PulseDB Time-Series Clustering Project

Project Overview

This project groups and analyzes short blood pressure (ABP) signals from the PulseDB dataset. Instead of using machine learning, it uses three main algorithms — divide-and-conquer clustering, closest pair search, and Kadane's algorithm — to find patterns and active periods in the data. The goal was to show how algorithmic logic alone can create useful and interpretable results in biomedical data.

Data and Setup

Dataset: VitalDB_CalBased_Test_Subset.mat (ABP signals, 10-second segments)

Tools Used: Python, NumPy, h5py, Matplotlib

Setup:

Place .mat file in data/raw/

Run python src/main.py --mat "C:\PulseTemp\VitalDB_CalBased_Test_Subset.mat" --out results

The system loads the ABP data, processes segments, and saves results (plots, clusters, and summaries) in the results/ folder.

How the Code Works

File	Description
main.py	Runs the full pipeline and organizes outputs.
clustering.py	Uses divide-and-conquer logic to cluster similar signals.
closest_pair.py	Finds the two most similar time series in each cluster.
kadane.py	Detects the most active interval in each signal.
pulsedb_from_mat.py	Loads and cleans ABP signal data from the .mat file.

Flowchart (simplified):
Data → Clustering → Closest Pair → Kadane Analysis → Results

Each module is small, modular, and easy to test independently.

Diagram

```
+----------------------+
|  VitalDB .mat file   |
+----------+-----------+
           |
           v
+----------------------+
| pulsedb_from_mat.py  |
| → Loads ABP signals  |
+----------+-----------+
           |
           v
+----------------------+
| clustering.py        |
| → Divide & Conquer   |
|   forms clusters     |
+----------+-----------+
           |
           v
+----------------------+
| closest_pair.py      |
| → Finds most similar |
|   pair in cluster    |
+----------+-----------+
```

```
          |
          v
+----------------------+
| kadane.py            |
| → Detects most active|
|   region in segment  |
+----------+-----------+
          |
          v
+----------------------+
| main.py              |
| → Runs pipeline &    |
|   saves results      |
+----------------------+
```

Algorithm Summaries

- **Divide-and-Conquer Clustering:**
  Splits the data into smaller groups based on similarity. Keeps dividing until each group is tight enough.
- **Closest Pair Algorithm:**
  Finds two signals in each cluster that are most alike (based on DTW or correlation).
- **Kadane's Algorithm:**
  Locates the strongest peak or activity region within each signal segment.

Verification with Toy Example

I first tested the algorithms on a small dataset of 10 synthetic signals:

- The clustering formed 2–3 clear groups.
- The closest pair function correctly found nearly identical shapes.
- Kadane's algorithm marked peak regions that matched visible pressure spikes.

This helped confirm that each part of the system worked correctly before scaling to real PulseDB data.

Execution and Results

It takes a little long for the result to execute.

After running the full dataset:

- **Loaded Segments:** 50
- **Clusters Formed:** 4
- **Done.**

Generated files include:

- results/clusters.txt – cluster summary
- results/c*/closest_a.png / closest_b.png – closest pair visualizations

Findings and Discussion

The project demonstrates that clustering by shape (correlation/DTW) is effective for ABP signals. Kadane's algorithm also adds interpretability by showing why certain segments are grouped (similar active regions).

Challenges:

- DTW distance can be slow for large data.
- Only ABP signals were used; ECG/PPG could add more insight.

Improvements:

- Speed up DTW with pruning or windowing.
- Add automatic report generation and more visuals.

Conclusion

This project successfully demonstrated how divide-and-conquer, closest-pair, and Kadane's algorithm can together cluster and explain physiological time-series data.

The results were accurate, interpretable, and showed real patterns in blood pressure behavior — all without using black-box ML models.