A continuación, en la figura 7, se muestra el código de programación desarrollado en el editor del App Designer del Matlab, para representar en función al tiempo tanto la señal periódica teórica definida por la variable "pp", así como la aproximada que fue definida por la variable "g". Asimismo, para dicha representación, se utilizó la variable temporal "tt" la cual se obtuvo de la función "GRÁFICA SEÑAL". De igual forma, la línea de programación 62 de dicha figura muestra el uso del comando SUMSQR para determinar el error medio cuadrático, y luego su visualización en un EditField de tipo numérico.



```
%-------------------------------------------
% Gráfica de la señal Teórica y Aproximada
%-------------------------------------------
plot(app.PulsePlotUIAxes, tt, pp ,'r',tt, g,'y', 'LineWidth',1);
axis(app.PulsePlotUIAxes,[0  T min(g)-abs(0.25*min(g)) max(g)+abs(0.25*max(g))])
app.PulsePlotUIAxes.XGrid="on";
app.PulsePlotUIAxes.YGrid="on";
legend(app.PulsePlotUIAxes,{'Teórica','Aproximada'},'FontSize',10,'TextColor','white');
error = sumsqr( pp-(g) ) / length(tt);
app.EditField.Value = error;
```

Figura 7. Código de programación para la representación gráfica temporal.

3. Representación gráfica frecuencial. La representación gráfica frecuencial consistió en mostrar un gráfico en el dominio de la frecuencia; para ello, se utilizaron los comandos "FFT" y "FFTSHIFT" para determinar la Transformada Rápida de Fourier (FFT) de la señal aproximada, y a su vez realizar un desplazamiento de la componente de frecuencia de 0 Hertz hacia el centro del gráfico. De esta manera, si la señal aproximada solo estaba constituida de 4 componentes sinusoidales, el resultado del cálculo del módulo de la FFT solamente se

limita a mostrar tal resultado. A continuación, en la figura 8, se muestra el código de programación desarrollado en el editor del App Designer del Matlab para representar en función a la frecuencia la señal periódica aproximada. Asimismo, para dicha representación, también se utilizó la variable tipo vector "ff" la cual fue obtenida del comando LINSPACE.

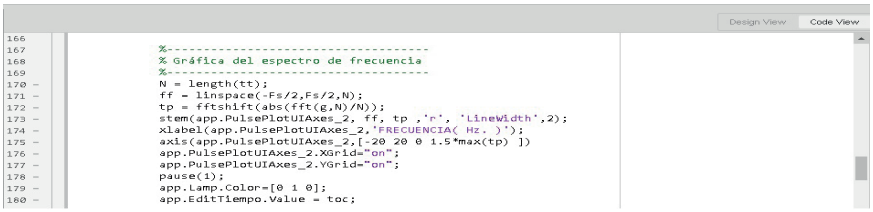


Figura 8. Código de programación para la representación gráfica frecuencial.

RESULTADOS

En cuanto a los resultados alcanzados, se muestran cuatro casos los cuales describen situaciones con amplitudes, frecuencias, niveles DC y tiempos diferentes.

Primer Resultado. Muestra la representación por defecto de la Interfaz Gráfica Desarrollada y Diseñada en el App Designer. Esta corresponde a la señal cuadrada con amplitud de ±1 voltio, frecuencia de 3 Hz, 0 voltios de nivel DC, con una representación de 0 a 1 segundo, y con 4 componentes sinusoidales para la representación de la Serie de Fourier. Por lo cual, el error medio cuadrático obtenido fue igual a 0.099458, mientras que el tiempo de ejecución de los cálculos alcanzó los 26.47 segundos. Ver la figura 9.

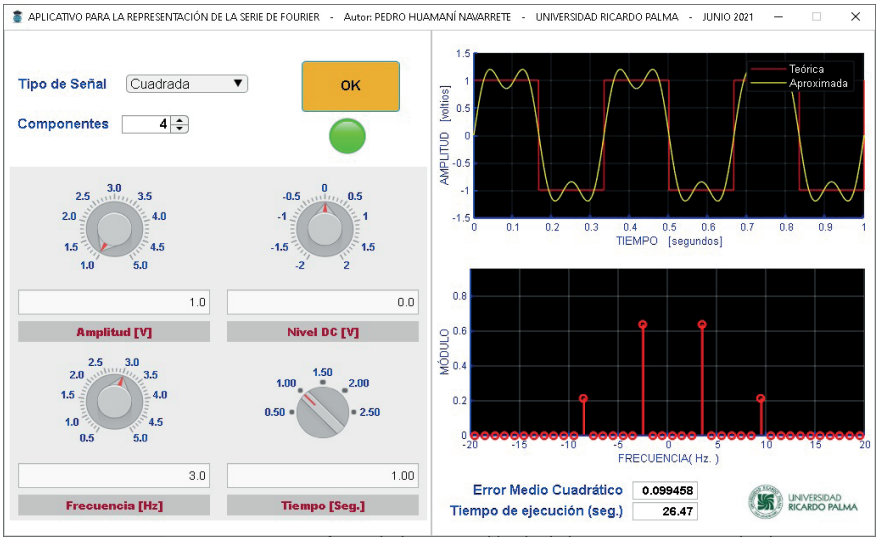


Figura 9. Interfaz gráfica para representar la señal cuadrada.

Segundo Resultado. Corresponde a la señal triangular con amplitud de ±2 voltios, frecuencia de 2 Hz, 1 voltio de nivel DC, con una representación de 0 a 1.5 segundos, y con 6 componentes sinusoidales para la representación de la Serie de Fourier. De esta manera, el error medio cuadrático obtenido fue igual a 0.001005, y el tiempo de ejecución de cálculo alcanzó los 411.80 segundos. Ver la figura 10.

Tercer Resultado. Corresponde a la señal arbitraria 1 con amplitud de ±1.5 voltios, frecuencia de 5 Hz, -1 voltio de nivel DC, con una representación de 0 a 0.5 segundos, y con 8 componentes sinusoidales para la representación de la Serie de Fourier. De esta manera, el error medio cuadrático obtenido fue igual a 0.000107, y el tiempo de ejecución de cálculo alcanzó los 215.94 segundos. Ver la figura 11.

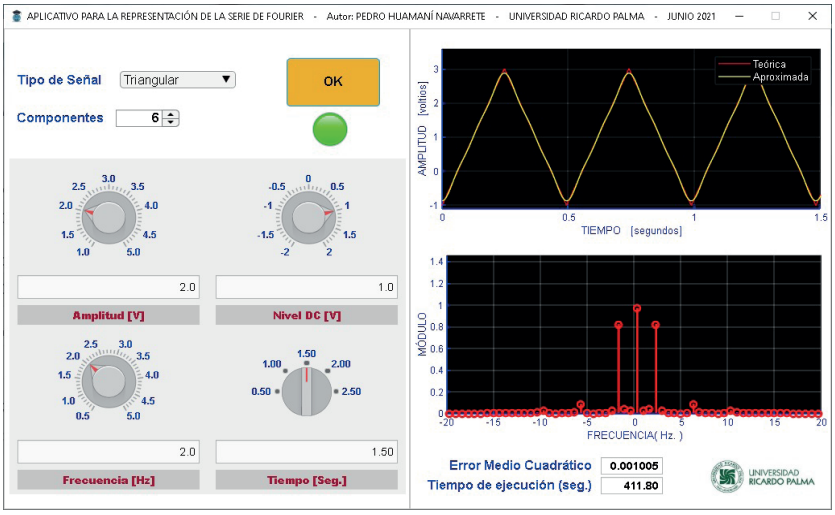


Figura 10. Interfaz gráfica para representar la señal triangular.

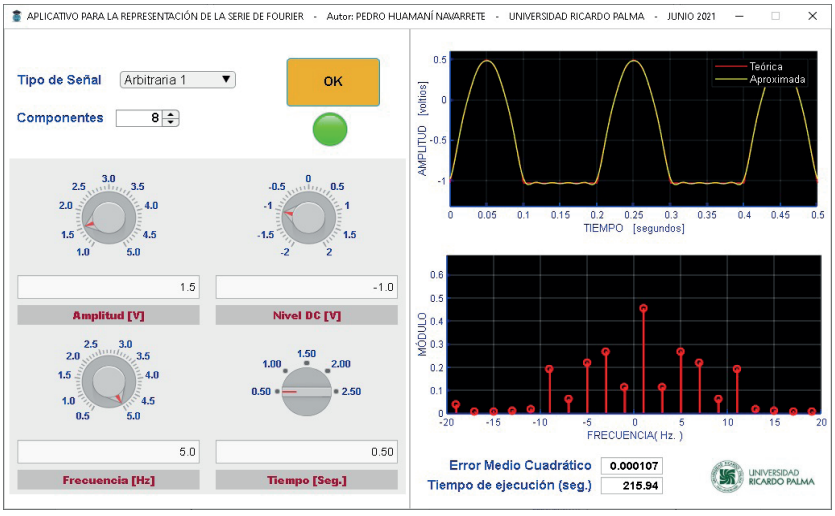


Figura 11. Interfaz gráfica para representar la señal arbitraria 1.

Cuarto Resultado. Corresponde a la señal arbitraria 2 con amplitud de ±1 voltio, frecuencia de 4 Hz, 2 voltios de nivel DC, con una representación de 0 a 1 segundo, y con 12 componentes sinusoidales para la representación de la Serie de Fourier. De esta manera, el error medio cuadrático obtenido fue igual a 0.033319, y el tiempo de ejecución de cálculo alcanzó los 1475.77 segundos. Ver la figura 12.

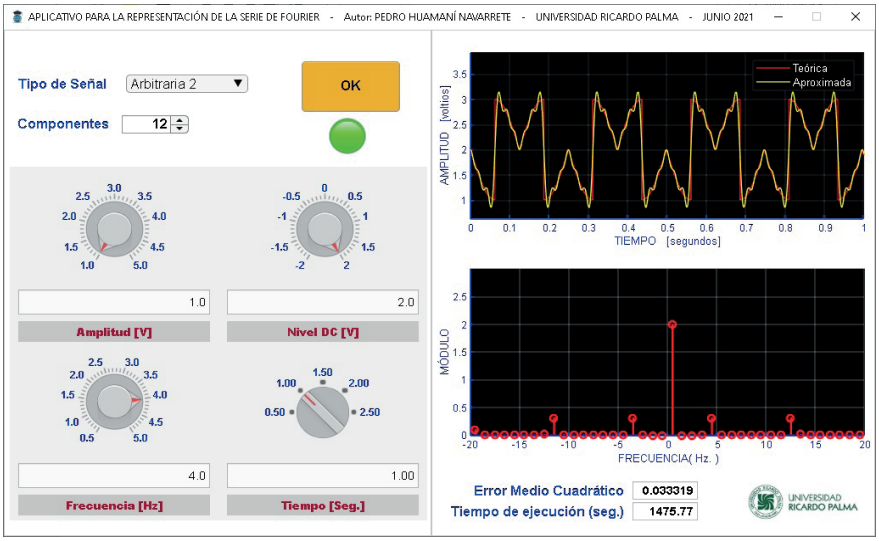


Figura 12. Interfaz gráfica para representar la señal arbitraria 2.

CONCLUSIONES

La implementación de la interfaz gráfica de usuario utilizando el App Designer permitió mostrar la representación temporal y frecuencial de una señal periódica, a partir de una descomposición de sus componentes sinusoidales. Asimismo, se mostró que el código de programación desarrollado con apoyo de los comandos de los Toolbox Signal Processing y Symbolic Math, no representaron complejidad alguna en su implementación; por lo contrario, una flexibilidad para implementar otros tipos de señales periódicas.

Así como también, se demostró que para representar la señal triangular no se requirió de un elevado número de componentes sinusoidales, por lo que el error medio cuadrático fue muy pequeño. Por otro lado, la representación de la señal arbitraria 2 si requirió de un elevado número de componentes sinusoidales y así disminuir el alto error medio cuadrático logrado. A ello, debe sumarse el alto esfuerzo computacional requerido para la obtención de los coeficientes de la Serie de Fourier; es decir, a mayor número de componentes sinusoidales utilizados, el tiempo de ejecución a través de una Desktop con procesador Intel Core i5 de 3 GHz y 8 GB de RAM superó los 20 minutos. Y, para lograr la medición de tales tiempos se optó por emplear las funciones TIC y TOC del Matlab en la línea de código, tal como se observa en la figura 8.

Por lo tanto, con la implementación de esta interfaz gráfica, se facilitó la aplicación de la Serie de Fourier para complementar el proceso de enseñanza-aprendizaje en un tema de la asignatura de Telecomunicaciones I, de la Carrera de Ingeniería Electrónica de la Universidad Ricardo Palma. Sin embargo, este desarrollo puede ampliarse de tal forma que muestre mayores opciones y características tal como en la figura 12.

REFERENCIAS BIBLIOGRÁFICAS

D. Hou, N. Wang, P. Li, G. Fan, Q. Zhao, H. Li, Z. Zhao, Y, Zhang. B. Han, Teaching Research of Signal and System Based on MATLAB, 2019 IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), October 11-13, Chongqing, China.

H. Ping, H. Weikun, S. Qingyan and H. Yan, An educational tool design for the course of signal processing based on Matlab GUI, 2014 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE), December 8-10, Wellington, New Zealand.

Y. Zhou and J. Dong, The Application of MATLAB Platform in the Teaching of Digital Signal Processing, 2010 International Conference on E-Business and E-Government, May 7-9, Guangzhou, China.

Mathworks (2021, mayo). Centro de Ayuda. Desarrollar apps mediante App Designer [Online]. Disponible: https://la.mathworks.com/help/matlab/app-designer.html

Matlab (2021, mayo). Cree apps web y de escritorio en Matlab [Online]. Disponible: https://la.mathworks.com/products/matlab/app-designer.html

Matlab (2021, mayo). Signal Processing Toolbox. Realice procesamiento y análisis de señales [Online]. Disponible: https://la.mathworks.com/products/signal.html

D. Báez-López y D. Báez. Matlab Handbook with Applications to Mathematics, Science, Engineering, and Finance. Florida: CRC Press, 2019.

Matlab (2021, mayo). Symbolic Math Toolbox. Realice cálculos de matemáticas simbólicas [Online]. Disponible: https://la.mathworks.com/products/symbolic.html

Matlab. Symbolic Math Toolbox. User's Guide. R2020a. USA: The MathWorks, Inc, 2020.

B. Lathi and R. Greeen. Linear Systems and Signals. Third Edition. New York: Oxford University Press, 2018.

B. Lathi. Introducción a la Teoría y Sistemas de Comunicación. México: Limusa, 2001.

M. Reis, S. Salviano, S. Cardeal, R. Morais, E. Peres and P. Ferreira, Teaching of Fourier series expansions in undergraduate education, IEEE Global Engineering Education Conference (EDUCON 2013), March 13-15, Berlin, Germany.

A. Kumar, A. Awasthi, O. Salari, A. Laha, A. Mathew and P. Jain, A Time-Domain Based APP Designer For Resonant Converters With GUI Features, 2020 IEEE Applied Power Electronics Conference and Exposition (APEC), March 15-19, New Orleans, USA.