Práctica 3

Almeraya, Kimberly.,Tinoco, Sergio.,De Los Ríos, Flavio.
Instituto Politécnico Nacional,Escuela Superior de Cómputo.
Procesamiento de Señales

18 de Noviembre de 2022

En esta práctica, se muestran 10 señales vistas anteriormente en la práctica 0 de manera gráfica con la ayuda de la Interfaz Gráfica de Usuario (GUI, por sus siglas en inglés) diseñada a través de la aplicación de Matlab denominada App Designer; en cada señal ya es posible modificar el intervalo de la variable independiente mostrando, así como su frecuencia de muestreo. Cabe señalar que ahora en las señales deterministas el usuario ya puede modificar su magnitud y su periodo. Ahora es posible calcular la energía y la potencia de las señales, además de poder elegir la paridad con la que estás se muestran. Asimismo, ya es posible grabar una señal de voz (como onceava señal), o bien, elegir una grabada previamente y manipularla cambiando el color, la paridad y calculando su energía y potencia.

Palabras clave: Energía, potencia, paridad.

I. INTRODUCCIÓN

La práctica se realizó con la finalidad de grabar una señal de audio usando la aplicación App Designer que permite manipular el intervalo de la variable independiente, así como su frecuencia de muestreo, calcular la energía y la potencia de esta, así como cambiar su paridad con la que se muestra en el gráfico.

App Designer es un entorno de desarrollo interactivo para diseñar una aplicación y programar su comportamiento. Proporciona una versión totalmente integrada del editor de MATLAB y un gran conjunto de componentes interactivos de la IU.

Se tiene en cuenta que una señal puede ser intangible, sin embargo una señal es una onda la cual indica o señala la existencia o el estado de un hecho u objeto. Entonces, se vuelve tangible cuando se ilustra gráficamente. Dentro de las señales se ven involucradas sus características, como las magnitudes, esto quiere decir que se mide o se mantiene un registro de “algo que está ocurriendo” o que “ocurrió cuando…” , del modo y frecuencia se presentan. La frecuencia de muestreo es la cantidad de muestras que se toman por unidad de tiempo para convertir la señal analógica en señal digital.

Para comprender mejor el objetivo de la práctica, se definen los conceptos principales de la app, por ejemplo, la potencia, se dice de una señal por unidades de tiempo, y su unidad son los watts, en cambio, la energía es atemporal y se mide en Joules. Por lo dicho anteriormente, la potencia y la energía de una señal están relacionadas entre sí.

La paridad, se refiere a los siguientes parámetros:

Señal par
Una señal x(t) ó x[n] es par si se «refleja» en el eje vertical u ordenadas.

$$x(t) = x(-t)x(t)=x(-t)$$
$$x[n] = x[-n]x[n]=x[-n]$$

La señal tiene los mismos valores para el lado positivo o negativo de |t|.

Es como si se aplicara el valor absoluto de t antes de hacerlo en la ecuación.

Señal Impar
Una señal x(t) ó x[n] es impar si se cumple que:

$$x(t) = -x(-t)x(t)=-x(-t)$$
$$x[n] = -x[-n]x[n]=-x[-n]$$

Una señal impar debe ser necesariamente 0 en t=0 o n=0.

O sea que son dos conceptos definidos de parámetros que deben cumplirse de acuerdo con la función de la señal.

No es posible graficar sin antes definir a las variables, pues estas permiten saber a quienes o qué está implicado. Una vez definidas las variables, se puede comenzar a usar App Designer.

Con el software Matlab y sus herramientas para visualizar gráficos, se pueden simular las diferentes señales, además de poder grabar audios obteniendo una señal, permite calcular su energía y potencia así como modificar su paridad.

OBJETIVO Y PLANTEAMIENTO DEL PROBLEMA

El objetivo de esta práctica fue observar en los cálculos de la energía y la potencia de una señal. Además de captar señales de sonido tomadas del entorno a través de una grabación, y poder visualizarlas en forma de señal.

En esta práctica se afrontaron diversas cuestiones como lo son el cálculo de energía y potencias de las señales, se debió crear una aplicación GUI en App Designer de Matlab, en la que se grafique las señales, agregar las funciones que permitan modificar el intervalo de la variable independiente mostrada, así como su frecuencia de muestreo, para el caso de señales deterministas, deberá ser capataz de modificar su magnitud y periodo. Además de poder cambiar su paridad y calcular las energías y potencias de dichas señales.

II. DESARROLLO EXPERIMENTAL

Para la aplicación creada en App Designer se usó el paradigma orientado a objetos. Esto debido a que así es como trabaja la herramienta App Designer, por medio de la creación de atributos y métodos y por ende, la programación de los mismos.

La programación orientada a objetos se basa en el concepto de crear un modelo del problema de destino en sus programas. La programación orientada a objetos disminuye los errores y promociona la reutilización del código.

En este caso, la aplicación se refiere a una GUI. La interfaz gráfica de usuario (GUI)

es un medio de interacción visual entre los usuarios y los dispositivos electrónicos.

Por ello, es usado en la mayoría de sitios web, programas para computadora, aplicaciones móviles, sistemas operativos, entre otros tipos de software existentes.

Este tipo de interacción posibilita el ofrecimiento de opciones WYSIWYG (what you see is what you get) dentro de los programas de software y permite manipular los elementos gráficos.

Para comenzar el desarrollo de la aplicación se deben definir los objetos, atributos y sus métodos.

Se definió a 'p' , 'a' y 'c' como variables globales auxiliares para gestionar el parámetro de frecuencia, amplitud y el color que desea cambiar el usuario, respectivamente. En la siguiente captura de pantalla, se definen las variables globales:

```
properties (Access = private)
    p       %Parametro para frecuencia
    c='r'       %Color de la linea en el plot
    a       %Parametro para la amplitud
end
```

Captura de pantalla 1. Elaboración propia. Definición de variables auxiliares.

La startupFcn inicializa el valor de la variable 'p' y 'a' con el valor del slider correspondiente. Y de igual manera se implementó la función *ColorListBoxValueChanged* para poder cambiar el color de las señales según la preferencia del usuario, esto para poder distinguir mejor el gráfico, eso se aprecia en la siguiente imagen:

```
% Code that executes after component creation
function startupFcn(app)
    app.p=app.PSlider.Value;
    app.a=app.ASlider.Value;
end
```

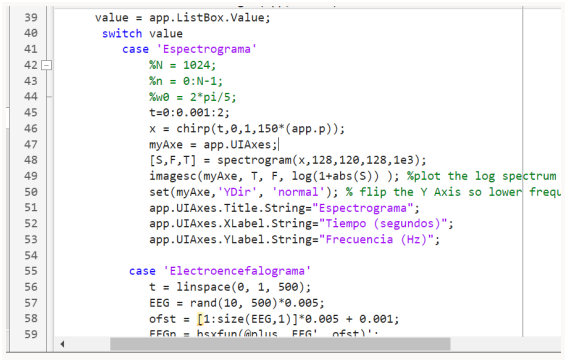
Captura de pantalla 2. Elaboración propia. Definición de startupFcn y función para elegir color.

Luego, usando la función "ListBoxValueChanged" se contienen las 11 señales diferentes en una cajita para poder ser seleccionadas por el usuario. Cabe señalar que dentro de esta función se implementó un switch en el que cada caso es una de las 11 señales diferentes, dentro de ellas podemos encontrar la linea de codigo "app.UIAxes.Xlabel.String="Tiempo (segundos)" la cual le permite al usuario cambiar el tiempo de la señal".

Otra de las líneas del código agregadas fue la de "app.UIAxes.Ylabel.String="Frecuencia (Hz)" la cúal permite al usuario cambiar la frecuencia con la que se muestra la señal.

Todos los casos tienen la misma estructura, cada uno tiene su propio 'case' y sigue la misma estructura. Cuando el usuario cambie el color o el parámetro de la variable independiente, se volverán a ejecutar las operaciones dependiendo del 'case' que esté seleccionado, en este caso de la señal que se haya elegido.

Esto se observa en la imagen a continuación.

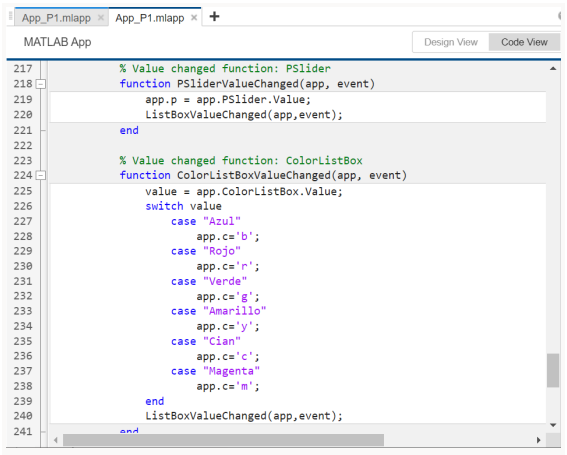


```
    value = app.ListBox.Value;
    switch value
        case 'Espectrograma'
            %N = 1024;
            %n = 0:N-1;
            %w0 = 2*pi/5;
            t=0:0.001:2;
            x = chirp(t,0,1,150*(app.p));
            myAxe = app.UIAxes;
            [S,F,T] = spectrogram(x,128,120,128,1e3);
            imagesc(myAxe, T, F, log(1+abs(S)) ); %plot the log spectrum
            set(myAxe,'YDir', 'normal'); % flip the Y Axis so lower frequ
            app.UIAxes.Title.String="Espectrograma";
            app.UIAxes.XLabel.String="Tiempo (segundos)";
            app.UIAxes.YLabel.String="Frecuencia (Hz)";

        case 'Electroencefalograma'
            t = linspace(0, 1, 500);
            EEG = rand(10, 500)*0.005;
            ofst = [1:size(EEG,1)]*0.005 + 0.001;
```

Captura de pantalla 3. Elaboración propia. Definición de Switch para las 10 señales y definición de frecuencia.

Cuando el usuario cambie el color de la línea de la señal, según este *switch case*, se asignará el valor correspondiente a la variable auxiliar 'c' de color, he aquí la la función del cambio de color:

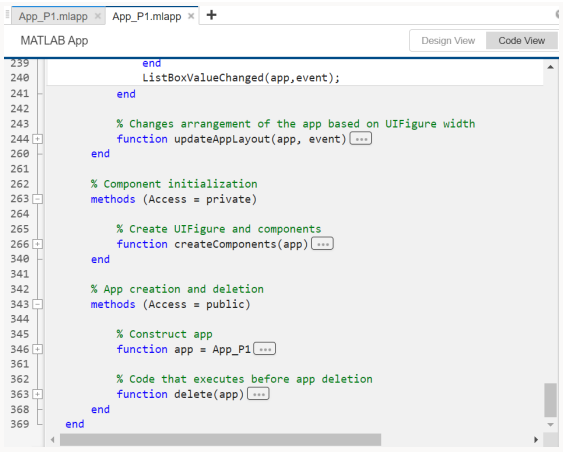


```
App_P1.mlapp   App_P1.mlapp
MATLAB App                                   Design View   Code View
        % Value changed function: PSlider
        function PSliderValueChanged(app, event)
            app.p = app.PSlider.Value;
            ListBoxValueChanged(app,event);
        end

        % Value changed function: ColorListBox
        function ColorListBoxValueChanged(app, event)
            value = app.ColorListBox.Value;
            switch value
                case "Azul"
                    app.c='b';
                case "Rojo"
                    app.c='r';
                case "Verde"
                    app.c='g';
                case "Amarillo"
                    app.c='y';
                case "Cian"
                    app.c='c';
                case "Magenta"
                    app.c='m';
            end
            ListBoxValueChanged(app,event);
        end
```

Captura de pantalla 4. Elaboración propia. Definición de función para elegir color (switch cases).

Por último, se muestran las funciones para actualizaciones de los contenedores.



```
App_P1.mlapp   App_P1.mlapp
MATLAB App                                   Design View   Code View
            end
            ListBoxValueChanged(app,event);
        end

        % Changes arrangement of the app based on UIFigure width
        function updateAppLayout(app, event) ...
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app) ...
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = App_P1 ...

        % Code that executes before app deletion
        function delete(app) ...
    end
end
```

Captura de pantalla 5. Elaboración propia. Definición de updateAppLayout.

Se agregaron dos nuevas variables, una para la paridad y otra para la energía como se muestra en la siguiente imagen:

```
paridad    %Valor de la periodicidad (periodida o aperiodica)
energia      %Valor de la energia de la señal
```

Captura de pantalla 6. Variables "paridad" y "energía".

Se agregó una nueva función para calcular la energía y la potencia de las señales como se muestra en la siguiente imagen:

```
function CalcularEyP(app)
    syms t;
    value = app.ListBox.Value;
    n = 0:7;
    sympref('HeavisideAtOrigin',1); %Cambiar el valor
```

```
case 'Grabar Audio'
    x1=app.x;
```

```
    app.EnergiaEditField.Value=string(Ex1);
    app.PotenciaEditField.Value=string(Px1);
end
```

Captura de pantalla 8. Función Calcular Energía y Potencia.

Caso completo si el usuario elige grabar un audio:

```
case 'Grabar Audio'


        if(app.defaultButton.Value)
            %Callback para grabar
            fs=30000;  %44100; %f.muestreo
            %n = (0:1/fs:3);
            segs=3;
            senal_salida=audiorecorder(fs,16,1); %Creacion del objeto de grabacion
            msgbox('Empezando Grabacion','Grabadora');%Mensaje de información
            recordblocking(senal_salida,segs); %Grabacion del sonido
            msgbox('Terminando grabación');%Mensaje de informacion
            %Paso los valores del objeto a una señal
            app.x=getaudiodata(senal_salida,'single');
            %Se graba y se guarda la señal
            %audiowrite("audio.wav",senal_grabada,fs);

           %[y,Fs0] = audioread("audio.wav");
            %plot(app.UIAxes, y, Color='#732BF5');
            app.Fs = fs;
            %app.x = y;

           % app.pl=y;

            %app.ParidadButtonGroup.Enable="on";

            app.ContinuaButton.Value=true;
            titulo="Señal de Audio";
```

Captura de pantalla 9. Caso completo grabar un audio.

Botón paridad para cambiar la paridad de la señal:

```
% Selection changed function: ParidadButtonGroup
function ParidadButtonGroupSelectionChanged(app, event)
    ListBoxValueChanged(app,event);
end
```

Captura de pantalla 10. Botón paridad.

```
% Button pushed function: ReproducirButton
function ReproducirButtonPushed(app, event)
        %Reproducir audio
          sound(app.x,app.Fs)
end
```

Captura de pantalla 11. Botón para reproducir el audio grabado.

III. DISCUSIÓN Y RESULTADOS

La App GUI se ve así inicialmente.

Figura 1. App GUI inicializada.

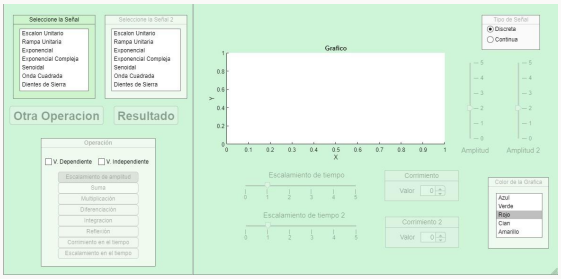


Tabla 1. Primera ejecución

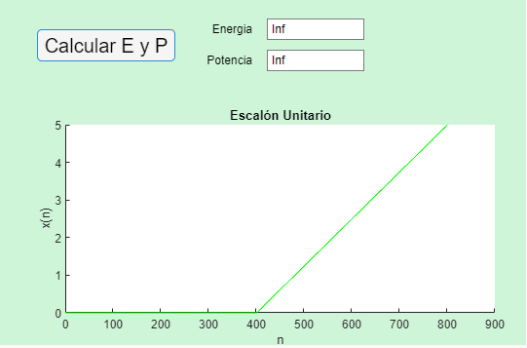
Señal: Rampa

Pantalla 1. Energía y potencia de una señal de rampa.
Descripción: Se muestra un resultado ‘inf’ en cada parámetro.

Tabla 2. Segunda ejecución

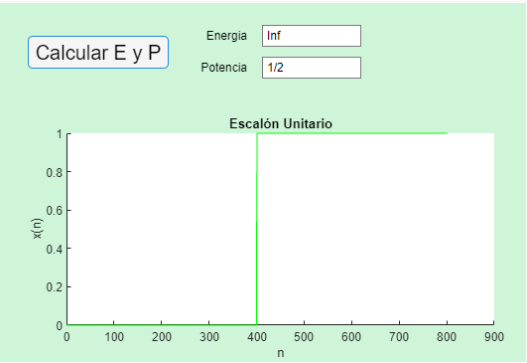
Señal: Escalón unitario

Pantalla 2. Energía y potencia de una señal de escalón unitario.
Descripción: Se muestra un resultado ‘inf’ en energía’ y ‘½’ en potencia.

Tabla 3. Tercera ejecución

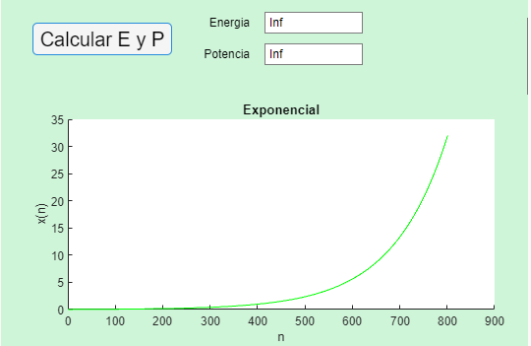
Señal: Exponencial
 Pantalla 3. Energía y potencia de una señal exponencial.
Descripción: Se muestra un resultado ‘inf’ en energía y en potencia.

Tabla 4. Cuarta ejecución

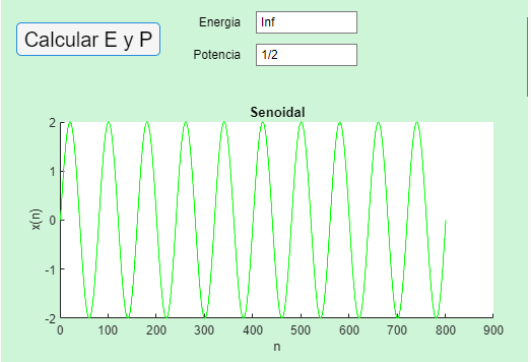
Señal: Senoidal
 Pantalla 4. Energía y potencia de una señal senoidal.
Descripción: Se muestra un resultado ‘inf’ en energía’ y ‘½’ en potencia.

Tabla 5. Quinta ejecución

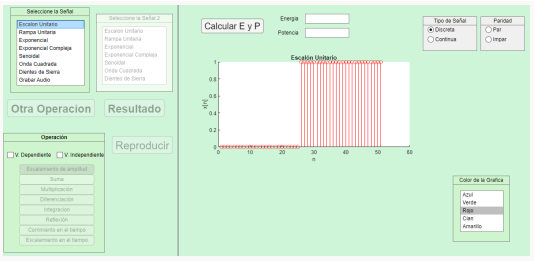
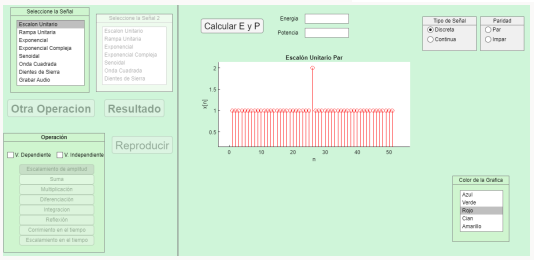
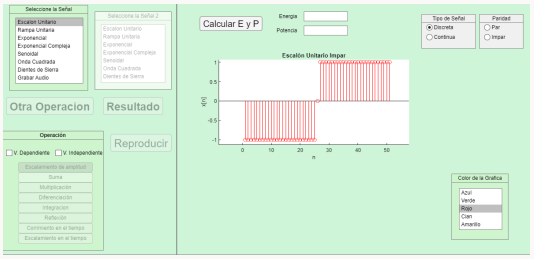
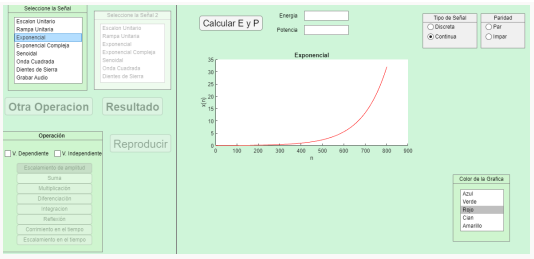
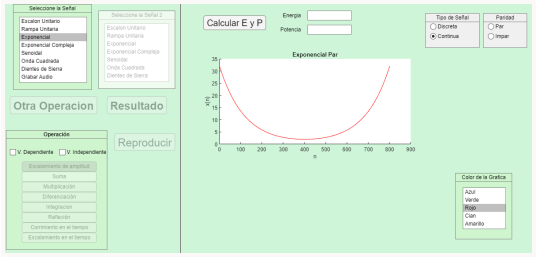
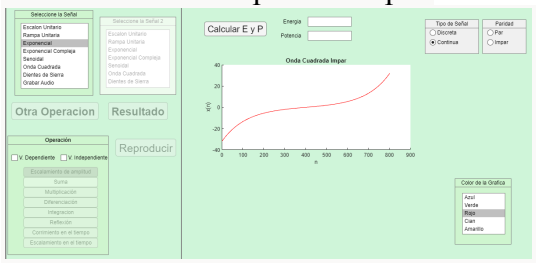
Señal: Escalón unitario
 Pantalla 5. Escalón unitario  Pantalla 6. Escalón unitario par  Pantalla 7. Escalón unitario impar
Descripción: Se eligió el escalón unitario (pantalla 5) y se seleccionó el botón que dice ‘Par’ (pantalla 6), después el botón que dice ‘Impar’ (pantalla 7).

Tabla 6. Sexta ejecución

Señal: Exponencial
 Pantalla 8. Señal exponencial.



Pantalla 9. Señal exponencial par.

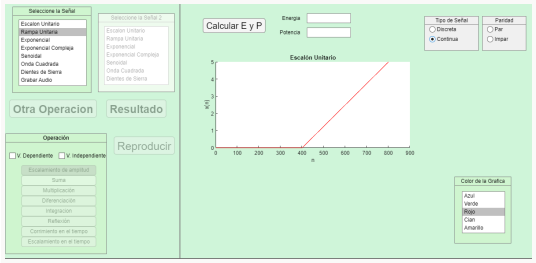


Pantalla 10. Señal exponencial impar.

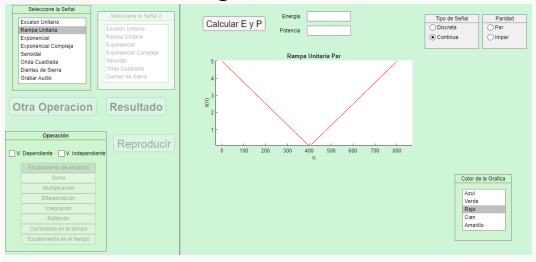
Descripción: Se eligió una señal exponencial (pantalla 8) y se seleccionó el botón que dice ‘Par’ (pantalla 9), después el botón que dice ‘Impar’ (pantalla 10).

Tabla 7. Séptima ejecución

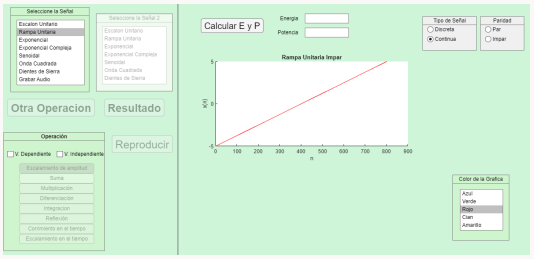
Señal: Rampa



Pantalla 11. Rampa



Pantalla 12. Rampa par

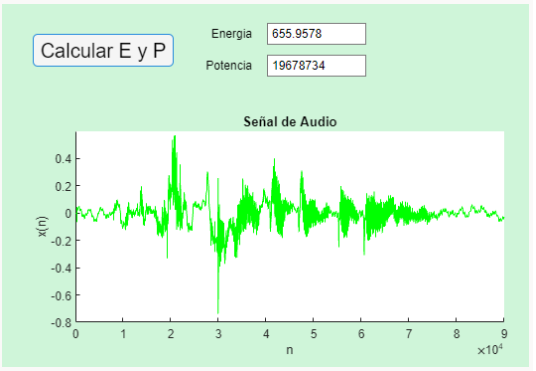


Pantalla 13. Rampa impar

Descripción: Se eligió la señal de rampa (pantalla 11) y se seleccionó el botón que dice ‘Par’ (pantalla 12), después el botón que dice ‘Impar’ (pantalla 13).

Tabla 8. Octava ejecución

Señal: Grabación.

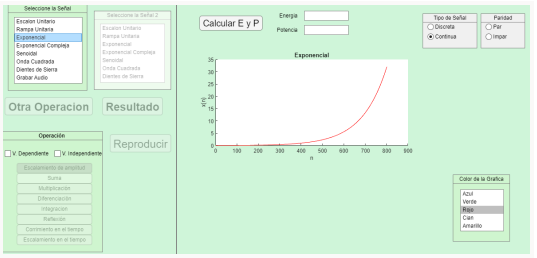


Pantalla 14. Energía y potencia de una señal de grabación.

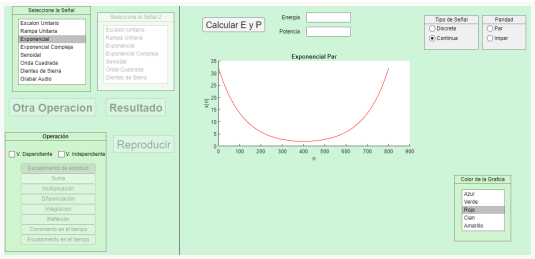
Descripción: Se muestra un resultado ‘655.9578’ en energía y de ‘19678734’ en potencia.

Tabla 9. Novena ejecución

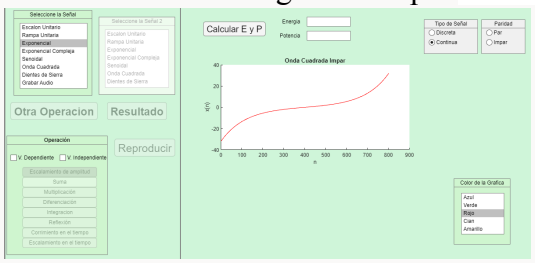
Señal: Grabación



Pantalla 15. Señal de grabación.



Pantalla 16. Señal de grabación par.



Pantalla 17. Señal de grabación impar.

Descripción: Se eligió una señal de grabación (pantalla 15) y se seleccionó el botón que dice 'Par' (pantalla 16), después el botón que dice 'Impar' (pantalla 17).

IV. CONCLUSIÓN

Para concluir, esta práctica fue de ayuda por la visualización de una señal par y/o impar. Así como para imaginar una estimación entre los resultados de las distintas señales básicas que hasta el momento hemos visto. Sin embargo, encontramos muy útil también el hecho de trabajar con distintas fórmulas para el cálculo tanto de la energía como de la potencia. Para comprender las operaciones entre funciones es fundamental poder ver los resultados de una forma tangible. Con ayuda de la App GUI, será mucho más fácil y rápido.

Esta aplicación de medir la energía y potencia de señales, podría funcionar para generar resultados de señales tomadas de una BD, y transformarlas, es decir, aplicarle alguna de estas operaciones y generar nuevos resultados. Así como hacer alguna predicción respecto a señales de audio en este caso.

V. BIBLIOGRAFÍA

Desarrollar apps mediante App Designer - MATLAB & Simulink - MathWorks América Latina. (s. f.). Recuperado 25 de octubre de 2022, de https://la.mathworks.com/help/matlab/app-designer.html

Geek for Geeks. (2022, 15 marzo). Creating Apps Using App Designer in MATLAB. Recuperado 25 de octubre de 2022, de https://www.geeksforgeeks.org/creating-apps-using-app-designer-in-matlab/

Object-Oriented Programming. (s. f.). Recuperado 25 de octubre de 2022, de https://la.mathworks.com/products/matlab/object-oriented-programming.html

Anónimo. (2015). *Espectrograma utilizando la transformada de Fourier de tiempo corto - MATLAB spectrogram - MathWorks América Latina*. MathWorks. Recuperado 6 de septiembre de 2022, de https://la.mathworks.com/help/signal/ref/spectrogram.html

Dr.Vijay Dudhal (2022). EEG ANALYSIS AND CLASSIFICATION (https://www.mathworks.com/matlabcentral/fileexchange/55112-eeg-analysis-and-classification), MATLAB Central File Exchange. Recuperado September 7, 2022.

MathWorks, M. (2020). *Funciones de impulso, escalón y rampa*. MathWorks. https://la.mathworks.com/help/signal/gs/impulse-step-and-ramp-functions.html

Rep. de Señales Componentes Par e Impar. (s. f.). Recuperado de http://www.unet.edu.ve/aula10c/Asenales/Unid01/Rep03.htm