Guía Laboratorio 3
Procesamiento Digital de Señales

Camilo Vásquez, Paula Pérez, Alejandro Escobar, Jhon Lopera y Cristian Ríos

2020-1

- Descargar el informe del laboratorio con el siguiente nombre: *Lab3_PDS_Apellido_Nombre.ipynb*
- Enviar junto con el informe los archivos adicionales generados y descargados. Todo esto debe ir en un archivo comprimido con el siguiente nombre: *Lab3_PDS_Apellido_Nombre.zip*

1. Introducción

En este laboratorio se observará el concepto de correlación, la cual es una medida de similitud entre dos series de tiempo en función del retraso de una respecto a la otra. Esta es definida para dos secuencias en tiempo discreto como:

$$R_{xy}[k] = \sum_{m=-\infty}^{\infty} x[m]\ y[m-k] \tag{1}$$

2. Análisis de pulsos de radar

Una de las aplicaciones clásicas de la correlación se centra en los sistemas de radar aeronáuticos. En estos sistemas, un transmisor envía un pulso electromagnético de forma conocida, el cual es reflejado por alguna aeronave y esta reflexión es captada por un receptor. La señal recibida es una copia retrasada y con ruido del pulso original. Para detectar el pulso original en esta señal, se calcula la correlación entre la señal recibida y el pulso original, allí donde se encuentre el máximo de la señal de correlación será la ubicación del pulso en la reflexión.

Otra aplicación de este mismo concepto es en las comunicaciones digitales, en este caso, un pulso que representa un símbolo (conjunto de bits) pasa por un canal que lo distorsiona. En el receptor se realiza la correlación con la forma original del pulso para encontrar el símbolo adecuado.

Para esta práctica se debe usar la función adjunta en el archivo *utils.py*, esta función nos permitirá simular la distorsión de la señal. No olvide incluirlo en la misma carpeta que tiene el jupyter-notebook de la práctica e importarla.

1. Genere un pulso rectangular de 20 muestras. Tip: Utilice la función de numpy *np.ones(20)*.
2. Calcule y grafique la función de autocorrelación del pulso generado.

```
Rxx=np.correlate(rect_pulse, rect_pulse, mode='full')
tau=np.arange(-len(rect_pulse)+1,len(rect_pulse),1) # vector de retraso
```

   rect_pulse es el vector con el pulso rectangular creado en 1, *Rxx* es la función de autocorrelación resultante y *tau* es el vector de retraso.

   ¿Qué puede concluir de la gráfica obtenida?

3. Utilice la función *delay_noise* incluida en la librería adjunta *'utils.py'* para simular el ruido introducido por el canal, y el desfase de la señal recibida respecto a la original. Grafique el resultado, ¿Es capaz de distinguir el pulso y decir cuanto es su retraso?

```
delayed_pulse=delay_noise(rect_pulse)
```

4. Utilice la función de numpy *np.correlate* para hallar la correlación entre la señal distorsionada y la señal original. Muestre el resultado gráficamente ¿Cuál es el retraso? ¿Qué se puede concluir?

   Tip 1: Para utilizar la función np.correlate utilice el siguiente comando:

```
xcorr=np.correlate(delayed_pulse,rect_pulse)
```

   Tip 2: Puede hallar el retraso con las funciones de numpy *np.where* o *np.argmax*.

3. Análisis de Electroencefalograma (EEG)

A continuación se trabajara con señales electroencefalográficas descrita en:

https://lampx.tugraz.at/~bci/database/001-2014/description.pdf

En los EEGs se capturan señales por medio de sensores puestos sobre el cuero cabelludo, estos sensores miden los cambios de potencial producidos por la sinapsis de las neuronas.

Para la base de datos se utilizaron 22 electrodos para captar señales neuronales, y 3 electrodos oculares para caracterizar los parpadeos, organizados como se muestra en figura 1. Las señales fueron capturadas con una frecuencia de muestreo de 250Hz y filtradas con un filtro pasa-bandas con frecuencias de corte de 0.5 Hz-100 Hz. El registro se hizo cuando el participante estaba en estado de reposo y con los ojos abiertos.

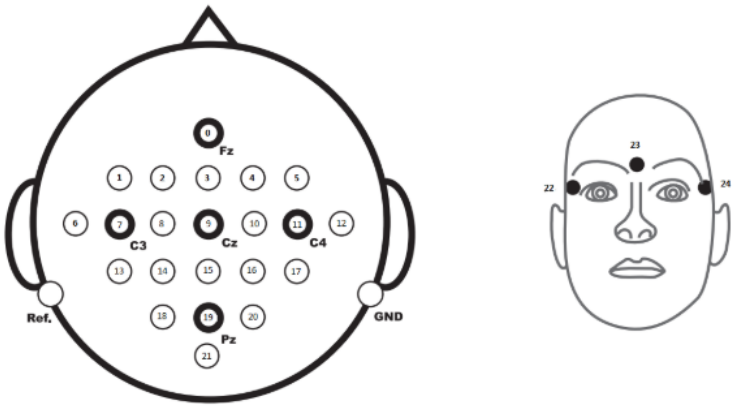

Figura 1: Posición de los electrodos.

NOTA: En esta base de datos los canales se cargan en filas y no por columnas como en muchos casos. Dimensiones del archivo (25, 29683), donde 25 es el numero de canales y 29683 es el numero de muestras.

- Cargue el archivo *eeg.npy* con *numpy.load* , para cada canal normalice en amplitud y elimine el nivel DC. Luego escoja el canal de acuerdo al ultimo numero de su cédula a partir de la siguiente lista:

  1. Canal 21
  3. Canal 4
  5. Canal 9
  7. Canal 14

9. Canal 19

Finalmente grafique el canal elegido con y sin normalización usando subplot para una fácil comparación.

- ¿Qué observa? Describa brevemente la diferencia entre las gráficas.

3.1. Coeficiente de correlación entre canales

1. Ahora calcule el coeficiente de correlación entre el canal que le corresponde y los otros canales. Finalmente, grafique estos coeficientes de correlación (deben ser 25) utilizando la función *plt.stem*.

   Tip: Use la función *np.corrcoef* para calcular los coeficientes.

2. ¿Cómo varia la correlación entre el canal que le corresponde y los demás canales?. Realice un breve análisis de lo observado.

3.2. Función de autocorrelación de un canal

1. Calcule la función de autocorrelación del canal que le corresponde, luego calcule la función de autocorrelación sobre uno de los canales de los ojos (canal 22 o 24), finalmente grafique y compare las dos funciones utilizando subplots. Sugerencia: Haga zoom sobre las señal de autocorrelación para observar mejor las diferencias.

2. ¿Qué observa?. Describa brevemente las gráficas obtenidas.

4. Conclusiones

Realice conclusiones generales sobre la práctica. Recuerde que las conclusiones son parte fundamental de su evaluación en el laboratorio, tómese el tiempo de pensar las conclusiones.