Ізолінія в даній програмі проходить через PQ, не через TP.

27.06.2025

1) Пошук зубця S.

2) Тестування програми на автоматичній детекції.

28.06.2025

3) Прорахунок ізолінії з врахуванням піків та їх границь правильно.

4) Перевірка, чи ізолінія проходить правильно?

Порівняльна характеристика бібліотек

# === Выделим R-пики с помощью WFDB (можно также использовать nk.ecg\_peaks)  
#xqrs = wfdb.processing.XQRS(sig=ecg\_signal, fs=fs)  
#xqrs.detect()  
  
#r\_peaks = xqrs.qrs\_inds

Wfdb дає похибку в оцінці піків.

# === Загрузка своего сигнала (или из PhysioNet)  
# Если у тебя сигнал в виде массива:  
# ecg\_signal = np.loadtxt("your\_ecg\_data.txt")  
  
  
# Или загрузим пример:  
#record = wfdb.rdrecord('100', pb\_dir='mitdb') # MIT-BIH Arrhythmia Database  
#ecg\_signal = record.p\_signal[:, 0]

Взяти ЕКГ без пропусків і ті, в яких є дані варіабельності серцевого ритму.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Параметри |  |  |  |  |  |  |
| Амплітуда зубця P |  |  |  |  |  |  |
| Тривалість зубця P |  |  |  |  |  |  |
| Амплітуда зубця R |  |  |  |  |  |  |
| Тривалість комплексу QRS |  |  |  |  |  |  |
| Амплітуда зубця T |  |  |  |  |  |  |
| Тривалість зубця T |  |  |  |  |  |  |
| Час ЕКГ в мілісекундах |  |  |  |  |  |  |
| Сегмент PQ |  |  |  |  |  |  |
| Сегмент ST |  |  |  |  |  |  |
| Частота серцебиття (ЧСС) |  |  |  |  |  |  |

 Амплитуда сигнала

 Частота сердцебиения

 Временные интервалы (PR, QRS, QT)

 Byltrcs djky (P? Q? R? S? T)

За допомогою цих значень оцінюють основні параметри кардіограми:

* зубці;
* інтервали;
* сегменти.

При проведенні кардіограми у трьох стандартних відведеннях найчастіше використовуються такі значення:

* зубець Р – показує ступінь скорочення та розслаблення передсердь;
* зубець Т – відображає реполяризацію міокарда та розслаблення шлуночків;
* зубець R – відображає поширення імпульсів по міокарду правого та лівого шлуночків;
* зубець S – показує збудження у базальному шарі шлуночків;
* зубець Q – показує збудження перегородки між шлуночками;
* інтервал PQ – фіксує час проведення збудження по передсердям міокарда;
* ST – показує період повного збудження шлуночків.

У дорослої людини нормою можуть вважатися такі параметри кардіограми:

ЧСС – у стані спокою від 60 до 80 ударів за хвилину;

синусовий серцевий ритм – однакові інтервали PR;

інтервал RR – від 0,6 до 1,2 с;

інтервал PR – від 120 до 200 мс;

сегмент ST - 320 мс;

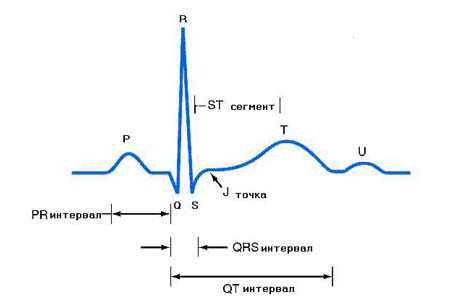
комплекс QRS – від 60 до 100-120 мс;

зубець Р – 80 мс;

зубець Т – 160 мс.

зубець J – у нормі повинен бути відсутнім.

<https://oxford-med.com.ua/ua/media-center/publikacii/kak-rasshifrovat-kardiogrammu/>



**Синусовий ритм**

***При синусовому ритмі PR-інтервали приблизно однакові*.**

import numpy as np

# Припустимо, що ці два масиви вже заповнені відповідними точками:

p\_onsets = np.array(waves\_peak["ECG\_P\_Onsets"][:count\_calc])

r\_peaks = np.array(waves\_peak["ECG\_R\_Peaks"][:count\_calc])

# Обчислюємо PR-інтервали

pr\_intervals = r\_peaks - p\_onsets

# Видаляємо NaN

pr\_intervals = pr\_intervals[~np.isnan(pr\_intervals)]

# Перевіримо варіацію:

std\_dev = np.std(pr\_intervals)

mean\_val = np.mean(pr\_intervals)

coefficient\_of\_variation = std\_dev / mean\_val

print(f"PR intervals: {pr\_intervals}")

print(f"Mean PR: {mean\_val:.2f}, Std: {std\_dev:.2f}, CoefVar: {coefficient\_of\_variation:.4f}")

**✅ Критерій "однаковості":**

* Якщо **коефіцієнт варіації** (std / mean) менше, скажімо, 0.1, то PR-інтервали досить стабільні → **можна припустити синусовий ритм**.

python

КопіюватиРедагувати

if coefficient\_of\_variation < 0.1:

print("Ймовірно синусовий ритм.")

else:

print("Можлива аритмія або не синусовий ритм.")

Це **спрощене правило**, але воно добре працює на нормальних ЕКГ. У реальній практиці ще враховується:

* наявність зубця P перед кожним QRS,
* фіксована форма зубця P,
* частота у межах 60–100 уд./хв.

Хочеш — можемо дописати повну перевірку синусового ритму.

Синхронізація масивів

**✅ Как это исправить:**

**1. Проверь соответствие R и P по времени:**

python

КопіюватиРедагувати

for i in range(len(p\_onsets)):

print(f"P: {p\_onsets[i]}, R: {r\_peaks[i]}, PR: {r\_peaks[i] - p\_onsets[i]}")

Если почти везде P > R, то массивы не соответствуют друг другу.

**2. Проверь визуально:**

Можно построить график:

python

КопіюватиРедагувати

import matplotlib.pyplot as plt

plt.plot(ecg\_signal, label='ECG')

plt.scatter(p\_onsets, ecg\_signal[p\_onsets], color='green', label='P onset')

plt.scatter(r\_peaks, ecg\_signal[r\_peaks], color='red', label='R peak')

plt.legend()

plt.title('Сопоставление P и R')

plt.show()

**3. Подбор пар "P перед R":**

Если p\_onsets и r\_peaks — просто массивы индексов, можно составить пары **P, ближайший R после него**:

python

КопіюватиРедагувати

corrected\_pr\_intervals = []

for p in p\_onsets:

r\_candidates = r\_peaks[r\_peaks > p]

if len(r\_candidates) > 0:

nearest\_r = r\_candidates[0]

corrected\_pr\_intervals.append(nearest\_r - p)

corrected\_pr\_intervals = np.array(corrected\_pr\_intervals)

print("PR интервалов:", corrected\_pr\_intervals)

print("Среднее:", np.mean(corrected\_pr\_intervals))

**💡 Вывод:**

Сейчас ты просто вычитаешь массивы одинаковой длины, не проверяя, соответствуют ли значения физиологически (P до R). Это главная причина отрицательных значений. Нужно **найти для каждого P соответствующий R позже него**, и тогда всё будет ок.

Хочешь, могу помочь обернуть это всё в функцию.

Можно ли применять pr\_intervals = pr\_intervals[~np.isnan(pr\_intervals)], не сместятся ли интервалы в массиве?

**Програма ChatGPT сказала:**

Отличный вопрос! Краткий ответ: **можно применять pr\_intervals = pr\_intervals[~np.isnan(pr\_intervals)], но только если тебе не нужно сохранять связь с оригинальными индексами P и R.**

**🔍 Что делает pr\_intervals[~np.isnan(pr\_intervals)]?**

Это просто фильтрация: удаляются значения nan. Это **не влияет** на массивы p\_onsets и r\_peaks — они останутся как были. Но при этом **позиции в pr\_intervals после фильтрации уже не соответствуют тем же индексам в p\_onsets или r\_peaks**.

**📌 Когда можно применять:**

* ✅ Ты работаешь **только с интервалами**, без привязки к исходным точкам.
* ✅ Хочешь посчитать среднее, стандартное отклонение, коэффициент вариации и т.п.
* ✅ Тебе не важно, где именно по времени были эти интервалы.

**❗ Когда нельзя:**

* ❌ Ты хочешь позже визуализировать пары P–R на графике.
* ❌ Нужно вернуться к исходным точкам (например, чтобы узнать, где был "плохой" интервал).
* ❌ Тебе нужна статистика по времени или индексам.

**💡 Решение, если нужна точность:**

Если хочешь сохранить соответствие, можно использовать маску для всех массивов:

python

КопіюватиРедагувати

mask = ~np.isnan(pr\_intervals)

pr\_intervals = pr\_intervals[mask]

p\_onsets = p\_onsets[mask]

r\_peaks = r\_peaks[mask]

Теперь все три массива остаются синхронизированными и без NaN.

Если хочешь — могу помочь написать функцию, которая автоматически находит PR-интервалы с правильной фильтрацией и возвращает чистые данные.

Если есть центр окна при подсчете фрактальной размерности Хигучи и сама фрактальная размерность, то можно по окнам усреднять фрактальную размерность по всей ЭКГ?

**Програма ChatGPT сказала:**

Обробка тривала 20s

Коротко: **да, можно усреднять HFD по окнам**, но делайте это аккуратно.

Вот что важно:

1. Что именно вы усредняете

* Усредняйте **готовые оценки HFD** из каждого окна. Это описательная сводка по всей записи.
* Если окна разной длины — используйте **взвешенное среднее** по длительности окна; иначе — обычное среднее/медиану.

1. Робастность

* Из-за артефактов и экстрасистол лучше брать **медиану** или **усечённое среднее (trimmed mean 10–20%)**, плюс интерквартильный размах как меру разброса.
* Дополнительно можно дать **доверительный интервал** бутстрэпом по окнам.

1. Размер окна и kmax

* Для метода Хигучи в каждом окне должно выполняться грубое правило: **N/kmax ≥ 10–20** (где N — число точек в окне). Иначе оценка шумная и смещённая.
* Проведите **анализ чувствительности**: посчитайте HFD при нескольких kmax (например, 6–12) и убедитесь, что тренд стабилен.

1. Перекрытие окон

* 50% перекрытие — хороший компромисс. Сильное перекрытие даёт зависимые оценки; это нормально, но не переоценивайте число «независимых» окон при статистике.

1. Стационарность

* ЭКГ/ВСР — нестационарны. Усреднение сглаживает **локальные изменения**. Поэтому помимо одной сводной цифры полезно хранить **профиль HFD(t)** и, например, его медиану + IQR.

1. Предобработка

* На сыром ЭКГ: уберите дрейф изолинии, высокочастотный шум, крупные артефакты.
* На RR-рядах: корректируйте/интерполируйте артефакты и экстрасистолы, иначе HFD «поплывёт».