Proyecto 3 - Filtros

Catalina Guerra Fernandez, Angie Tovar Bohorquez, Juan Esteban Campillo

Universidad de Antioquia

Noviembre 8, 2024

---

1. ***Consultar y explicar los dos métodos de reducción de ruido usados en el artículo: Robust LOESS y Non Local Means , mostrar cómo se podrían implementar en Python***

   1.1 Filtro Robust LOESS

El término LOESS hace referencia a locally estimated scatterplot smoothing, el cual es un algoritmo de regresión local que combina la regresión local por mínimos cuadrados con la flexibilidad de la regresión no lineal [1].

En general lo que hace un algoritmo LOESS es suavizar la curva de tendencia de datos aplicando una regresión polinómica a una ventana ponderada de un conjunto de datos, de esta manera se encuentra la curva que mejor se ajusta a los datos de un diagrama de dispersión. El término robust hace referencia a la asignación de pesos a los valores para de esta manera asignar un menor peso a los datos atípicos o que se encuentren más alejados de esta curva de tendencia [2].

En la *Figura 1* se observa como se aplica la curva a un diagrama de dispersión en la cual se hace una regresión polinómica por un conjunto de datos determinados.

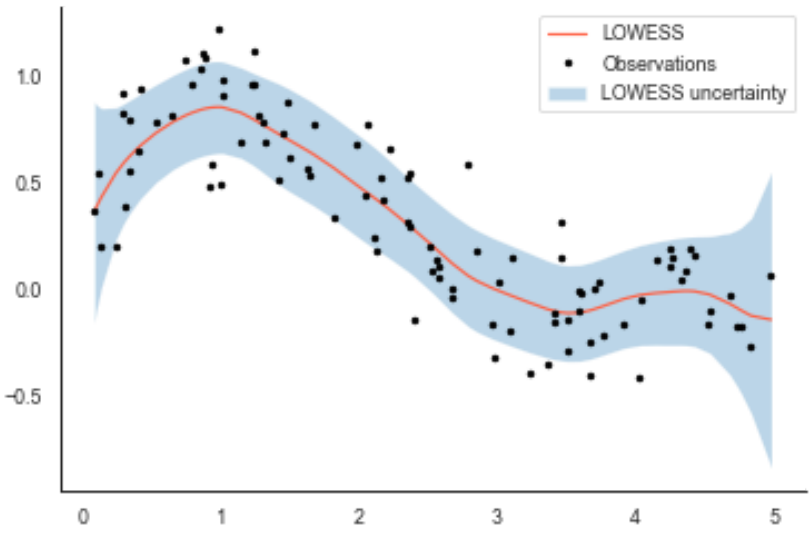


Figura 1. Curva de tendencia ajustada a un diagramad de dispersión mediante LOESS [3] .

Para implementar este método de eliminación de ruido en Python, se puede hacer uso de la librería stats model, se implementa LOESS y para que sea robusto se agrega un ajuste en iteraciones.

```
[ ] from statsmodels.nonparametric.smoothers_lowess import lowess
    robust_loess_result = lowess(y, x, frac=0.2, it=3)
```

Figura 2. Implementación en python de un LOESS robusto, el número de iteraciones determina que tantos outliers van a ser eliminados.

1.2 Non Local Means

Elimina ruido utilizando información de toda la señal, es muy eficaz para preservar detalles y estructuras importantes al filtrar. La eliminación de ruido NLM reconstruye la señal verdadera S(i) en todos los puntos de tiempo i a través del promedio ponderado de todos los puntos D(j) dentro del rango predefinido. Los pesos se determinan mediante una medida de similitud entre D(i + δ) y D(j + δ), δ ∈ Δ.

$$S(i) = \frac{1}{Z(i)} \sum_{j \epsilon N(i)} w(i,j) D(j)$$

$$\text{donde } Z(i) = \sum_{j} w(i,j) \; y$$

$$w(i,j) = exp\left(-\frac{\sum_{\delta \epsilon \Delta} [D(i+\delta) - D(j+\delta)]^2}{2L_\Delta \lambda^2}\right)$$

λ es un parámetro de control de suavidad y Δ representa un parche local de muestras (contienen L_Δ muestras).

En cada punto, el suavizado NLM toma prestada información de todos los puntos que tienen patrones similares dentro del rango de búsqueda. La medida de similitud determina cuántos períodos se incluirán y promediarán.

A Continuación se muestra la implementación en python

```
20  def NLM_1dDarbon(signal,Nvar,P,PatchHW):
21      if isinstance(P,int): # scalar has been entered; expand into patch sample index vect
22          P = P-1 #Python start index from 0
23          Pvec = np.array(range(-P,P+1))
24      else:
25          Pvec = P # use the vector that has been input
26      signal = np.array(signal)
27      #debug = [];
28      N = len(signal)
```

```
    denoisedSig = np.empty(len(signal)) #NaN * ones(size(signal));
    denoisedSig[:] = np.nan
    # to simpify, don't bother denoising edges
    iStart = PatchHW+1
    iEnd = N - PatchHW
    denoisedSig[iStart: iEnd] = 0

    #debug.iStart = iStart;
    #debug.iEnd = iEnd;

    # initialize weight normalization
    Z = np.zeros(len(signal))
    cnt = np.zeros(len(signal))

    # convert lambda value to  'h', denominator, as in original Buades papers
    Npatch = 2 * PatchHW + 1
    h = 2 * Npatch * Nvar**2
```

```
for idx in Pvec: # loop over all possible differences: s - t
    # do summation over p - Eq.3 in Darbon
    k = np.array(range(N))
    kplus = k + idx
    igood = np.where((kplus >=0) & (kplus < N)) # ignore OOB data; we could also handle it
    SSD = np.zeros(len(k))
    SSD[igood] = (signal[k[igood]] - signal[kplus[igood]])**2
    Sdx = np.cumsum(SSD)

    for ii in range(iStart,iEnd): # loop over all points 's'
        distance = Sdx[ii + PatchHW] - Sdx[ii - PatchHW-1] #Eq 4;this is in place of point - by - point MSE
        # but note the - 1; we want to icnlude the point ii - iPatchHW

        w = math.exp(-distance/h) # Eq 2 in Darbon
        t = ii + idx # in the papers, this is not made explicit

        if t>0 and t<N:
            denoisedSig[ii] = denoisedSig[ii] + w * signal[t]
            Z[ii] = Z[ii] + w
            #cnt[ii] = cnt[ii] + 1
            #print('ii',ii)
            #print('t',t)
            #print('w',w)
            #print('denoisedSig[ii]', denoisedSig[ii])
            #print('Z[ii]',Z[ii])
 # loop over shifts

# now apply normalization
denoisedSig = denoisedSig/(Z + sys.float_info.epsilon)

denoisedSig[0: PatchHW+1] =signal[0: PatchHW+1]
denoisedSig[ - PatchHW: ] =signal[- PatchHW: ]
#debug.Z = Z;

return denoisedSig#,debug
```

Figura 3. Implementación del algoritmo Non local Means en Python.

2. ***Consultar qué otros tipos de señales wavelet se pueden usar para el análisis de señales ECG y adaptar el código del filtro wavelet que se entrega en el curso de acuerdo a la consulta***

La transformación wavelet se basa en un conjunto de wavelets que descompone la señal de ECG en coeficientes, permitiendo eliminar los de alta frecuencia asociados al ruido y reconstruir la señal. La esencia de las wavelets es su capacidad para analizar funciones en diferentes escalas, donde la elección de la escala influye en la resolución del análisis. Los algoritmos que utilizan wavelets procesan los datos en múltiples escalas, facilitando la identificación de patrones y anomalías en la señal de ECG, lo que mejora la precisión del diagnóstico [4]. Algunos tipos de señales wavelet son:

```
>>> for family in pywt.families():
...     print("%s family: " % family + ', '.join(pywt.wavelist(family)))
haar family: haar
db family: db1, db2, db3, db4, db5, db6, db7, db8, db9, db10, db11, db12, db13, db14,
sym family: sym2, sym3, sym4, sym5, sym6, sym7, sym8, sym9, sym10, sym11, sym12, sym1
coif family: coif1, coif2, coif3, coif4, coif5, coif6, coif7, coif8, coif9, coif10, co
bior family: bior1.1, bior1.3, bior1.5, bior2.2, bior2.4, bior2.6, bior2.8, bior3.1, b
rbio family: rbio1.1, rbio1.3, rbio1.5, rbio2.2, rbio2.4, rbio2.6, rbio2.8, rbio3.1, r
dmey family: dmey
gaus family: gaus1, gaus2, gaus3, gaus4, gaus5, gaus6, gaus7, gaus8
mexh family: mexh
morl family: morl
cgau family: cgau1, cgau2, cgau3, cgau4, cgau5, cgau6, cgau7, cgau8
shan family: shan
fbsp family: fbsp
cmor family: cmor
```

Los tipos más usados para el procesamiento de ECG son:

2.1 Wavelet de Haar

Una de las formas más simples de Wavelets, este realiza una descomposición de la señal en la mitad de su longitud, no es tan suave como otras wavelets, pero es útil en aplicaciones que requieren una descomposición rápida [5].

2.2 Wavelet de Daubechies (db)

Tiene una buena capacidad de localización tanto en el tiempo como en la frecuencia. Las wavelets de Daubechies, especialmente db4, son populares en el análisis de ECG porque tienen una forma que se asemeja a las ondas características del ECG. Muy utilizadas para la detección de picos QRS, descomposición de señales y reducción de ruido en ECG [6].

2.3 Wavelet de Symlet

Son una variante de las wavelets de Daubechies, con mayor simetría. Las Symlets ofrecen mejores resultados en términos de artefactos de reconstrucción, cuentan con menos distorsión en la reconstrucción de la señal comparada con las Daubechies. Se utiliza bastante para filtrado de señales ECG y análisis de variabilidad de la frecuencia cardíaca.

Para implementar las wavelets en el código, es necesario seleccionar la familia deseada y modificarla en esta línea,

```
data_wavelet = pywt.wavedec( data, 'familia', level=8 );
```

En nuestro caso se eligió la familia sym orden 6 (sym6) para procesar las señales, ya que según la literatura es el más adecuado para este procesamiento [7].

3. Escoger 10 señales al azar y aplicar un flujo de procesamiento que conste de

3.1 Flujo 1

1. Filtro pasa-altas usando filtro IIR a 0.5 Hz.
2. Filtro wavelet modificado del punto 3
3. Filtrado pasa bajas 50 Hz. Justificar la elección de parámetros y si se usa FIR o IIR

Para realizar el Flujo 1, se optó por el uso de un filtro pasa altas IIR, ya que este tipo de filtro es más efectivo para atenuar el ruido en las señales de ECG y ofrece una mayor velocidad de procesamiento. Los parámetros establecidos para este filtro son: la frecuencia de muestreo (Fs) utilizada tanto en los datos del proyecto dos como en este caso es de 500 Hz; la frecuencia de corte (Fc) es de 0.5 Hz, tal como se propone en el enunciado del ejercicio; y el orden del filtro es de 5, dado que, según la literatura, los filtros con un orden menor a 6 son los más adecuados para este tipo de aplicaciones [8].Para el paso tres, se optó nuevamente por un filtro IIR debido a sus ventajas en términos de eficiencia computacional, ya que, en general, requiere un orden más bajo que un filtro FIR para lograr un rendimiento similar. Esto se traduce en una menor demanda de recursos computacionales y permite mantener un rendimiento adecuado, esto va acompañado de el uso de menor memoria o menos uso de almacenamiento para los coeficientes, incluso al procesar señales de alta frecuencia, lo que es crucial en aplicaciones como el análisis de ECG [9].

3.2 Flujo 2

1. Detrend
2. Filtro wavelet modificado del punto 3
3. Filtrado pasa bajas 50 Hz. Justificar la elección de parámetros y si se usa FIR o IIR

Para el flujo 2, se realizó un detrend, seguido de el filtro wavelet (sym6) realizado en el apartado 2.3, luego, se utilizó un filtro IIR pasa bajas, se escoge este ya que, comparado con el FIR atenúa mejor el ruido, según la literatura. Como se explica anteriormente en el tercer paso del flujo 1. Los parámetros utilizados fueron la frecuencia de muestreo igual a 500 Hz, la frecuencia de corte 50 Hz, y el mismo orden 5 del filtrado pasa bajas IIR del flujo 1, ya que este tipo de filtros requieren órdenes muy bajos.

3.3 Flujo 3

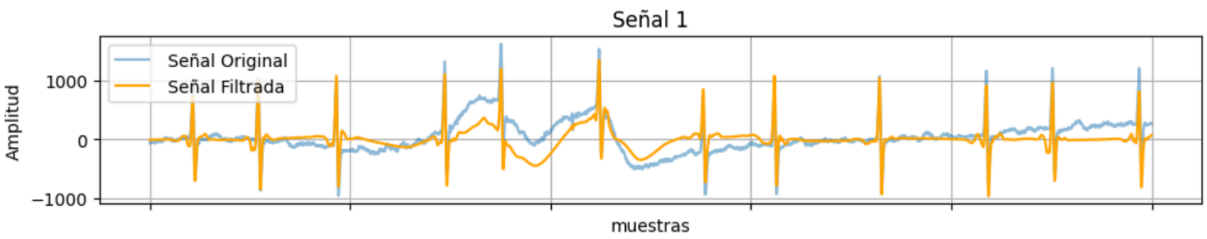
1. Filtro pasa-altas usando filtro IIR a 0.5 Hz. Justificar la elección de parámetros y si se usa FIR o IIR
2. Filtrado pasa bajas 50 Hz. Justificar la elección de parámetros y si se usa FIR o IIR

Para el flujo tres, se utilizaron los filtros IIR previamente diseñados, debido a las razones mencionadas anteriormente. Se mantuvieron los mismos parámetros de frecuencia de corte, frecuencia de muestreo y orden

4. Describir los resultados obtenidos y decidir si el resto del procesamiento se hace con el flujo 1, el flujo 2 o el flujo 3

Para la descripción de los resultados, se ejemplificará el análisis utilizando una de las 10 señales. Las demás señales se pueden visualizar en el archivo de Colab.

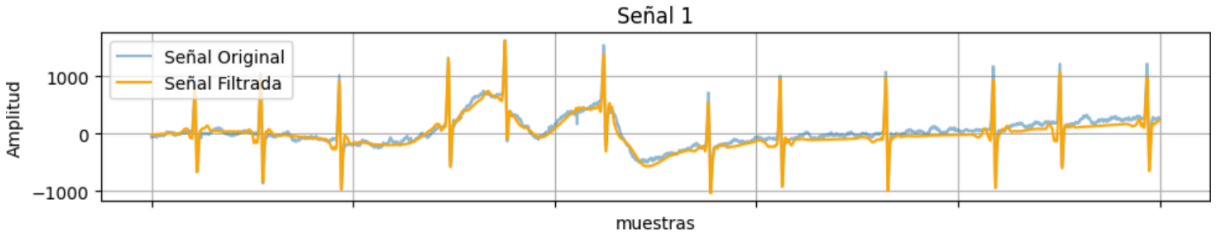
4.1 Flujo 1



Gráfica 2. señal 1 original y filtrada por flujo 1

En la *gráfica 1.* se observa que la señal original, tiene picos pronunciados que parecen ruidos, con oscilaciones irregulares, estos picos destacan por su altura en comparación con los segmentos donde la señal está más estable, la señal filtrada suaviza estos picos reduciendo la amplitud y haciendo un tipo de offset, pero conservando la tendencia de la señal, a lo largo de las muestras, sigue un patrón similar a la señal original pero sin transiciones bruscas y sin ruido.

4.2 Flujo 2



Gráfica 2. señal 1 original y filtrada por flujo 2

En la gráfica 2. se observa que la señal filtrada presenta una disminución en la variabilidad de los picos, elimina los picos de menor amplitud y disminuye la de los picos más pronunciados, esto indica que el filtro logró suavizar la señal. También se muestra que la señal filtrada se adapta muy bien a la forma de la señal original, esto posiblemente debido al filtro wavelet que ayuda a captar eventos transitorios sin distorsionar demasiado la señal.

4.3 Flujo 3

Gráfica 3. señal 1 original y filtrada por flujo 3

En la *gráfica 3*, se puede observar que la señal filtrada conserva la forma general de la señal original. Sin embargo, a pesar de esta conservación de la forma, se mantiene perceptible una cierta cantidad de ruido en la señal filtrada, lo que indica que el proceso de filtrado ha atenuado parcialmente, pero no ha eliminado por completo.

4.4 Elección de flujo

Se optó por el flujo número 2, ya que consideramos que es el que más se adaptó a la forma de la señal sin dejar de cumplir con la función de filtrado. En el caso del flujo 1, a pesar que

filtró bien, se presentó una distorsión en la forma de la onda, mientras que en el flujo 3, aunque se adaptó completamente a la forma, no filtró lo necesario.

5. ***Para cada señal extraer la frecuencia que contiene la máxima potencia usando Welch.***

Para el cálculo de frecuencia de máxima potencia (fmp) se utilizó una frecuencia de muestreo de 500, ya que fue con la que se tomaron los datos. Se escogió un ancho de ventana de 1000 para tener un buen balance entre la resolución en frecuencia y la reducción de la varianza en la estimación de potencia. Así mismo, un solapamiento del 50% del ancho de ventana (500) para obtener un suavizado adecuado sin perder mucha información. Por último, una ventana tipo Hanning debido a que ésta suele ser la opción segura para la mayoría de los análisis de potencia, ya que minimiza el efecto de las discontinuidades en los bordes de los segmentos [10].

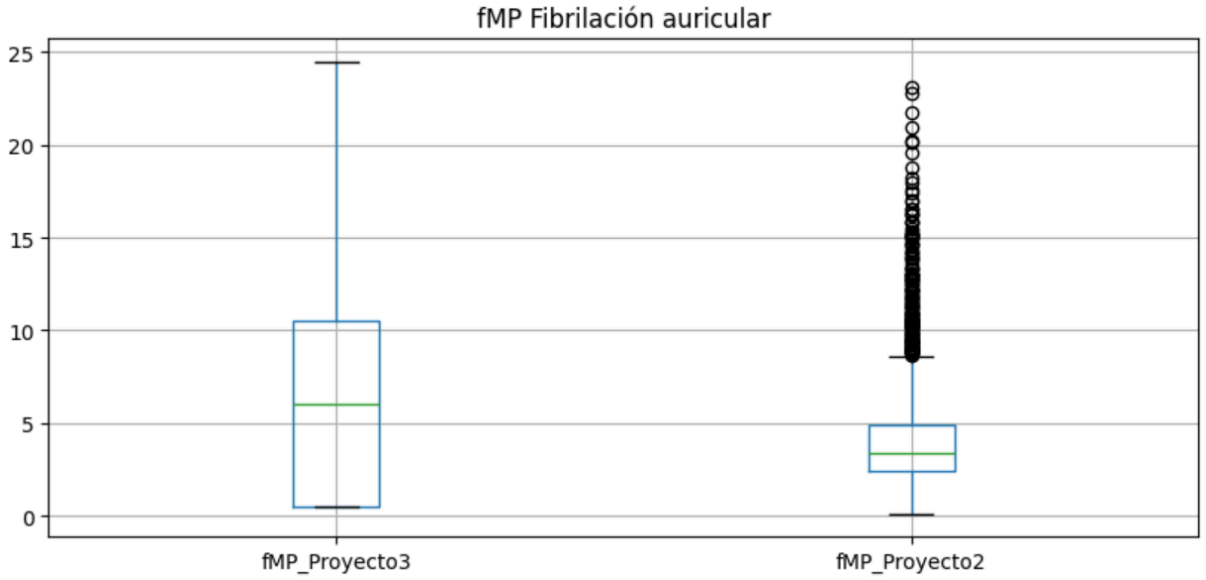
6. ***Crear una rutina que aplique sobre todos los archivos de la base de datos las rutina 3 al 5 y almacene los resultados en un dataframe donde se pueda registro, tipo de patología y el frecuencia de máxima potencia (fMP)***

Para la creación de la rutina se implementó una función que tuviera diferentes pasos. El primer paso se encargó de cargar correctamente la señal, realizando un procesamiento en el cual se elimina el primer valor que no corresponde a datos del ECG, y se convirtieron todos los valores a flotantes, ya que se encontraban en string. Una vez se tiene la señal cargada se le aplica el flujo 2, empleando primero un detrend, luego el filtro Wavelet y finalmente un filtro pasabajas IIR.
Para hacer un análisis comparativo con la señal del proyecto dos se le aplica el mismo proceso de detrend y normalización a la señal de este proyecto.

7. ***Comparar los resultados de fMP del proyecto 3 con los del proyecto 2 usando estadística descriptiva: gráficos y pruebas de hipótesis Discuta si hay más diferencias entre los tipos de señales con el flujo de procesamiento propuesto respecto al que se trabajó en el proyecto 2.***

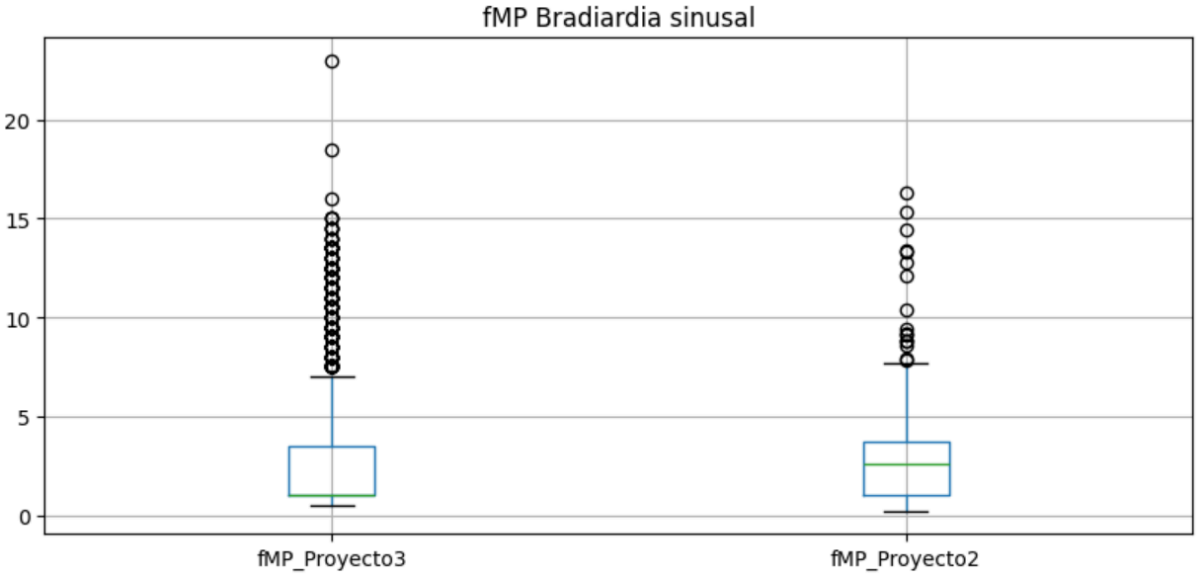
Para llevar a cabo la comparación entre el fMP de las señales del proyecto dos con las del proyecto tres por medio de la estadística descriptiva, se realizó una gráfica de cajas y bigotes

donde se evidencian las diferencias entre las señales, tanto de fibrilación auricular como de bradicardia sinusal.



Gráfica 4. fMP de ambos proyectos en señales de fibrilación auricular.

En la *gráfica 4.* se observa que la frecuencia de máxima potencia (fMP) del proyecto tres, presenta una caja con mayor amplitud en comparación con la del proyecto dos, lo que indica una mayor dispersión o variabilidad en los valores; la mediana cercana a seis, está situada cerca del centro de la caja, indicando una distribución relativamente simétrica dentro del rango intercuartílico. En el caso del proyecto dos, la mediana también se encuentra cerca de la mitad de la caja, pero se observa una notable concentración de valores atípicos en la parte superior.



Gráfica 5. fMP de ambos proyectos en señales de bradicardia sinusal.

En la *gráfica 5* se observa que en ambos proyectos tienen medianas de fMP similares, siendo menor la del proyecto tres. Esto indica que las frecuencias centrales de potencia no cambiaron significativamente entre los dos métodos de filtrado. El rango de las fMP en el proyecto tres es ligeramente mayor que en el proyecto dos , lo que significa que los valores de frecuencia están más dispersos en el primero, por lo tanto, el filtrado en el proyecto tres tiene una mayor variabilidad en las frecuencias de máxima potencia. Por otra parte, Ambas señales tienen muchos valores atípicos, la del proyecto tres parece tener más valores atípicos en comparación con el proyecto dos y de mayor magnitud, lo que puede indicar que el método de procesamiento en proyecto tres no redujo las frecuencias altas tanto como el procesamiento en proyecto dos.

Posteriormente, se realizaron las pruebas de hipótesis. En primer lugar, validamos los tres supuestos para verificar si se realiza una prueba paramétrica o no paramétrica.

7.1 Prueba de normalidad

Ho : los datos siguen una distribución normal
H1 : los datos no siguen una distribución normal

- Normalidad Bradicardia sinusal:
  fMP_proyecto3: 1.2109677382183844e-50
  fMP_proyecto2: 5.360338716805215e-40
- Normalidad Fibrilación auricular:
  fMP_proyecto3: 1.4415511080382088e-35
  fMP_proyecto2: 3.4876076159698926e-43

Como en todos los casos, el valor - $p < 0.05$, se rechaza H_0, por lo tanto, los datos no siguen una distribución normal.

7.2 Prueba de independencia

Son dos grupos dependientes, ya que aunque las señales han sido procesadas de manera diferente ambas provienen de la misma fuente.

7.3 Prueba de homocedasticidad

Ho: Las varianzas son iguales entre los grupos (homocedasticidad)
H1: Las varianzas no son iguales entre los grupos (heterocedasticidad)

- fMP de bradicardia sinusal en ambos proyectos: 4.3350669431935126e-11
- fMP de fibrilación auricular en ambos proyectos: 9.14939315963602e-194

En todos los casos, el valor $p < 0.05$, se rechaza Ho, por lo que las varianzas no son iguales entre los grupos.

Como solamente uno de los supuestos se cumple, se va a analizar como una prueba no paramétrica. Se realiza la prueba de U de mann Whitney

7. 4 Prueba de U de mann whitney

Ho: La mediana de las dos poblaciones es igual
H1: La mediana de las dos poblaciones no es igual

- fMP de bradicardia sinusal en ambos proyectos: 1.6000214221668402e-10
- fMP de fibrilación auricular en ambos proyectos: 0.0007349480366478733

Como todos los p- valor < 0.05 se rechaza Ho, lo que significa que los filtros analizados tienen una diferencia estadísticamente significativa entre ellos.

8. Conclusiones

1. El análisis estadístico arroja una diferencia significativa entre los dos proyectos, lo que demuestra que la metodología de filtrado y extracción de características en las señales puede llegar a impactar en los resultados obtenidos. Esto implica que la manera en que se procesan las señales es crucial, ya que puede influir en la precisión del análisis diagnóstico al preservar o alterar información relevante en las señales.

2. La extracción de características espectrales, proporciona una visión profunda de la composición frecuencial de la señal, revelando patrones y características que no son evidentes en el dominio del tiempo. Estas características son esenciales para tareas de clasificación, diagnóstico y monitoreo, por lo que si se tiene un filtrado correcto, estas tareas se realizan de forma eficiente.

3. El filtrado IIR es un método efectivo para eliminar ruido en señales ECG, debido a su bajo orden y menor uso de recursos. Sin embargo, el filtrado por transformada wavelet ofrece una ventaja significativa en la preservación de detalles de la señal. La elección de uno u otro depende de la necesidad específica.

9. Referencias

[1] W. S. Cleveland, «Robust Locally Weighted Regression and Smoothing Scatterplots», *J. Am. Stat. Assoc.*, vol. 74, n.º 368, pp. 829-836, dic. 1979, doi: 10.1080/01621459.1979.10481038.

[2] «Suavizado lowess - MATLAB & Simulink - MathWorks América Latina». Accedido: 7 de noviembre de 2024. [En línea]. Disponible en: https://la.mathworks.com/help/curvefit/lowess-smoothing.html

[3] «Confidence intervals for LOWESS models in python». Accedido: 7 de noviembre de 2024. [En línea]. Disponible en: https://james-brennan.github.io/posts/lowess_conf/

[4] C. Saritha, V. Sukanya, y Y. N. Murthy, «ECG Signal Analysis Using Wavelet Transforms».

[5] A. S. González, «CLASIFICACIÓN DE MÉTODOS USANDO WAVELETS».

[6] «filtros.pdf». Accedido: 29 de octubre de 2024. [En línea]. Disponible en: https://www.pablocetta.com/pdfs/publicaciones/filtros.pdf

[7] E. L. Lema-Condo, F. L. Bueno-Palomeque, S. E. Castro-Villalobos, E. F. Ordoñez-Morales, y L. J. Serpa-Andrade, «Comparison of wavelet transform symlets (2-10) and daubechies (2-10) for an electroencephalographic signal analysis», en *2017 IEEE XXIV International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*, ago. 2017, pp. 1-4. doi: 10.1109/INTERCON.2017.8079702.

[8] «Comparison of wavelet transform symlets (2-10) and daubechies (2-10) for an electroencephalographic signal analysis | IEEE Conference Publication | IEEE Xplore». Accedido: 7 de noviembre de 2024. [En línea]. Disponible en: https://ieeexplore.ieee.org/document/8079702

[9] A. Milchevski y M. Gusev, «Performance Evaluation of FIR and IIR Filtering of ECG Signals», en *ICT Innovations 2016*, G. Stojanov y A. Kulakov, Eds., Cham: Springer International Publishing, 2018, pp. 103-112. doi: 10.1007/978-3-319-68855-8_10.

[10] «Estimaciones de densidad espectral de potencia de poblaciones de ECG de 12 derivaciones y derivaciones de Frank normales y anormales | Publicación de la conferencia IEEE | IEEE Xplorar». Accedido: 7 de noviembre de 2024. [En línea]. Disponible en: https://ieeexplore.ieee.org/document/647478