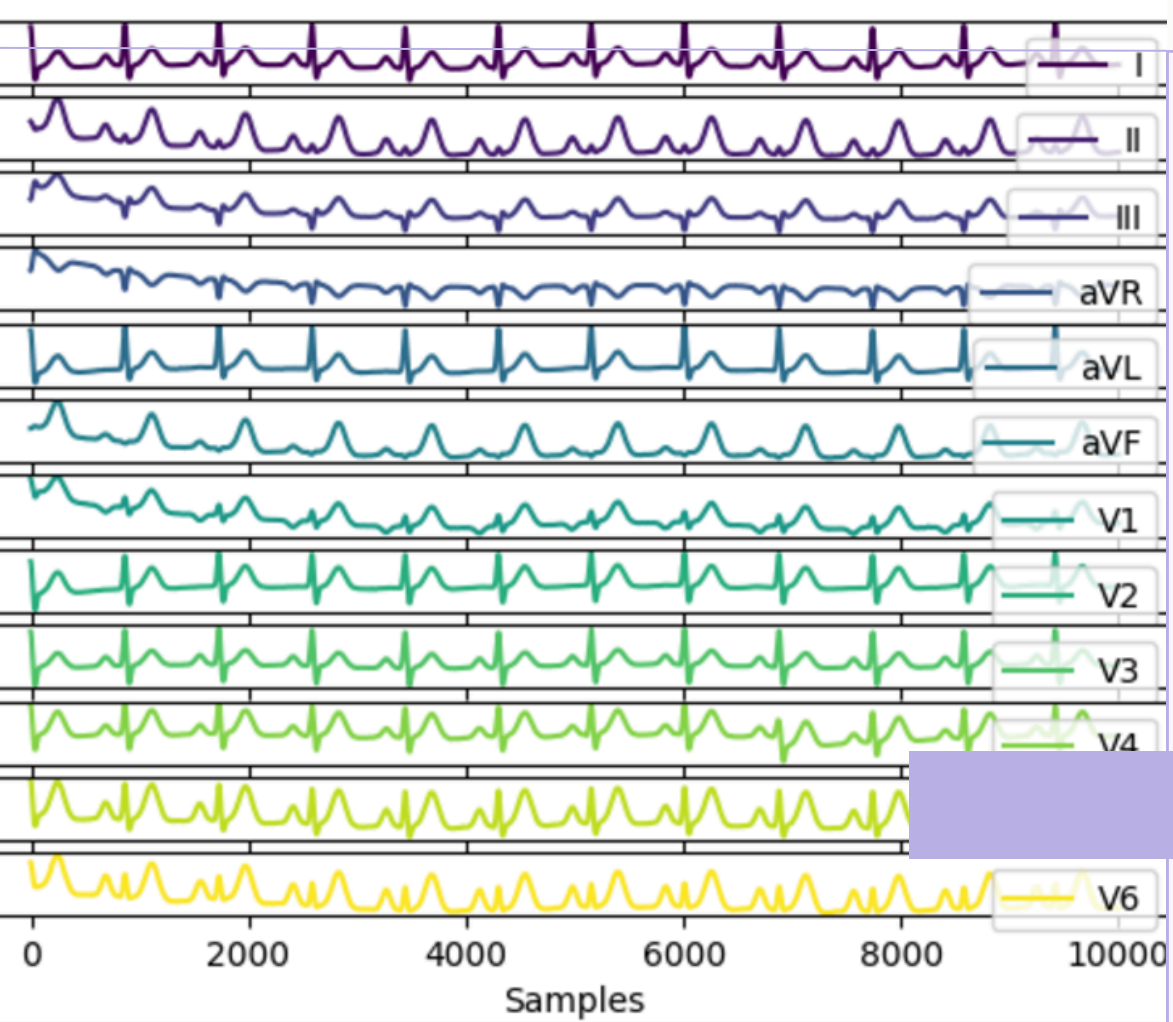
DICIEMBRE 2024

NeuroKit2

MANUAL

La caja de herramientas de Python para el procesamiento de señales neurofisiológicas



GUÍA DE MANEJO ECG

GENERALIDADES

¿ QUÉ ES?

Este paquete es una herramienta fácil de utilizar que ofrece acceso simplificado a potentes rutinas para el procesamiento de bioseñales. Está diseñado para que tanto investigadores como clínicos sin experiencia avanzada en programación o en técnicas de procesamiento de bioseñales puedan analizar datos fisiológicos de manera eficiente

¿CÓMO SE UTILIZA?

1. Abre la terminal donde se desarrollará el trabajo e instala la libreria.

```
pip install neurokit2
```

2. En cada script de Python importa el modulo.

```
import neurokit2 as nk
```

TABLA DE LIBRO DE CODIGOS

Detalla los códigos que puedes usar para calcular las variables necesarios en el procesamiento de señales, en este caso ECG. Se puede descargar acá

https://neuropsychology.github.io/NeuroKit/_static/neurokit_codebook.csv

se observa de esta manera:

Nombre del campo	Descripción del campo	Categoría de campo	Nombre del archivo de origen
ECG_Raw	La señal sin procesar.	Electrocardiografía	ecg_process.py
ECG_Clean	La señal limpia.	Electrocardiografía	ecg_process.py
ECG_Rate	Frecuencia cardíaca interpolada entre picos R.	Electrocardiografía	ecg_process.py

PROCEDMIENTO PARA LOCALIZACIÓN DE LAS ONDAS PQS Y T

Este ejemplo enseña cómo utilizar NeuroKit en Python para identificar y localizar los diferentes componentes del complejo QRS en una señal de ECG, incluyendo los picos P y T, y sus inicios y finales. El proceso es el siguiente:

1. Se cargan las librerías, la primera para el procesamiento y análisis de señales fisiológicas, la segunda para realizar operaciones matemáticas y la tercera para el análisis de datos estructurados

```
import neurokit2 as nk
import numpy as np
import pandas as pd
```

2. Se carga la señal ECG de interés, en este caso, una señal de ECG de fibrilación auricular con una frecuencia de muestreo de 500Hz llamada *señal_ecg*

PROCESAMIENTO DE LA SEÑAL

1. Se aplica la función *ecg_peaks()* para visualizar los picos R

ECG_PEAKS

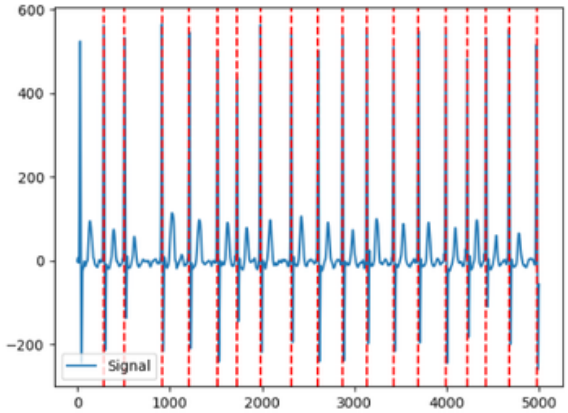
Detecta los picos R en una señal de ECG usando el método especificado. Como entrada, se recomienda usar una señal filtrada para mejores resultados y la frecuencia de muestreo. Devuelve un diccionario que contiene las muestras en las que se encuentran los picos R.

SINTAXIS

```
ecg_peaks(ecg_cleaned, sampling_rate=1000, method='neurokit',
correct_artifacts=False, show=False, **kwargs)
```

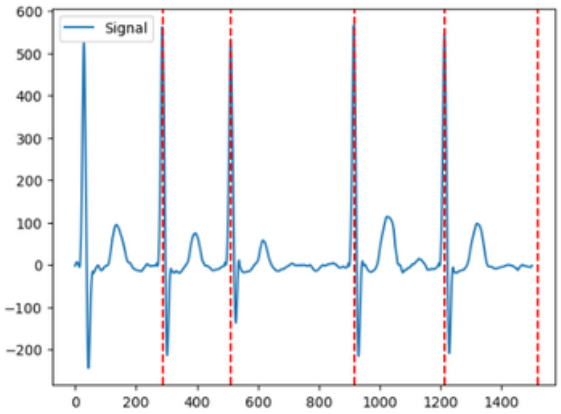
```
_, rpeaks = nk.ecg_peaks(señal_ecg, sampling_rate=500)
```

```
plot = nk.events_plot(rpeaks['ECG_R_Peaks'], señal_ecg)
```



2. Se seleccionan los primeros 5 picos R con el fin de verificar su correcta detección y asegurarse de que coincidan con las características esperadas de la señal cardíaca.

```
plot = nk.events_plot(rpeaks['ECG_R_Peaks'][:5], señal_ecg[:1500])
```



3. Se aplica la función ecg_delineate y se grfican los tres primeros picos

ECG_DELINEATE

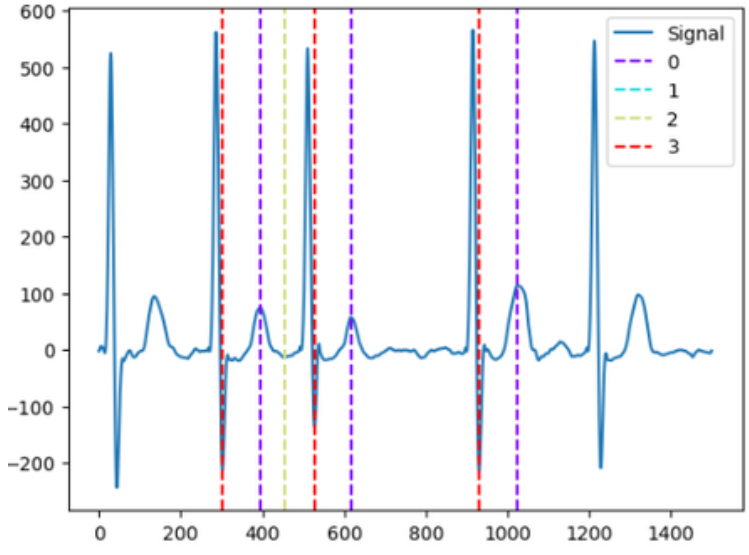
Delinea (es decir, identifica y segmenta) las diferentes ondas del ciclo cardíaco en una señal de ECG, específicamente el complejo QRS. Se le ingresa una señal ECG limpia, las muestras en las que se producen picos R y la frecuencia de muestreo. Devuelve un diccionario con información de los picos y un DataFrame en la que se producen picos marcados como "1" en una lista de ceros.

SINTAXIS

```
ecg_delineate(ecg_cleaned, rpeaks=None, sampling_rate=1000, method='dwt
show=False, show_type='peaks', check=False, **kwargs)
```

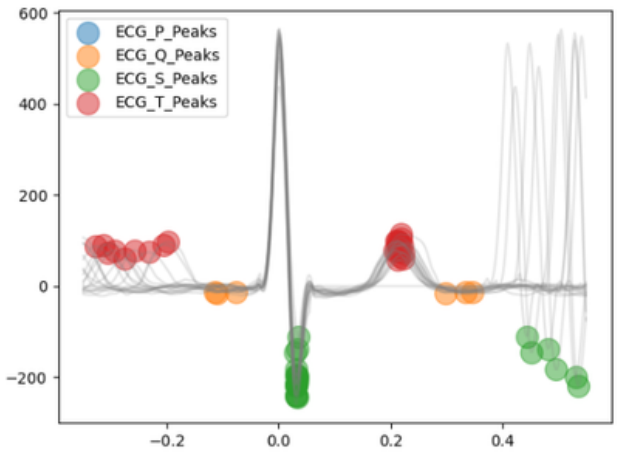
```
signals, waves = nk.ecg_delineate(señal_ecg, rpeaks, sampling_rate=500, method="peak")

plot = nk.events_plot([waves['ECG_T_Peaks'][:3],
                       waves['ECG_P_Peaks'][:3],
                       waves['ECG_Q_Peaks'][:3],
                       waves['ECG_S_Peaks'][:3]], señal_ecg[:1500])
```



4. Existe otra alternativa para tener una mejor visión general de todos los picos a la vez, esto se hace por medio del argumento *show* en la misma función delineate()

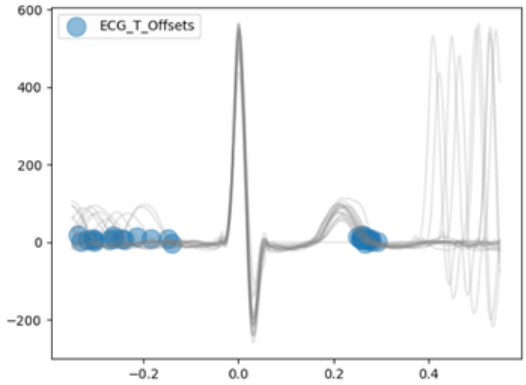
```
_, waves_peak = nk.ecg_delineate(señal_ecg,
                                 rpeaks,
                                 sampling_rate=500,
                                 method="peak",
                                 show=True,
                                 show_type='peaks')
```



EEn la gráfica se puede observar la ausencia de la onda P, lo cual es característico de la fibrilación auricular, ya que esta onda tiende a ocultarse en este trastorno.

5. Si se desea identificar los limites de las ondas se puede variar el argumento *show_type*, es decir el incio de los picos P y el desplazamiento de los picos T

```
(señal_ecg,
  rpeaks,
  sampling_rate=500,
  method="peak",
  show=True,
  show_type='bounds_T')
```



6. Existen tres métodos de visualización de las ondas, anteriormente, se utilizó "peaks", veamos los demás métodos "CWT" y "CWD"

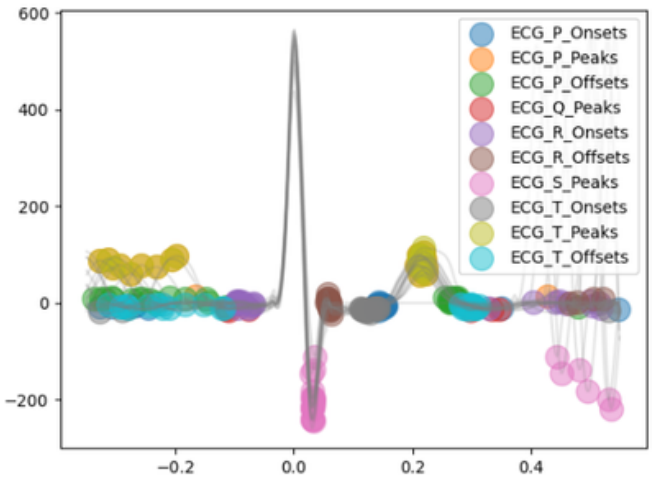
CWT Y CWD

Para realizar el método de ondiculas continuas y el método de Wavelet discreto es necesario instalar la biblioteca *PyWavelets*

```
pip install PyWavelets
```

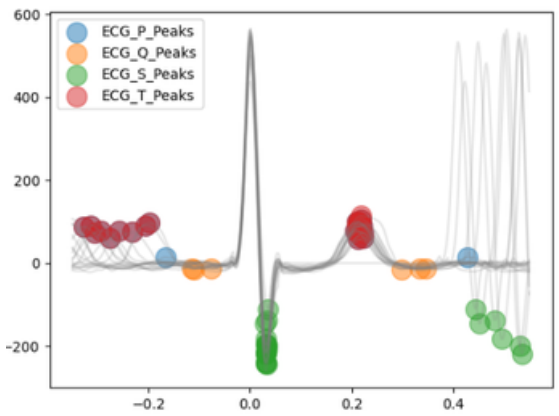
7. Se utiliza nuevamente la función *delineate()*, pero en este caso la parte *method* se reemplza por “cwt” o “cwd” dependiendo si es continua o discreta. El comando ‘all’ en show_type permite visualizar todos los picos con sus entradas y salidas.

```
signal_cwt, waves_cwt = nk.ecg_delineate(señal_ecg,
                                         rpeaks,
                                         sampling_rate=500,
                                         method="cwt",
                                         show=True,
                                         show_type='all')
```

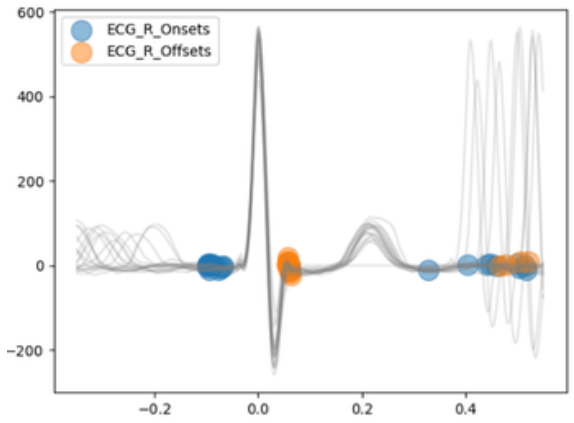


8. En estos métodos, se varía el argumento show_type si se quiere identificar los picos “picks” y los limites de las ondas “bounds_R”, “bounds_T” y “bounds_P”

```
show_type='peaks')
```




```
show_type='bounds_R')
```



Como se puede evidenciar, Neurolink es una herramienta diseñada para facilitar la identificación, análisis y procesamiento de señales ECG. Sus funciones intuitivas permiten a los usuarios, tanto principiantes como expertos, acceder a potentes algoritmos de procesamiento que ayudan a la detección de patrones y la extracción de características importantes.