In [ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

from os import path

import scipy.signal as signal
```

Variáveis da análise

In [ ]:
```
datasetFolder = "./dataset/picked"
ecg = "1002867.txt"
```

Carregando a base de dados

In [ ]:
```
header = [
    "amostra",
    "lead I",
    "lead II",
    "lead  II",
    "aVR",
    "aVL",
    "aVF",
    "V1",
    "V2",
    "V3",
    "V4",
    "V5",
    "V6"
]

dataset = pd.read_csv(
    path.join(datasetFolder, ecg)
)

dataset.columns = header

print(dataset.head())
print(dataset.shape)
```

```
   amostra  lead I  lead II  lead  II  aVR  aVL  aVF   V1  V2   V3   V4  V
5  \
0        1    -300     -614      -314  457 -464    7  150  64  184  154  8
8
1        2    -298     -608      -310  453 -459    6  151  64  184  155  8
8
2        3    -296     -604      -308  450 -456    6  151  64  184  155  8
8
3        4    -295     -599      -304  447 -451    5  150  63  183  152  8
8
4        5    -292     -591      -299  441 -445    4  147  62  180  150  8
8

    V6
0  188
1  188
2  188
3  188
4  187
(4999, 13)
```

Características do Dataset

In [ ]:
```
samplingFrequency = 500
samplingPeriod = 1 / samplingFrequency
nyquistFrequency = samplingFrequency / 2
```

Pré-processamento

In [ ]:
```
times = np.arange(
    0,
    dataset.shape[0] / samplingFrequency,
    samplingPeriod
)

frequencies = np.fft.fftfreq(dataset.shape[0], samplingPeriod)

dataset["Tempo"] = times
dataset["Frequencia"] = frequencies
```

Filtros

In [ ]:
```
def lowPassFilter(dataset, derivationName, cutoff):
    derivation = dataset[derivationName]
    order = 2

    normalCutoff = cutoff / nyquistFrequency

    b, a = signal.butter(order, normalCutoff, btype="low")

    derivationFiltred = signal.filtfilt(b, a, derivation)

    return derivationFiltred

def highPassFilter(dataset, derivationName, cutoff):
    derivation = dataset[derivationName]
    order = 2

    normalCutoff = cutoff / nyquistFrequency
```

```
    b, a = signal.butter(order, normalCutoff, btype="high")

    derivationFiltred = signal.filtfilt(b, a, derivation)

    return derivationFiltred


def notchFilter(dataset, derivationName, cutoff):
    derivation = dataset[derivationName]
    normalCutoff = cutoff / nyquistFrequency

    b, a = signal.iirnotch(normalCutoff, 60, samplingFrequency)

    derivationFiltred = signal.lfilter(b, a, derivation)

    return derivationFiltred
```

Derivadas

In [ ]:
```
def firstDerivate(dataset, derivationName):
    times = dataset["Tempo"]
    deltaTime = times[1] - times[0]

    derivation = dataset[derivationName]
    deltaDerivation = np.diff(derivation)

    return deltaDerivation / deltaTime

def secondDerivate(dataset, derivationName):
    times = dataset["Tempo"]
    deltaTime = times[1] - times[0]

    firstDerivative = firstDerivate(dataset, derivationName)
    deltaFirstDerivative = np.diff(firstDerivative)

    return deltaFirstDerivative / deltaTime
```

Subamostragem

In [ ]:
```
def subSampling(dataset, derivationName, factor):
    derivation = dataset[derivationName]
    times = dataset["Tempo"]

    return derivation[::factor], times[::factor]
```

Funções de Plot

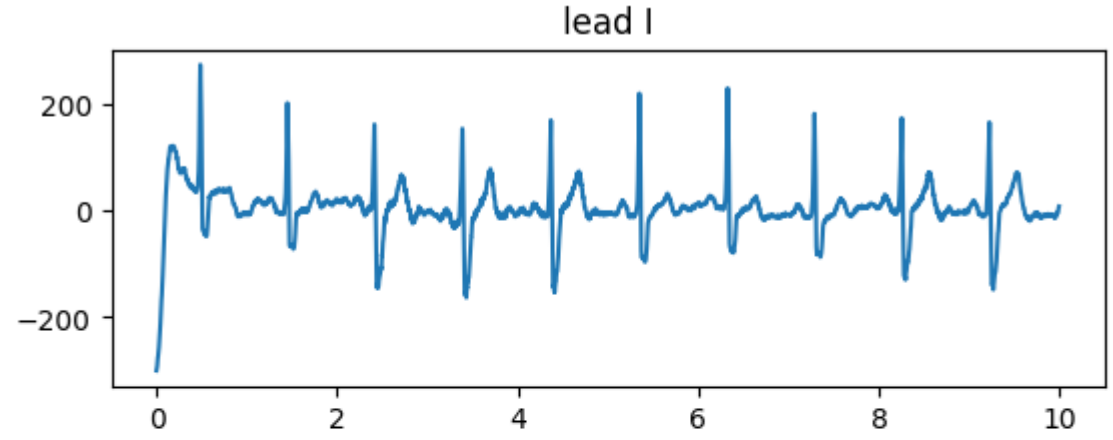
In [ ]:
```
def derivationFourierPlot(dataset, derivationName):
    times = dataset["Tempo"]
    frequencies = dataset["Frequencia"]
    derivation = dataset[derivationName]

    derivationFourier = np.fft.fft(derivation)
    derivationFourier = np.abs(derivationFourier)

    plt.figure()
```

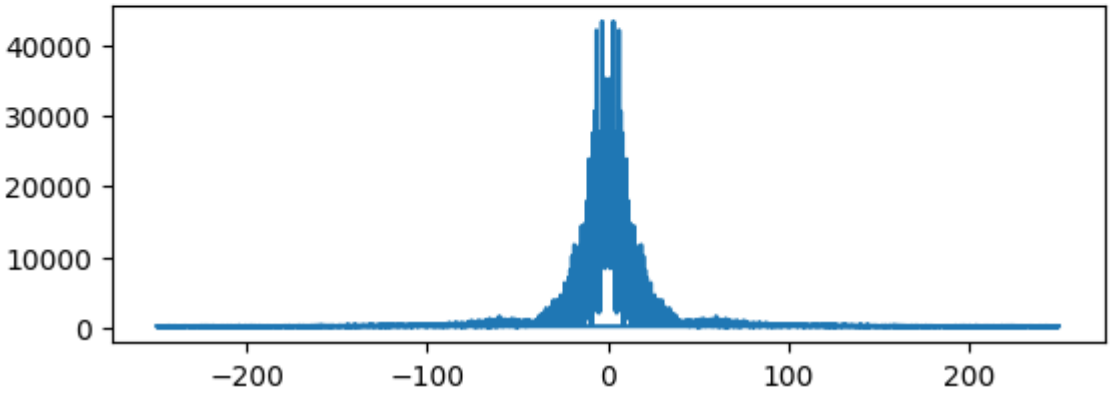
```
    plt.subplot(2, 1, 1)
    plt.plot(times, derivation)

    plt.title(derivationName)

    plt.subplot(2, 1, 2)
    plt.plot(frequencies, derivationFourier)

def derivationLowPassPlot(dataset, derivationName, cutoff):
    times = dataset["Tempo"]
    derivation = dataset[derivationName]

    derivationFiltred = lowPassFilter(dataset, derivationName, cutoff)

    plt.figure()

    plt.subplot(2, 1, 1)
    plt.plot(times, derivation)

    plt.title(derivationName)

    plt.subplot(2, 1, 2)
    plt.plot(times, derivationFiltred)

def derivationHighPassPlot(dataset, derivationName, cutoff):
    times = dataset["Tempo"]
    derivation = dataset[derivationName]

    derivationFiltred = highPassFilter(dataset, derivationName, cutoff)

    plt.figure()

    plt.subplot(2, 1, 1)
    plt.plot(times, derivation)

    plt.title(derivationName)

    plt.subplot(2, 1, 2)
    plt.plot(times, derivationFiltred)

def derivationNocthPlot(dataset, derivationName, cutoff):
    times = dataset["Tempo"]
    derivation = dataset[derivationName]
    derivationFiltred = notchFilter(dataset, derivationName, cutoff)

    plt.figure()

    plt.subplot(2, 1, 1)
    plt.plot(times, derivation)

    plt.title(derivationName)

    plt.subplot(2, 1, 2)
    plt.plot(times, derivationFiltred)

def derivationDerivatesPlot(dataset, derivationName):
    times = dataset["Tempo"]
    derivation = dataset[derivationName]

    firstDerivateDerivation = firstDerivate(dataset, derivationName)
```

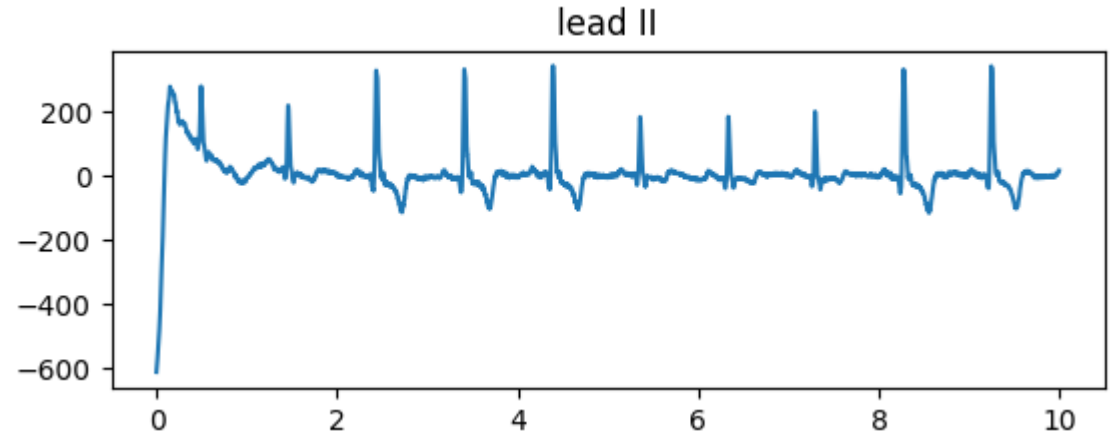
```
    secondDerivateDerivation = secondDerivate(dataset, derivationName)

    _, axes = plt.subplots(3, 1, sharex = True)

    axes[0].set_title(derivationName)
    axes[0].plot(times, derivation)

    axes[1].set_title("First Derivate")
    axes[1].plot(times[:-1], firstDerivateDerivation)

    axes[2].set_title("Second Derivate")
    axes[2].plot(times[:-2], secondDerivateDerivation)

    plt.tight_layout()
    plt.show()


def subSamplingPlot(dataset, derivationName, factor):
    times = dataset["Tempo"]
    derivation = dataset[derivationName]

    [
        derivationSubSampling,
        timesSubSampling
    ] = subSampling(dataset, derivationName, factor)

    plt.figure()

    plt.subplot(2, 1, 1)
    plt.plot(times, derivation)

    plt.title(derivationName)

    plt.subplot(2, 1, 2)
    plt.plot(timesSubSampling, derivationSubSampling)
```

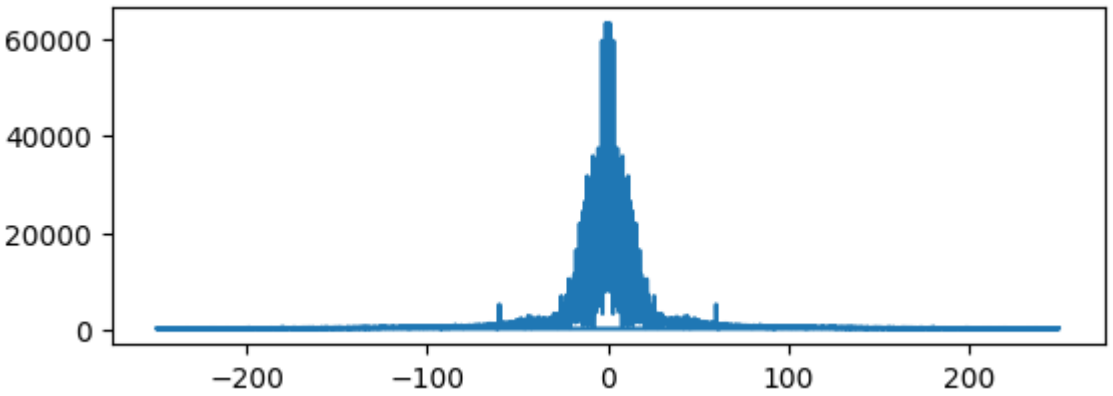
Derivações no domínio do Tempo e da Frequência

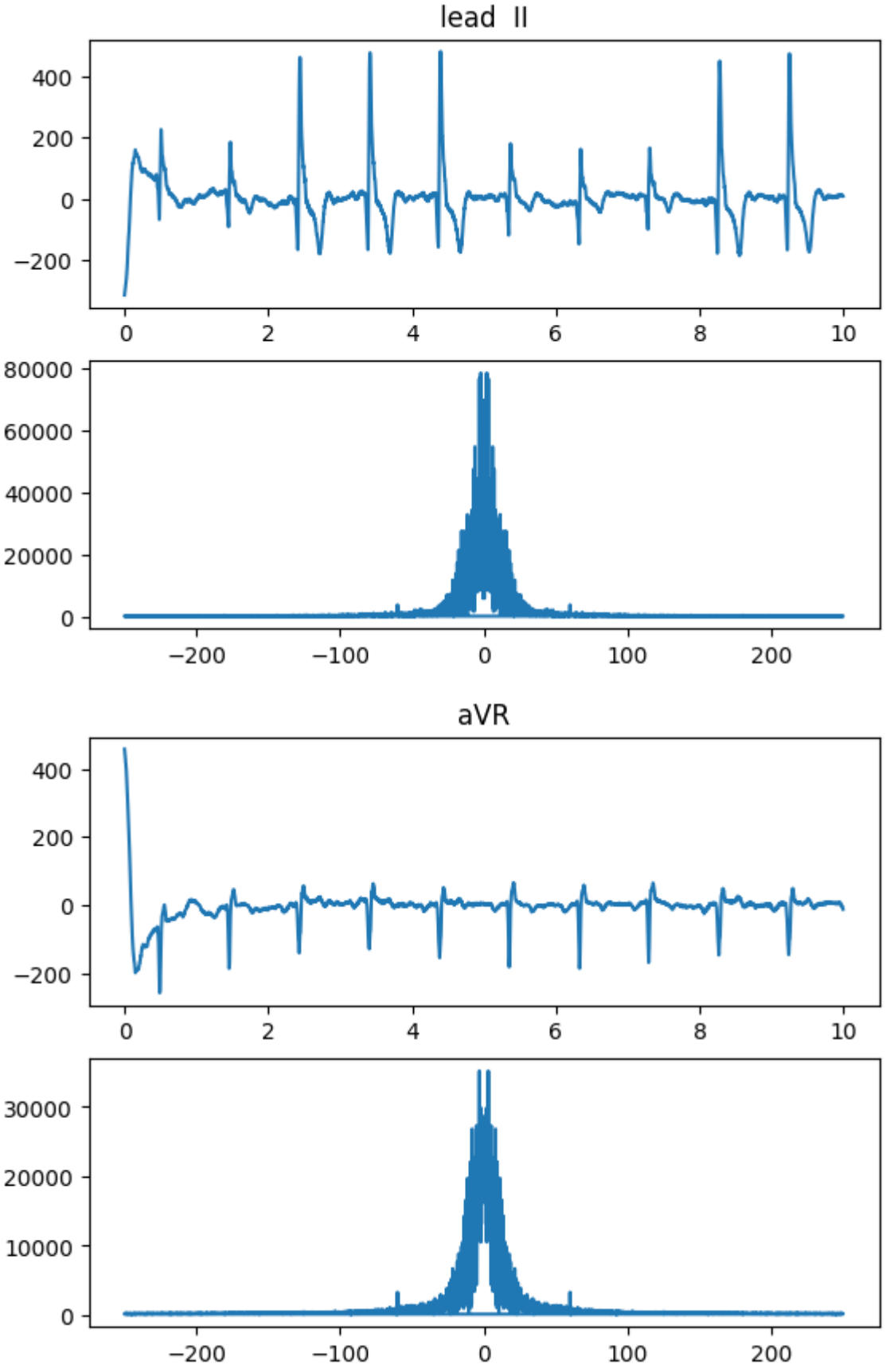
In [ ]:
```
for derivation in header[1::]:
    derivationFourierPlot(dataset, derivation)
```

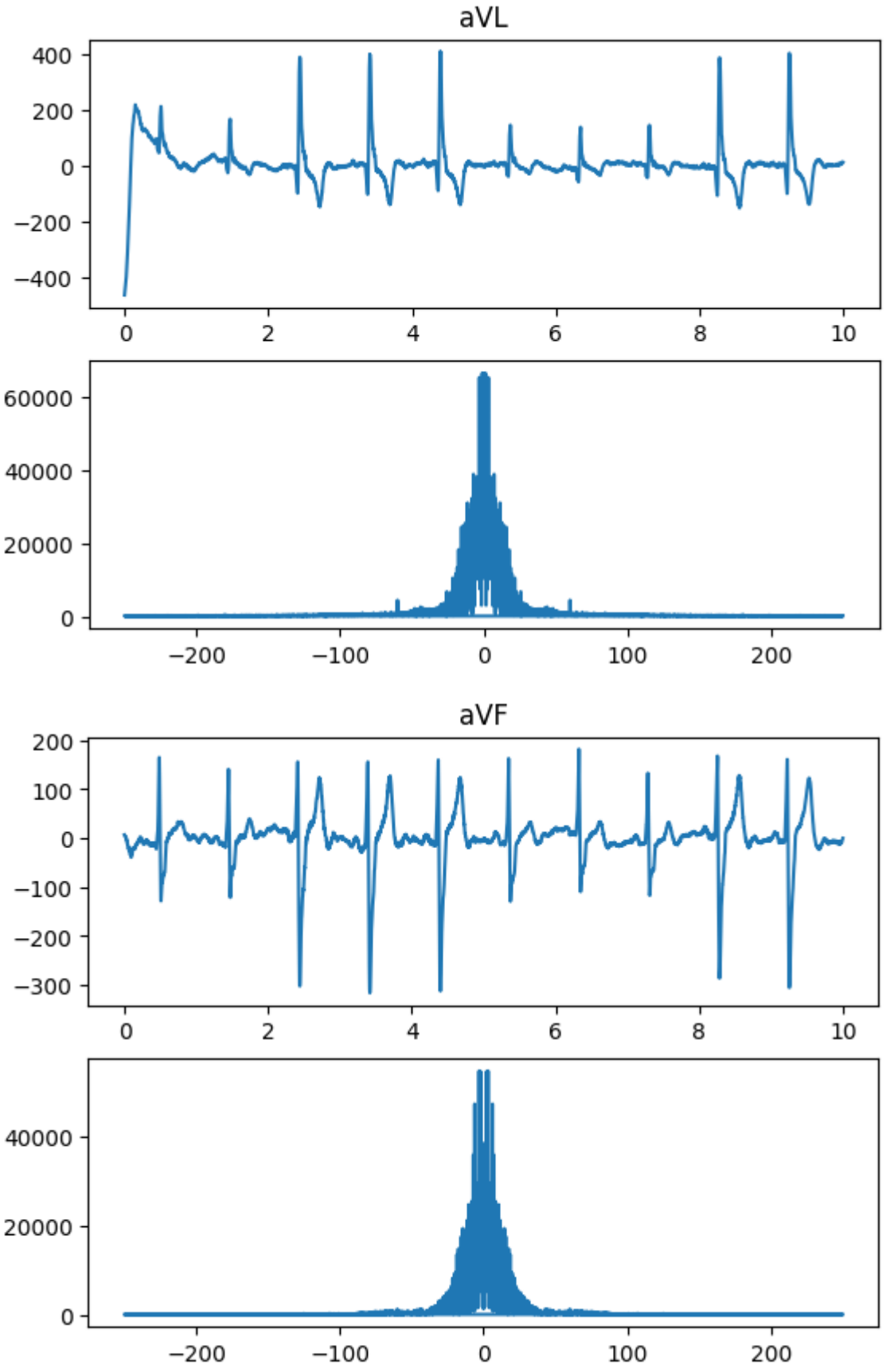

lead I
200
0
−200
0
2
4
6
8
10

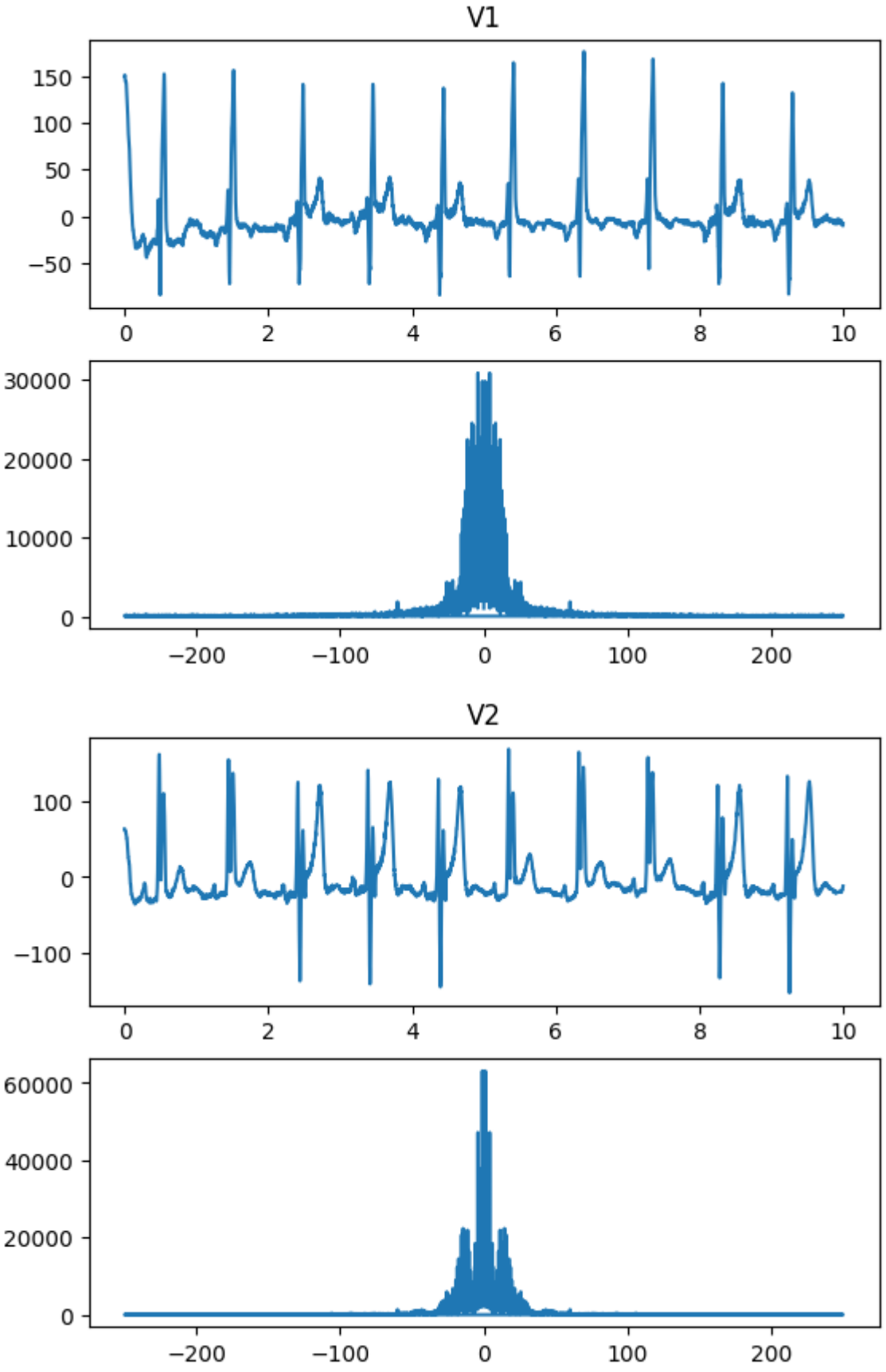

40000
30000
20000
10000
0
−200
−100
0
100
200

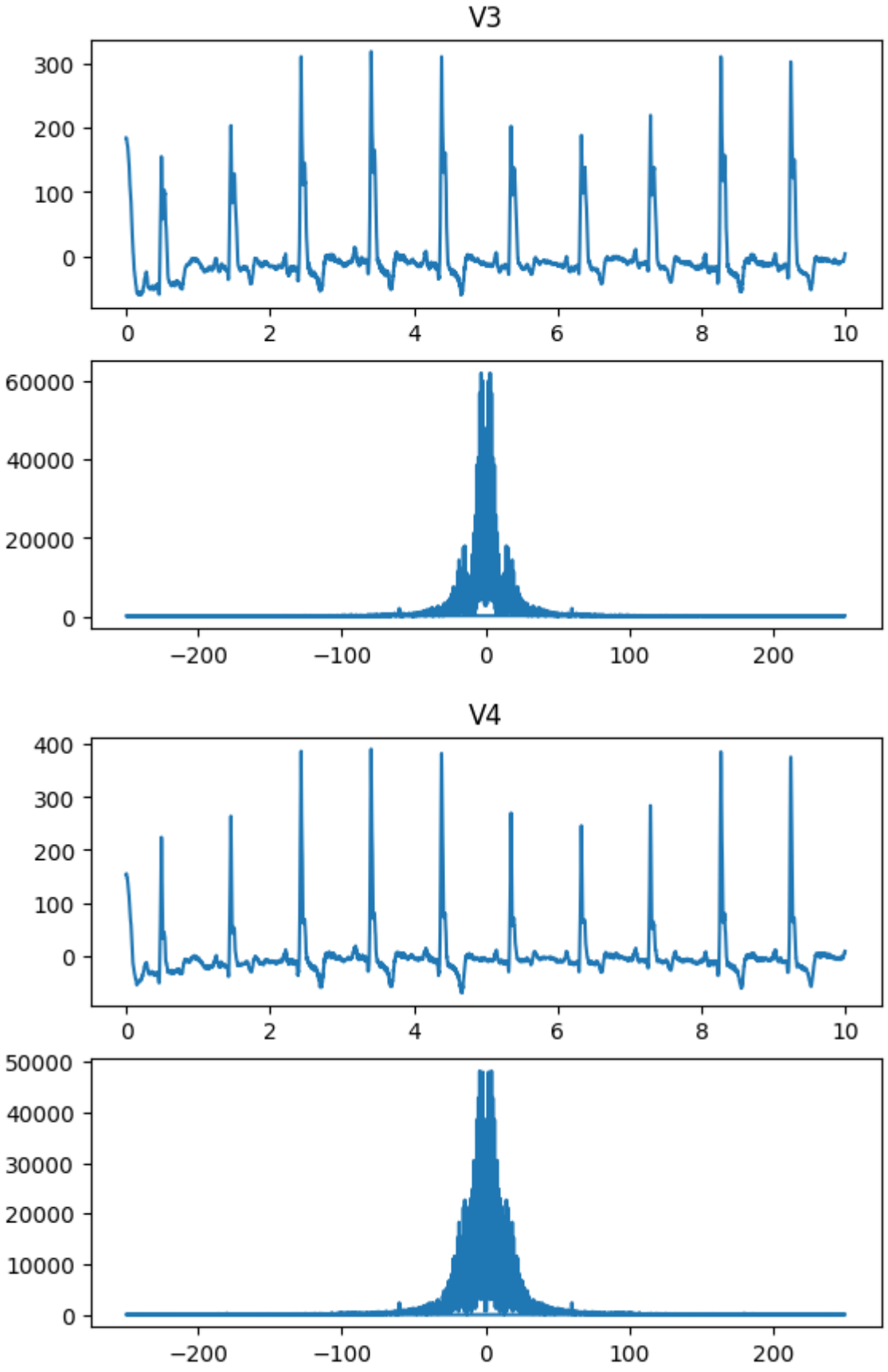

lead II
200
0
−200
−400
−600
0
2
4
6
8
10

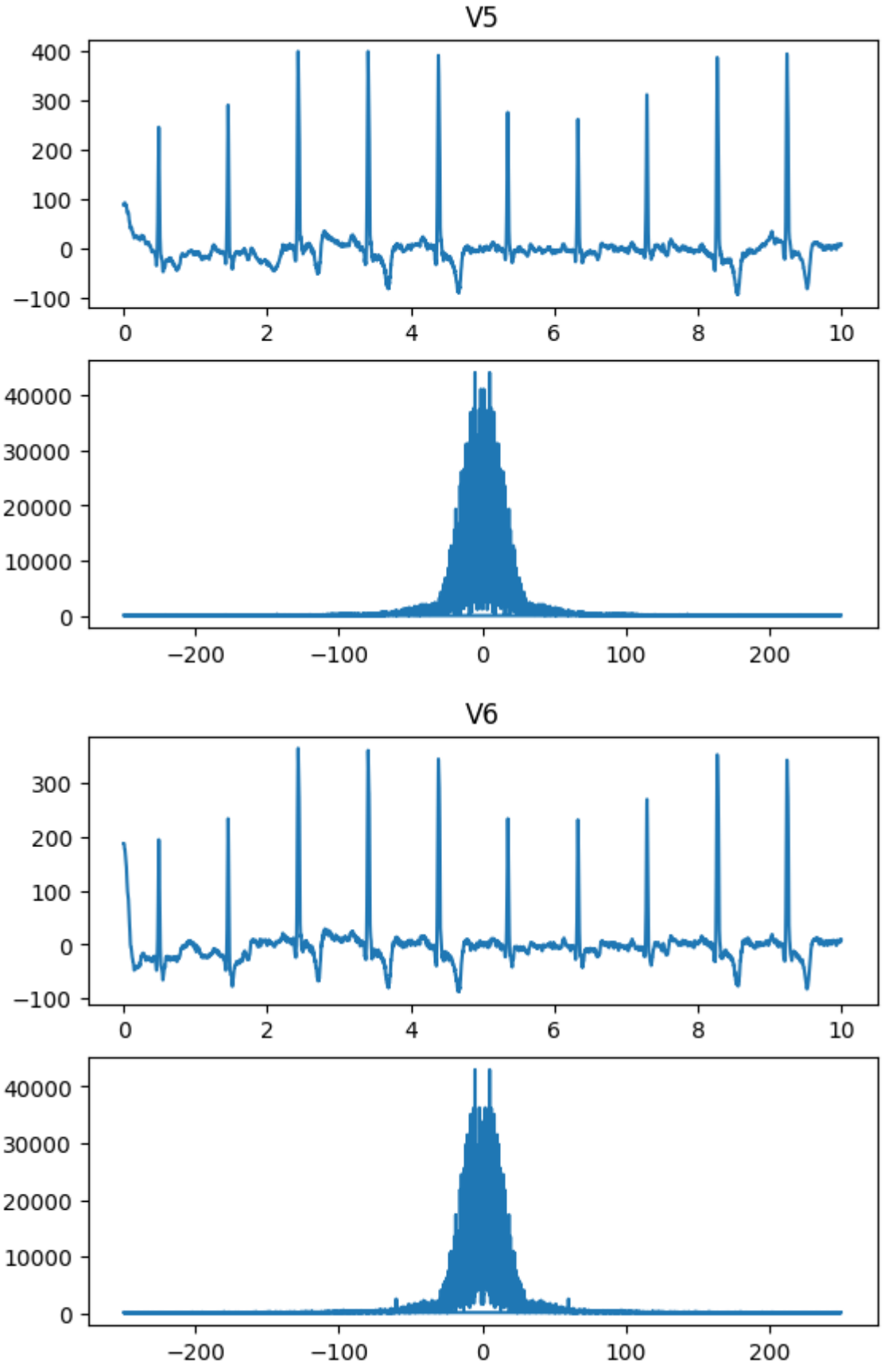

60000
40000
20000
0
−200
−100
0
100
200


lead II
400
200
0
−200
0
2
4
6
8
10
80000
60000
40000
20000
0
−200
−100
0
100
200
aVR
400
200
0
−200
0
2
4
6
8
10
30000
20000
10000
0
−200
−100
0
100
200


aVL
400
200
0
−200
−400
0
2
4
6
8
10
60000
40000
20000
0
−200
−100
0
100
200
aVF
200
100
0
−100
−200
−300
0
2
4
6
8
10
40000
20000
0
−200
−100
0
100
200

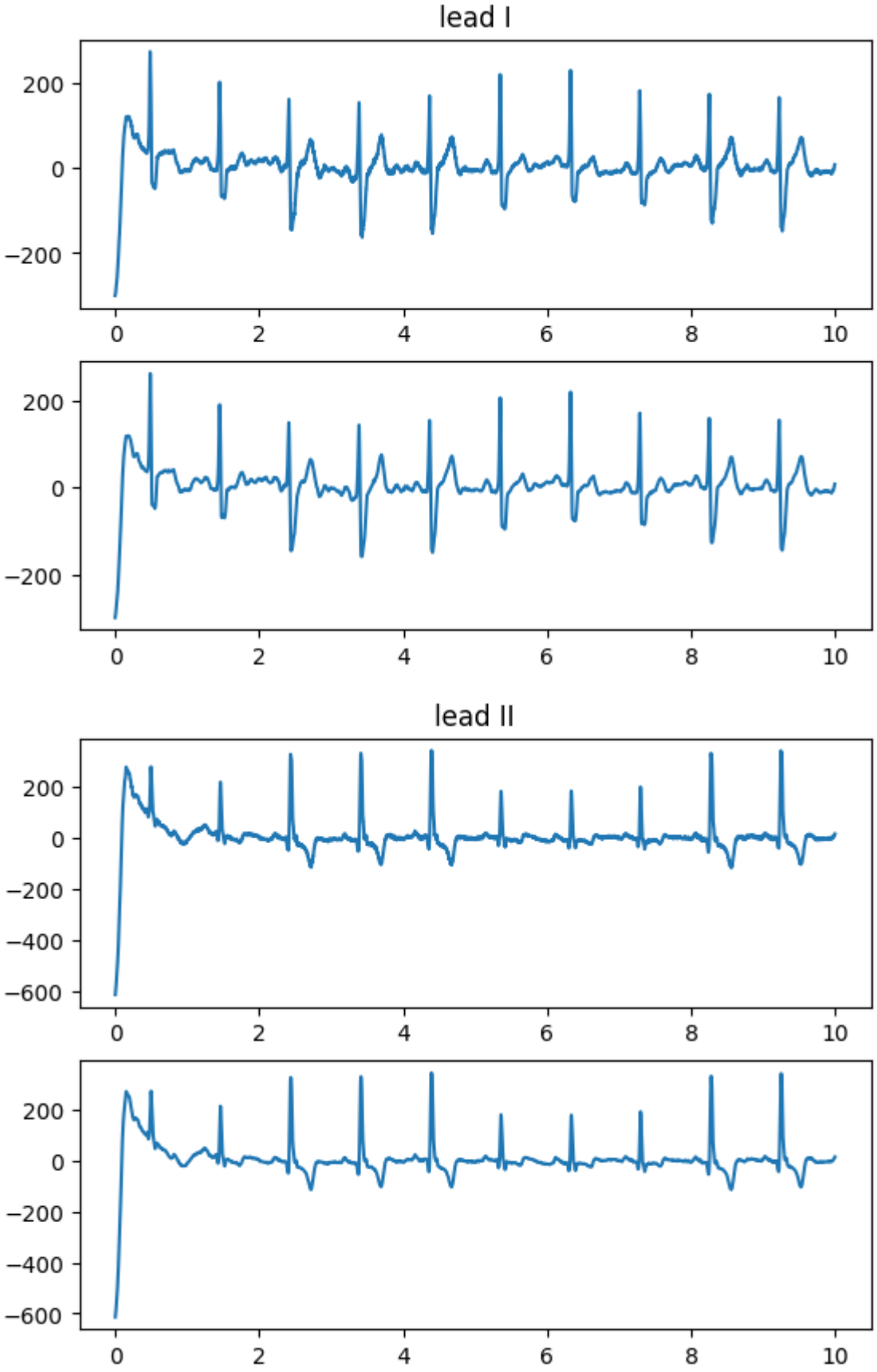

V1
150
100
50
0
−50
0
2
4
6
8
10
30000
20000
10000
0
−200
−100
0
100
200
V2
100
0
−100
0
2
4
6
8
10
60000
40000
20000
0
−200
−100
0
100
200

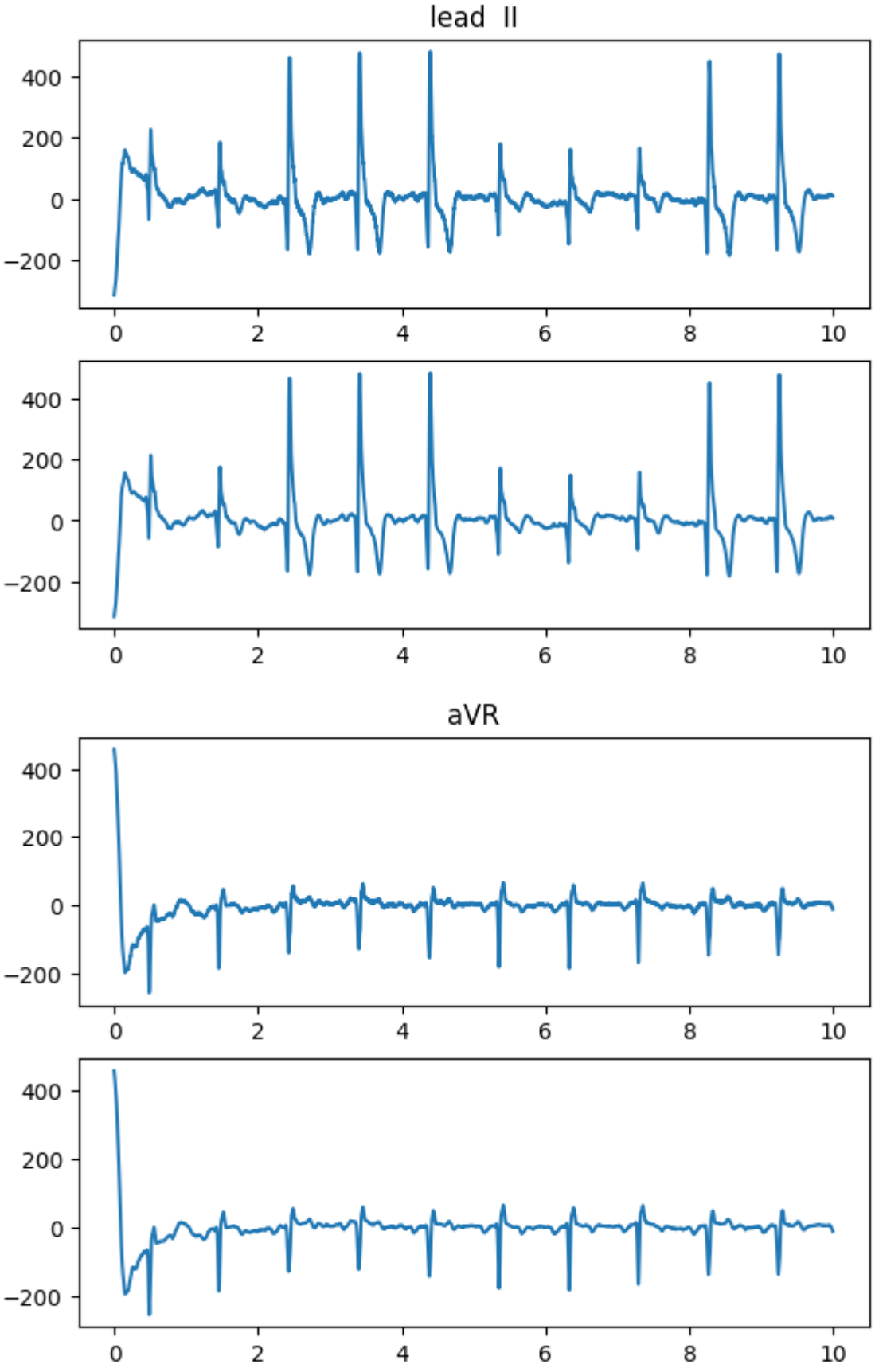

V3
300
200
100
0
0
2
4
6
8
10
60000
40000
20000
0
−200
−100
0
100
200
V4
400
300
200
100
0
0
2
4
6
8
10
50000
40000
30000
20000
10000
0
−200
−100
0
100
200

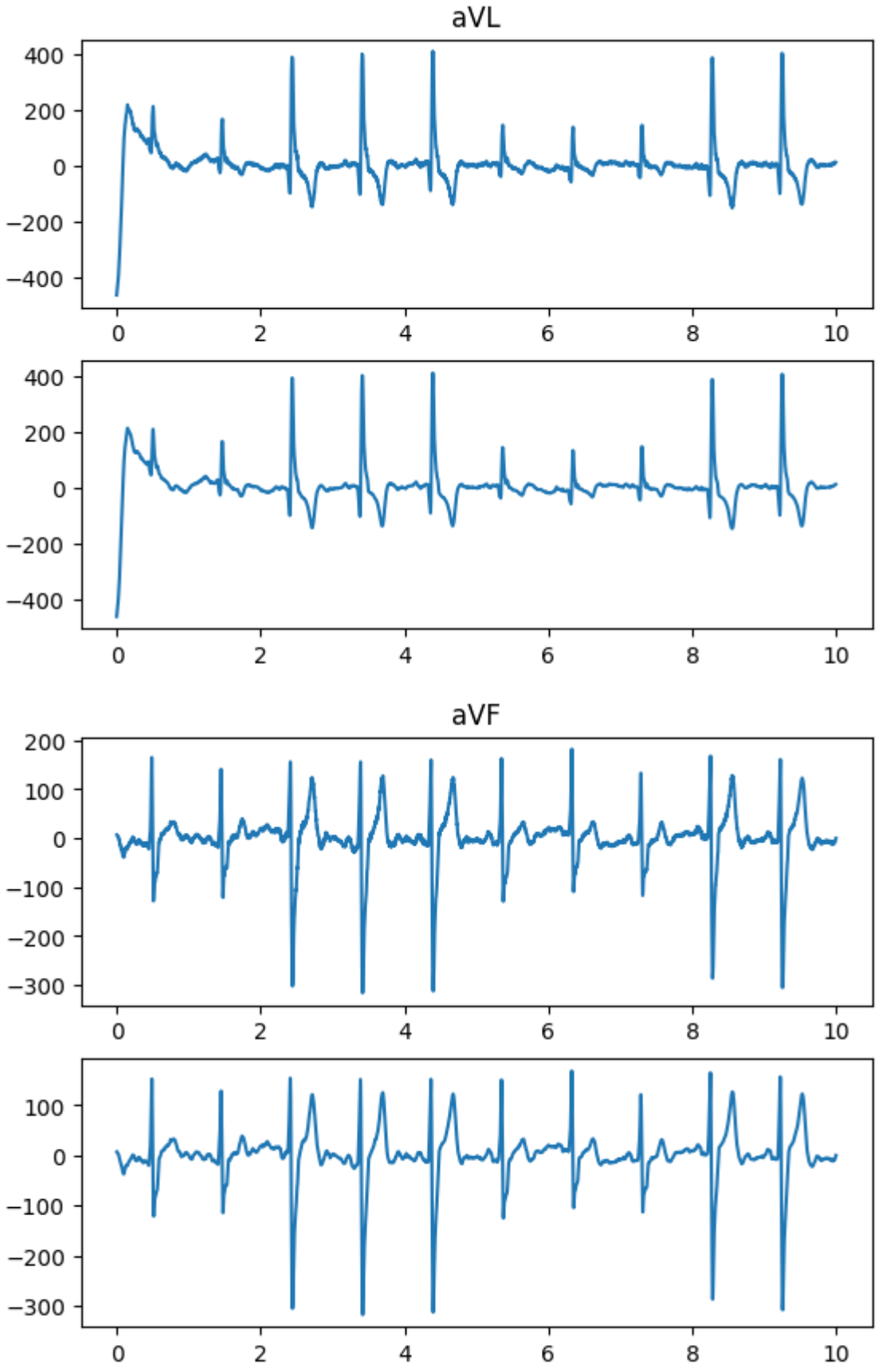


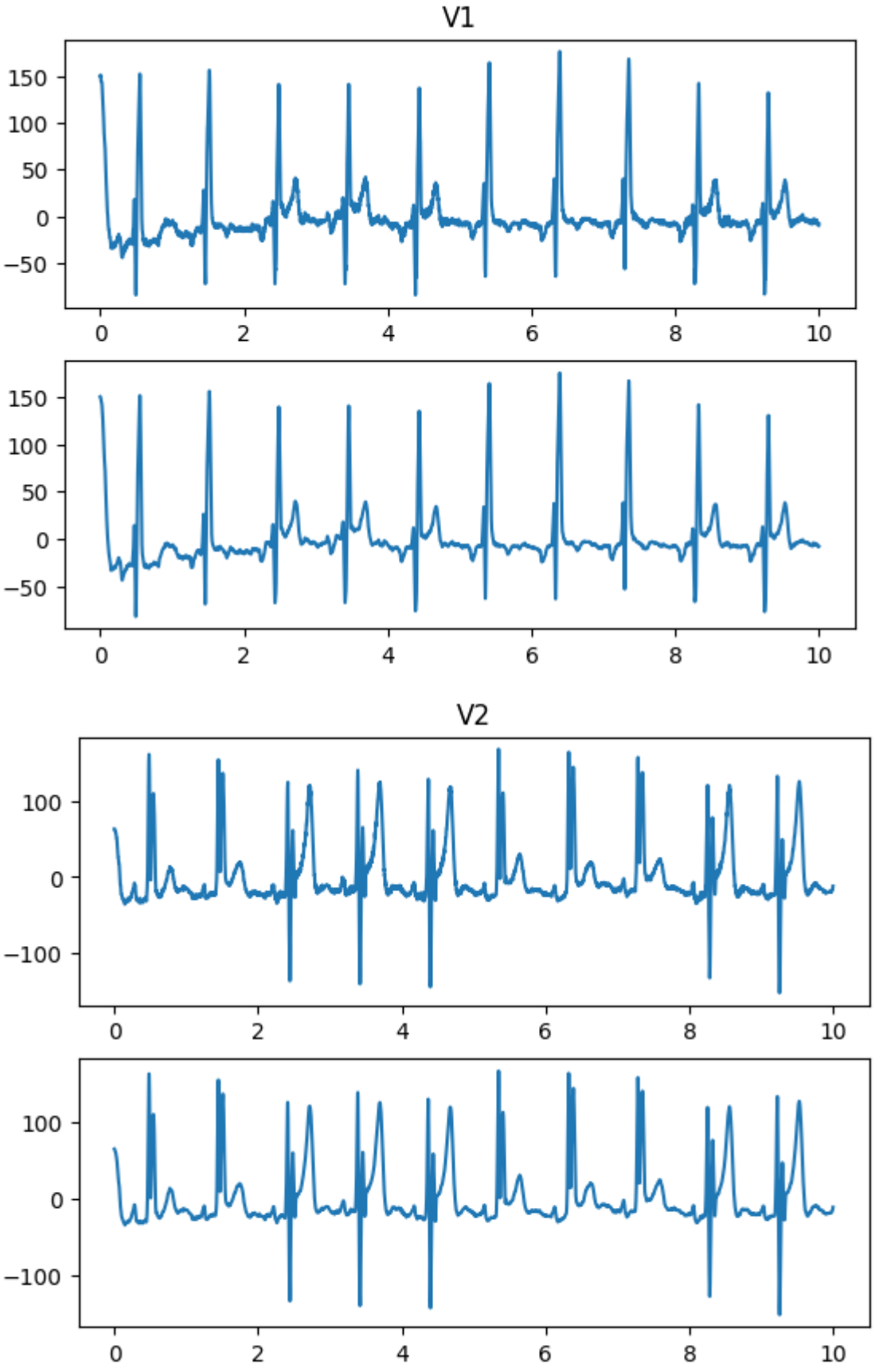
Derivações com Filtro Passa-Baixa com frequência de corte de 50hz

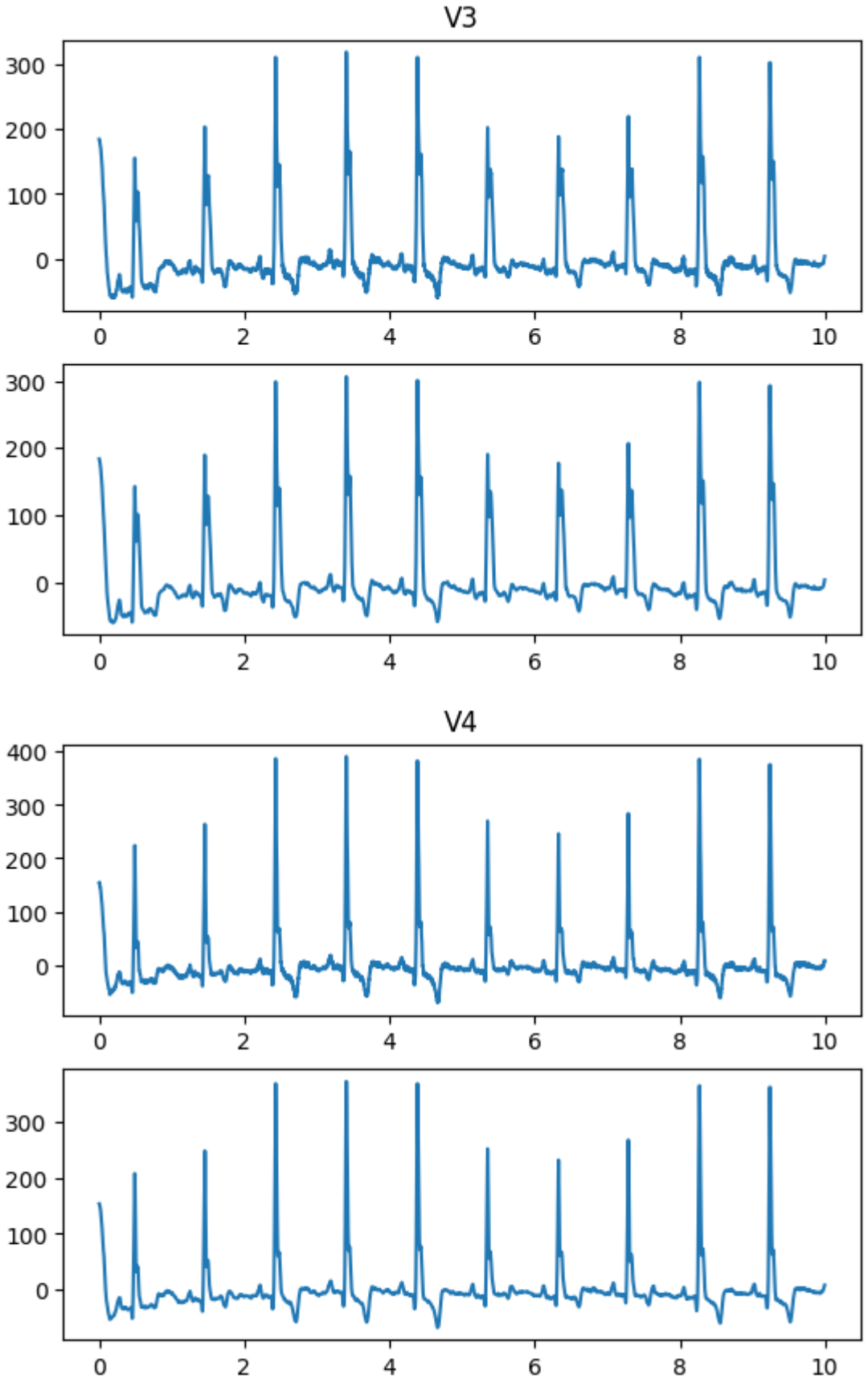
In [ ]:
```
for derivation in header[1::]:
    derivationLowPassPlot(dataset, derivation, 50)
```

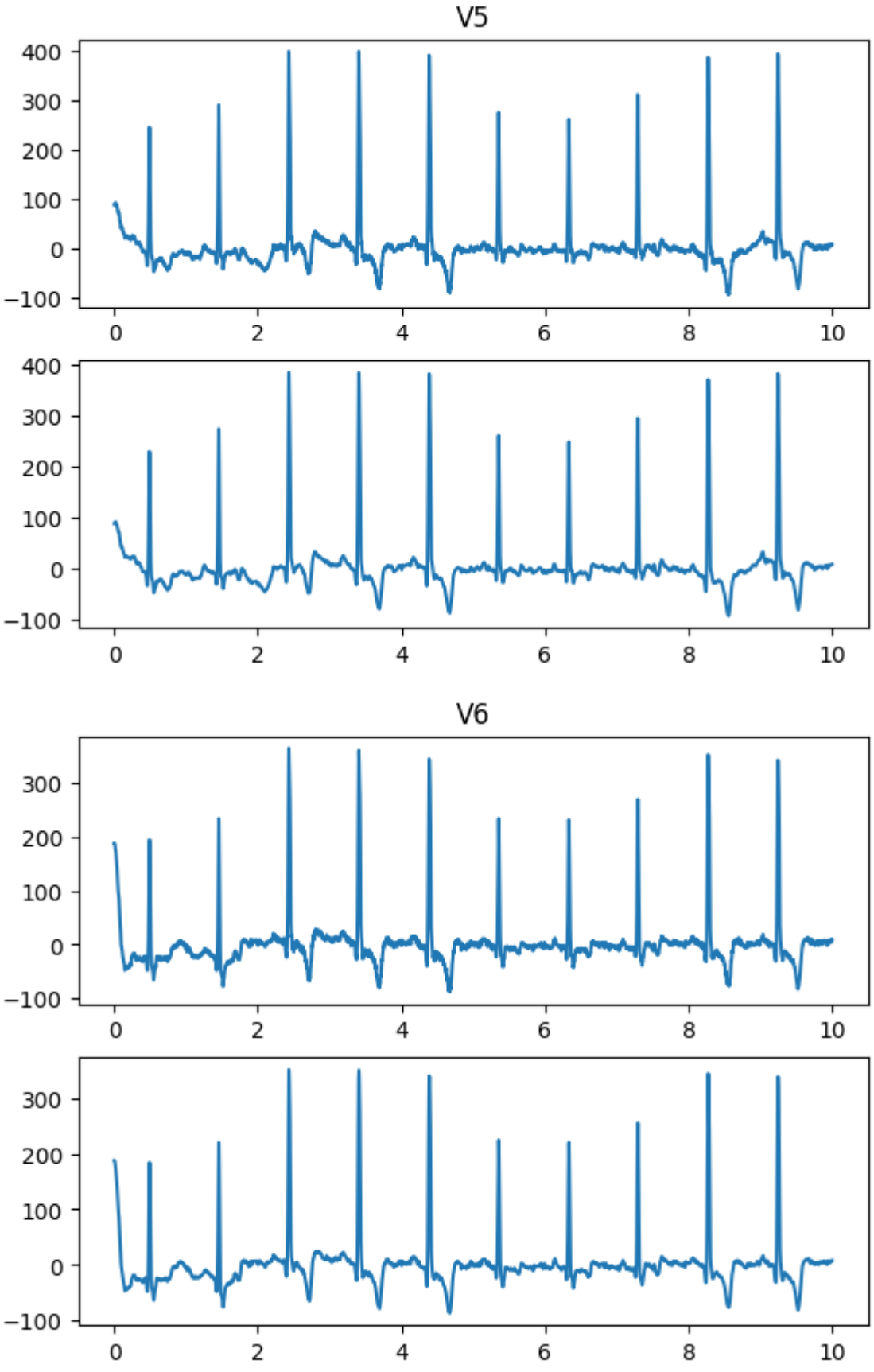

lead I
200
0
−200
0
2
4
6
8
10
200
0
−200
0
2
4
6
8
10
lead II
200
0
−200
−400
−600
0
2
4
6
8
10
200
0
−200
−400
−600
0
2
4
6
8
10


lead II
400
200
0
−200
0
2
4
6
8
10
aVR


aVL
400
200
0
−200
−400
0
2
4
6
8
10
400
200
0
−200
−400
0
2
4
6
8
10
aVF
200
100
0
−100
−200
−300
0
2
4
6
8
10
100
0
−100
−200
−300
0
2
4
6
8
10

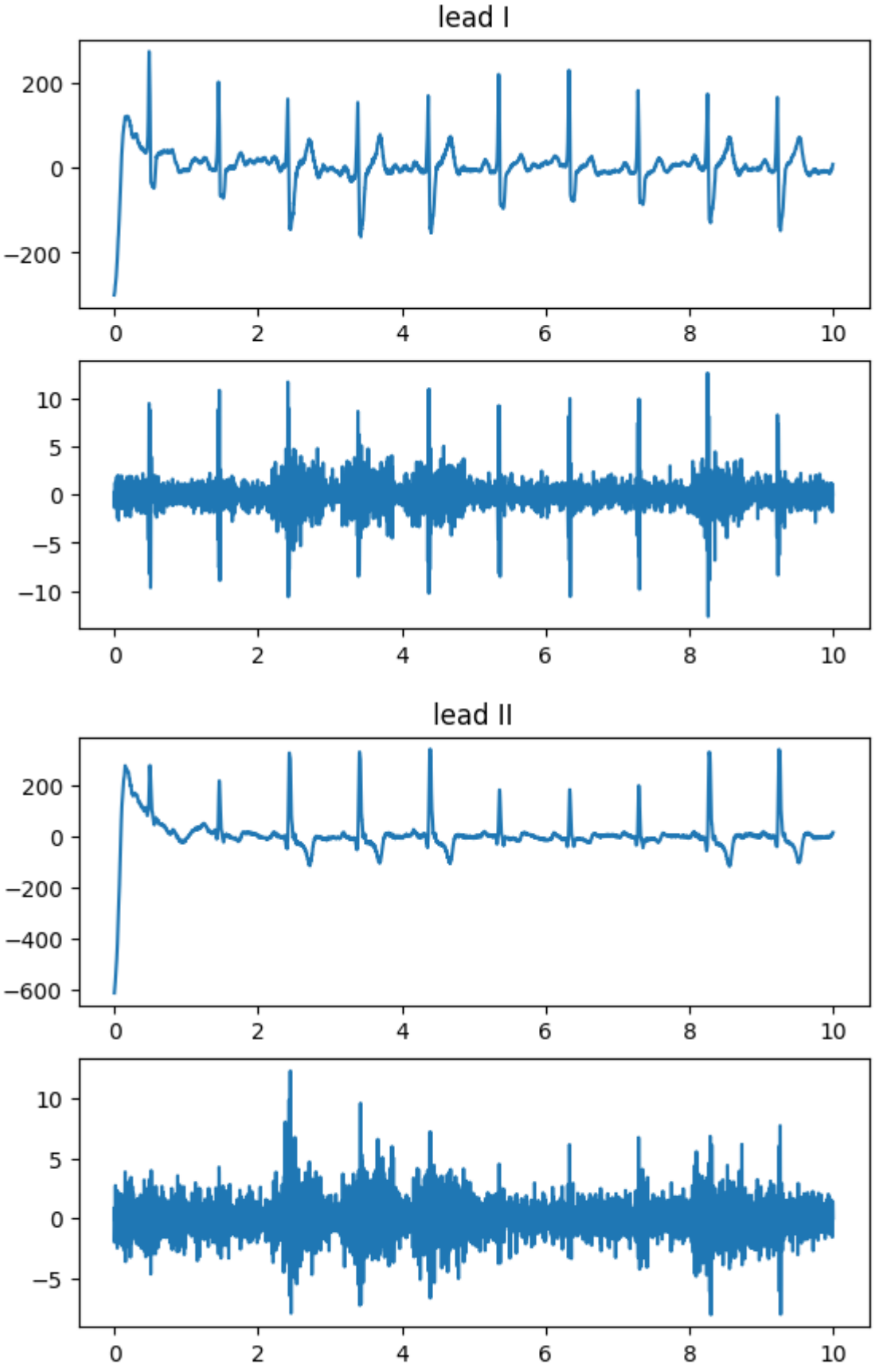

V1
150
100
50
0
−50
0
2
4
6
8
10
V2
100
0
−100

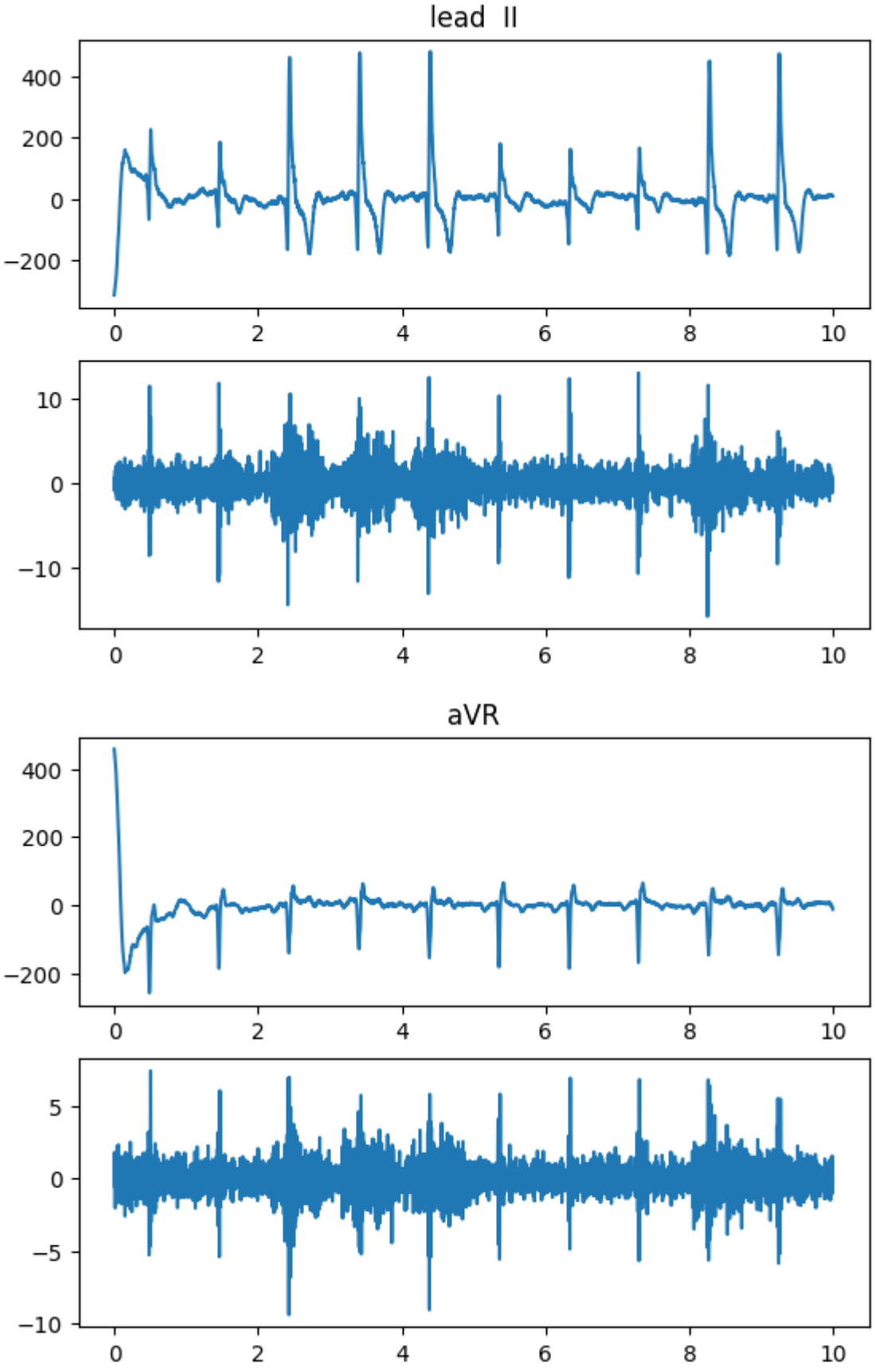

V3
300
200
100
0
0
2
4
6
8
10
300
200
100
0
0
2
4
6
8
10
V4
400
300
200
100
0
0
2
4
6
8
10
300
200
100
0
0
2
4
6
8
10

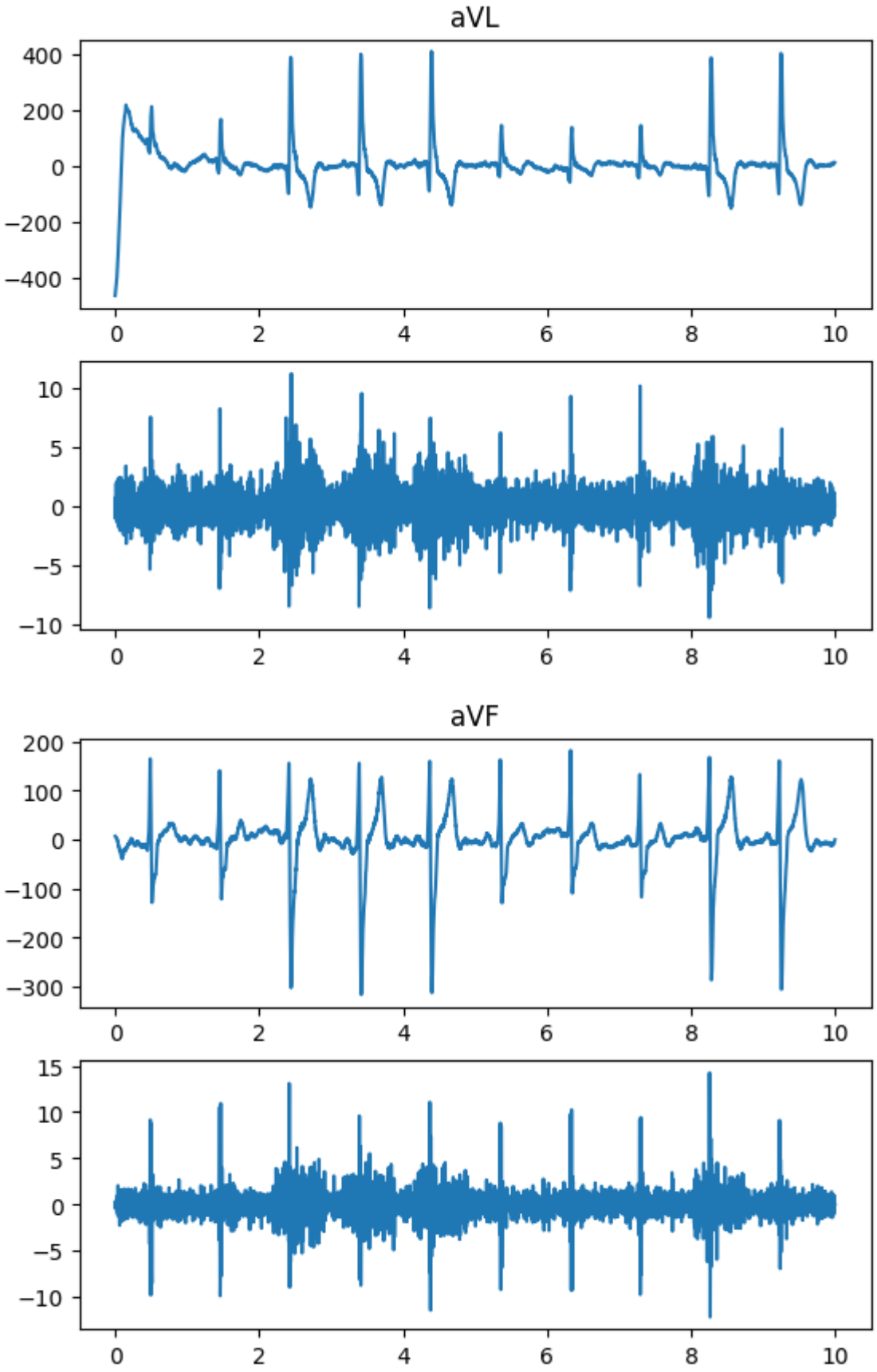


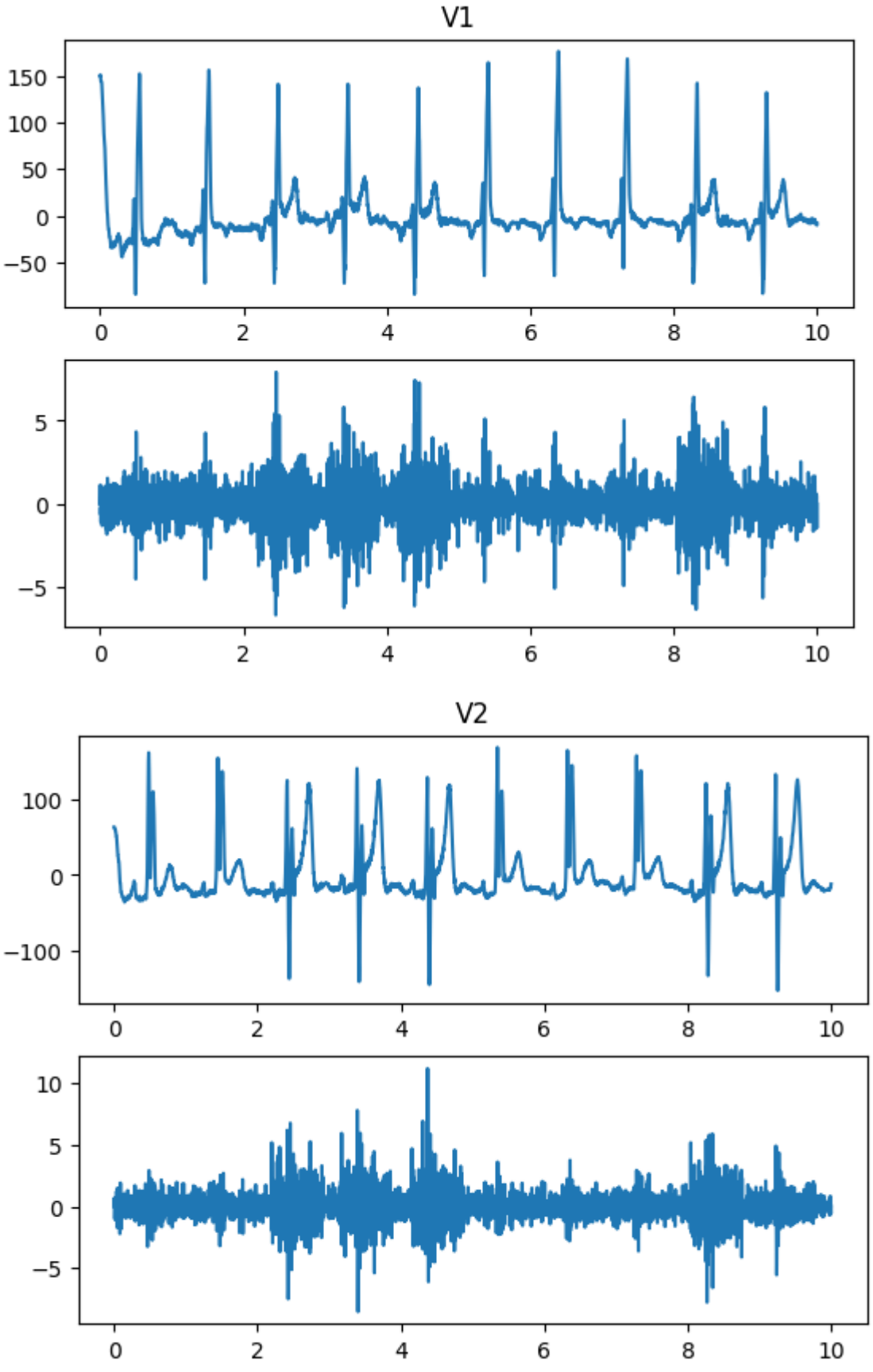
Derivações com Filtro Passa-Alta com frequência de corte de 60hz

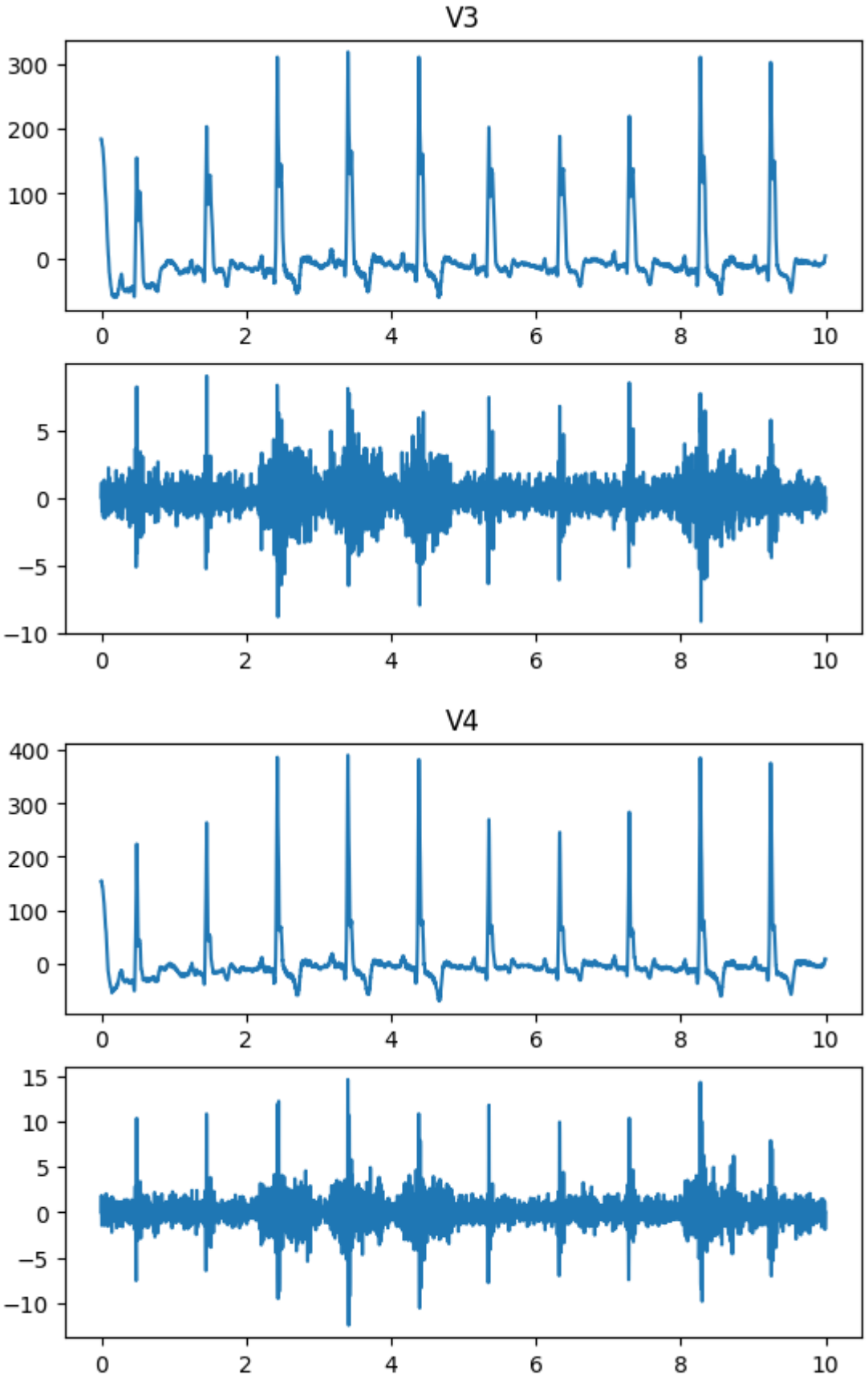
In [ ]:
```
for derivation in header[1::]:
    derivationHighPassPlot(dataset, derivation, 60)
```

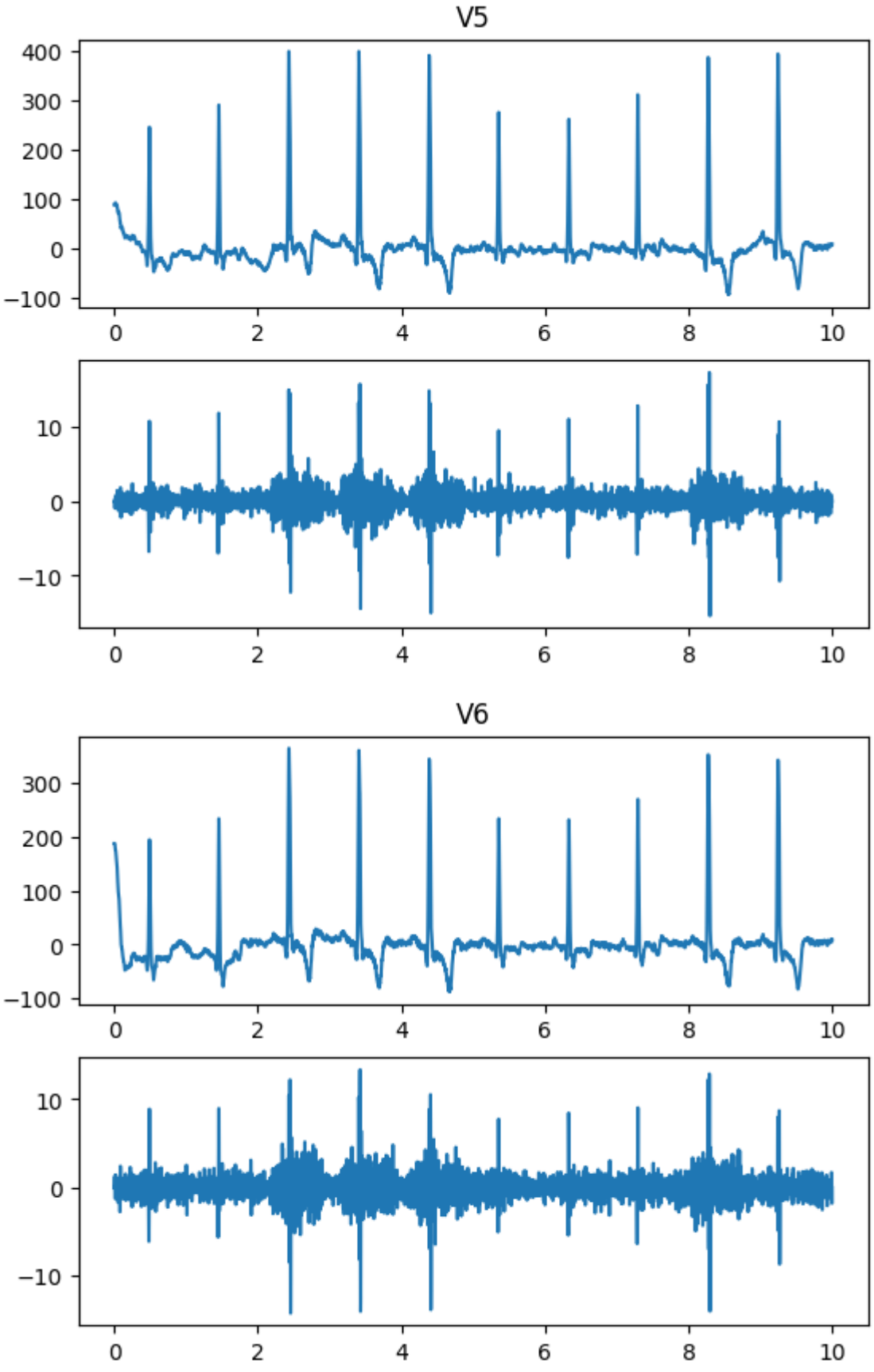

lead I
200
0
−200
0
2
4
6
8
10
10
5
0
−5
−10
0
2
4
6
8
10
lead II
200
0
−200
−400
−600
0
2
4
6
8
10
10
5
0
−5
0
2
4
6
8
10


lead II
400
200
0
−200
0
2
4
6
8
10
10
0
−10
0
2
4
6
8
10
aVR
400
200
0
−200
0
2
4
6
8
10
5
0
−5
−10
0
2
4
6
8
10


aVL
400
200
0
−200
−400
0
2
4
6
8
10
10
5
0
−5
−10
0
2
4
6
8
10
aVF
200
100
0
−100
−200
−300
0
2
4
6
8
10
15
10
5
0
−5
−10
0
2
4
6
8
10

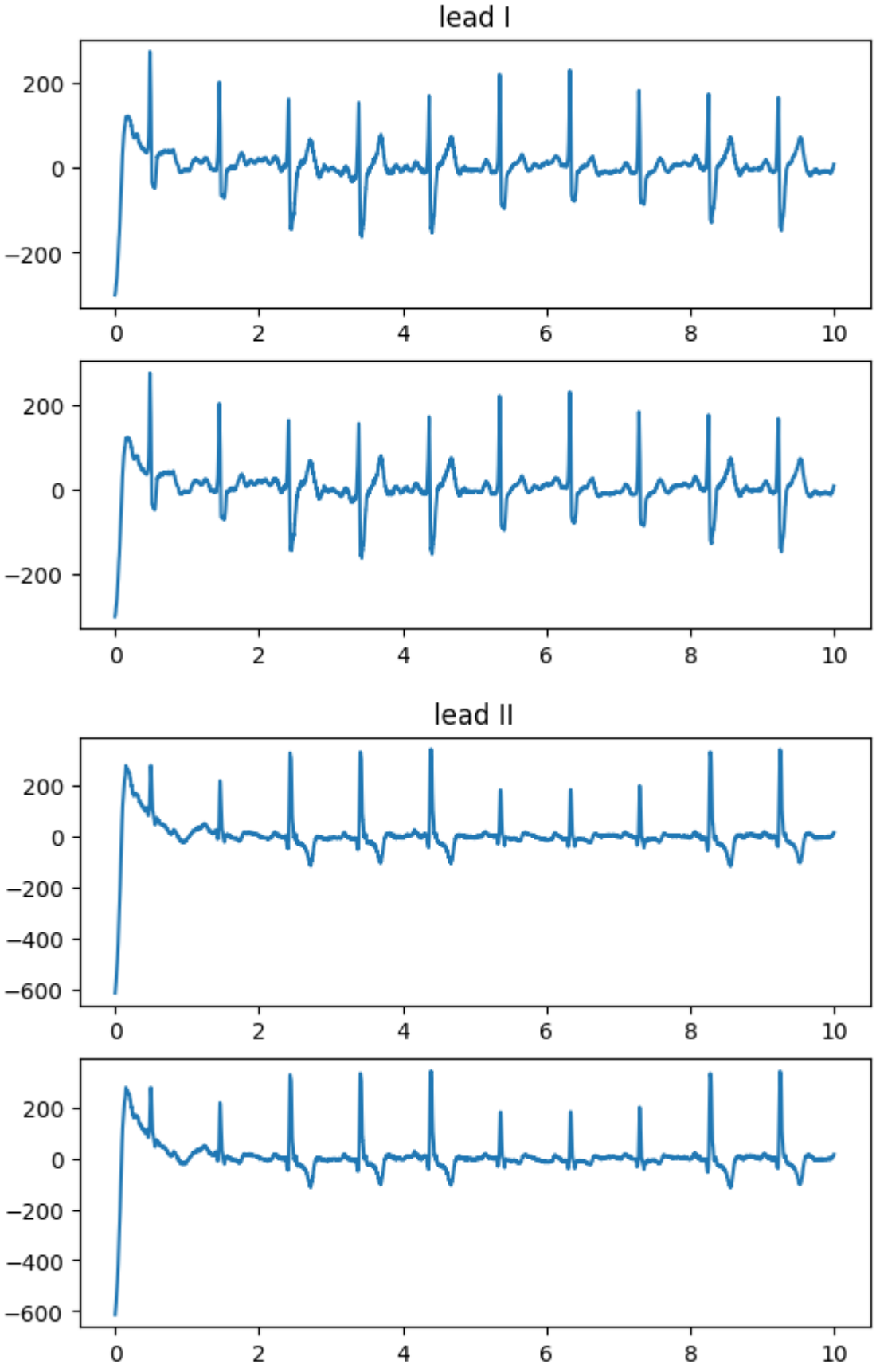

V1
150
100
50
0
−50
0
2
4
6
8
10
5
0
−5
0
2
4
6
8
10
V2
100
0
−100
0
2
4
6
8
10
10
5
0
−5
0
2
4
6
8
10

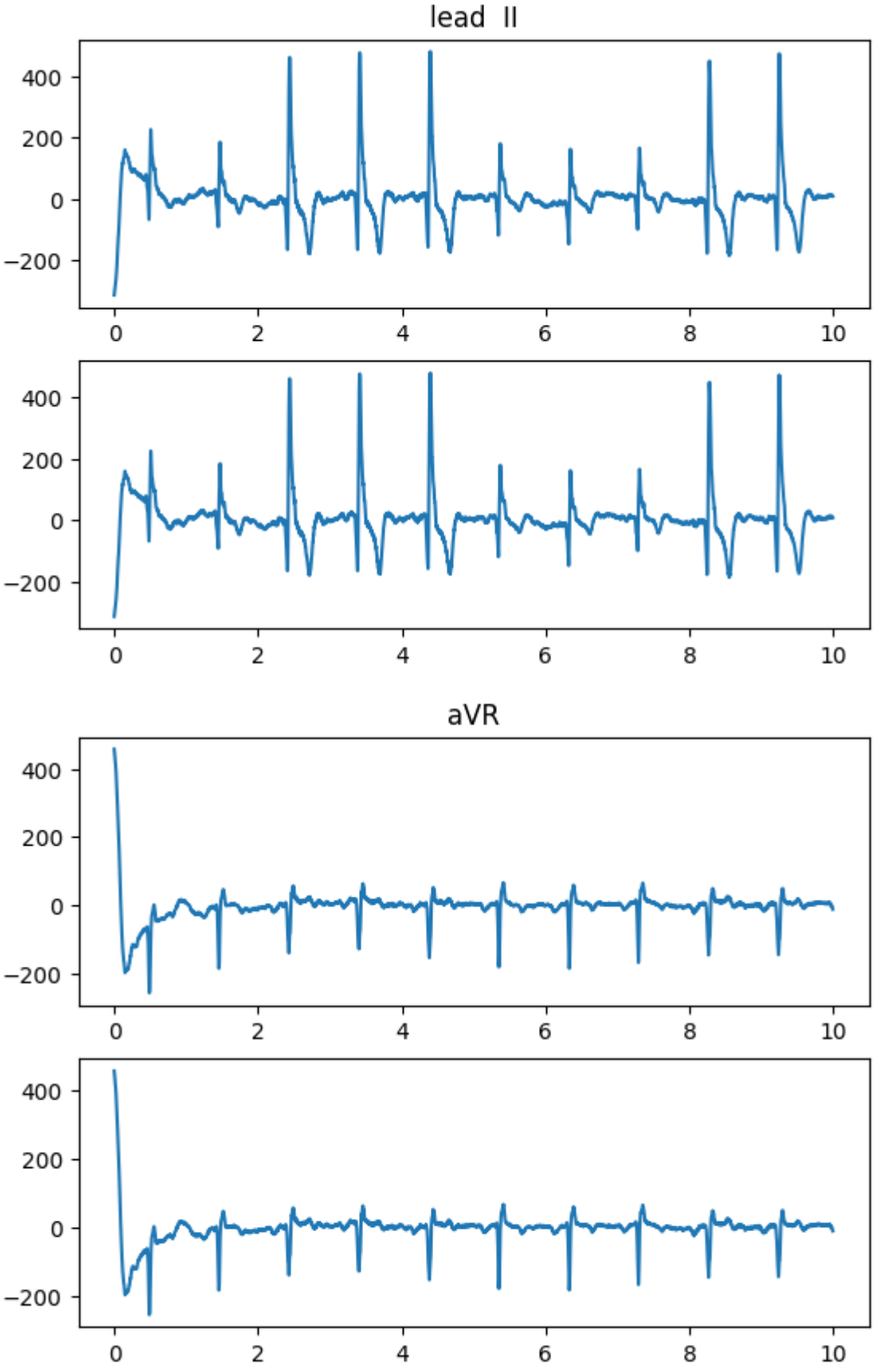

V3
300
200
100
0
0
2
4
6
8
10
5
0
−5
−10
0
2
4
6
8
10
V4
400
300
200
100
0
0
2
4
6
8
10
15
10
5
0
−5
−10
0
2
4
6
8
10

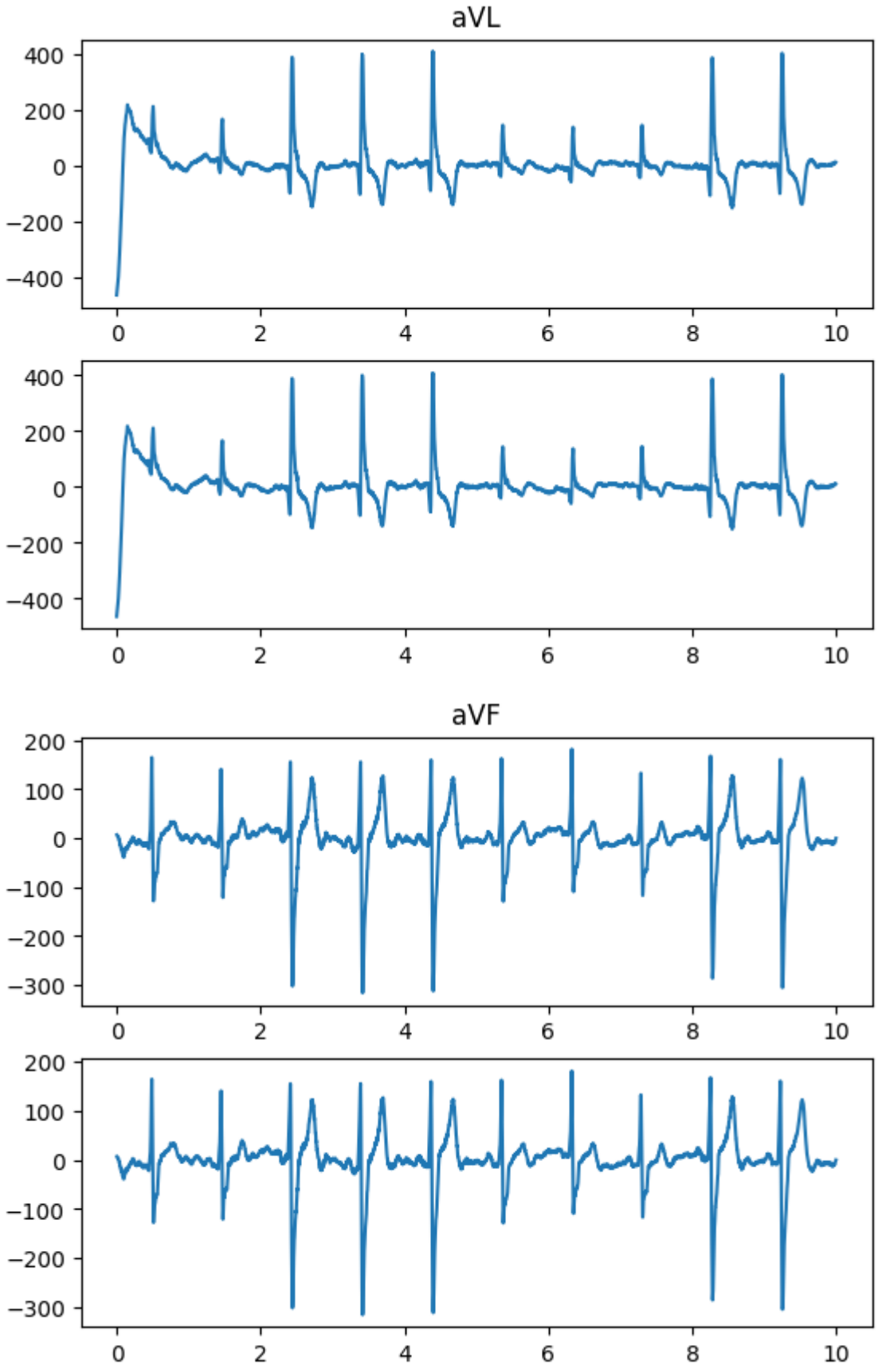


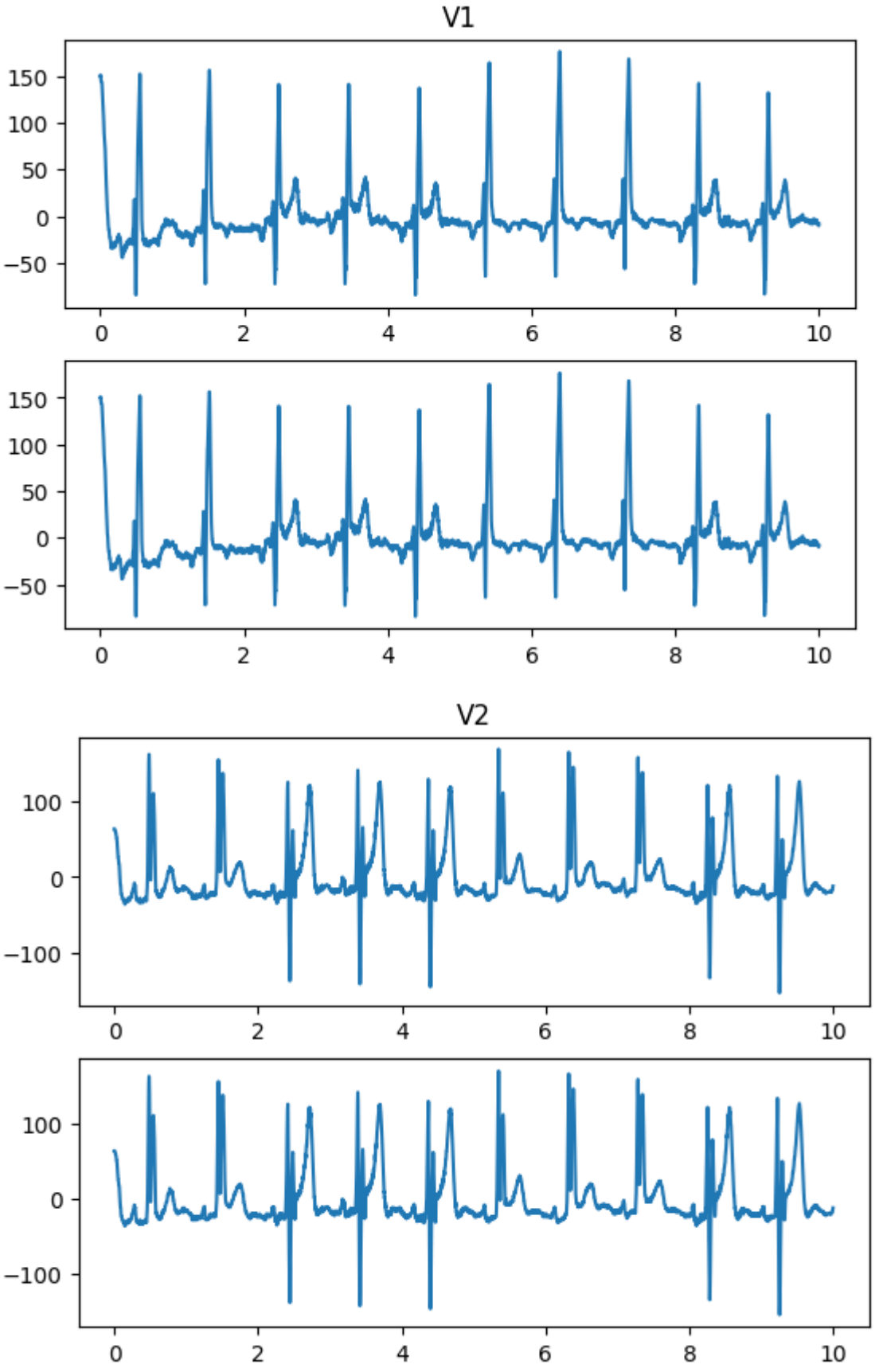
Derivações com Filtro Notch com banda de corte 50 e 60hz

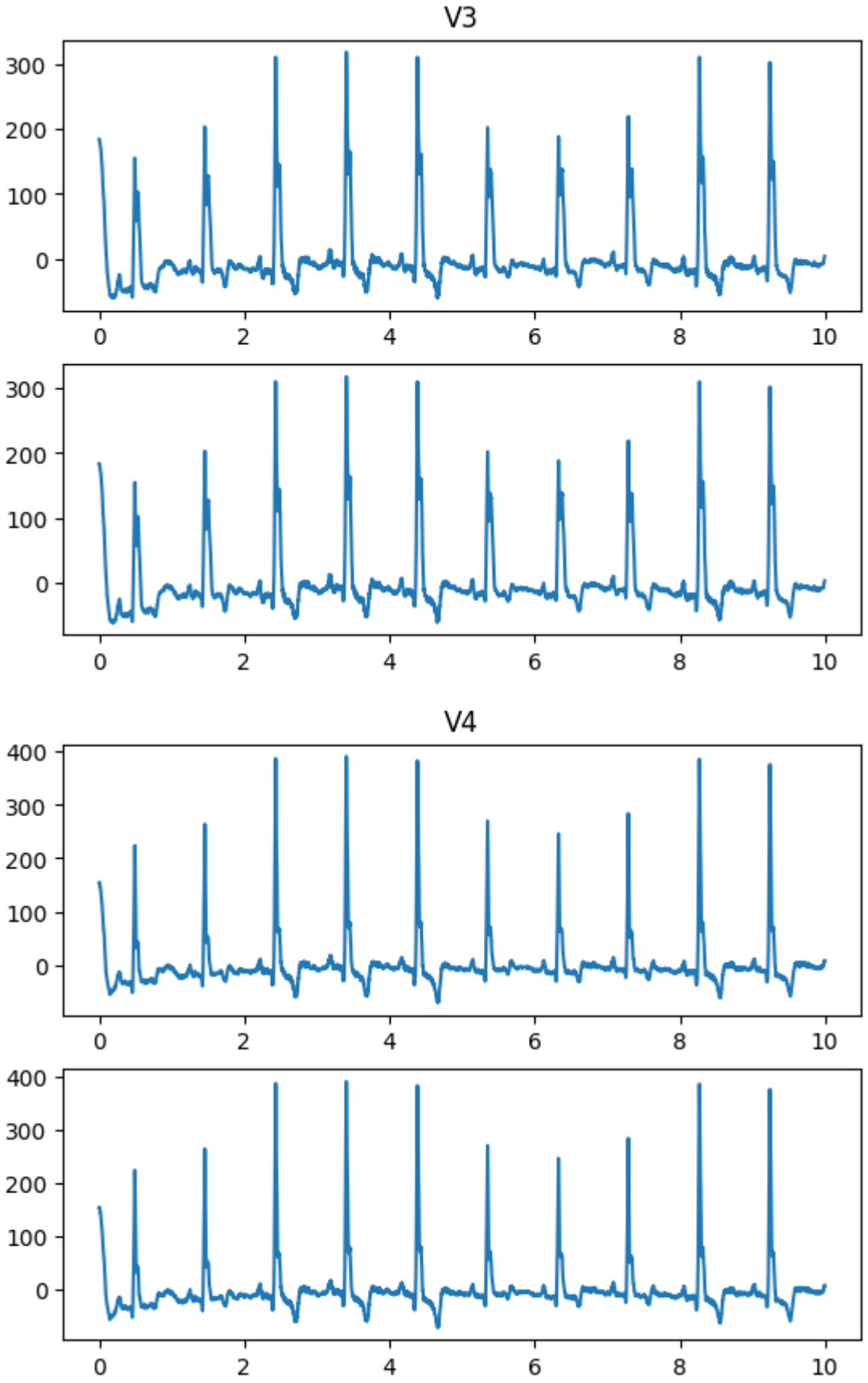
In [ ]:
```
for derivation in header[1::]:
    derivationNocthPlot(dataset, derivation, 50)
```

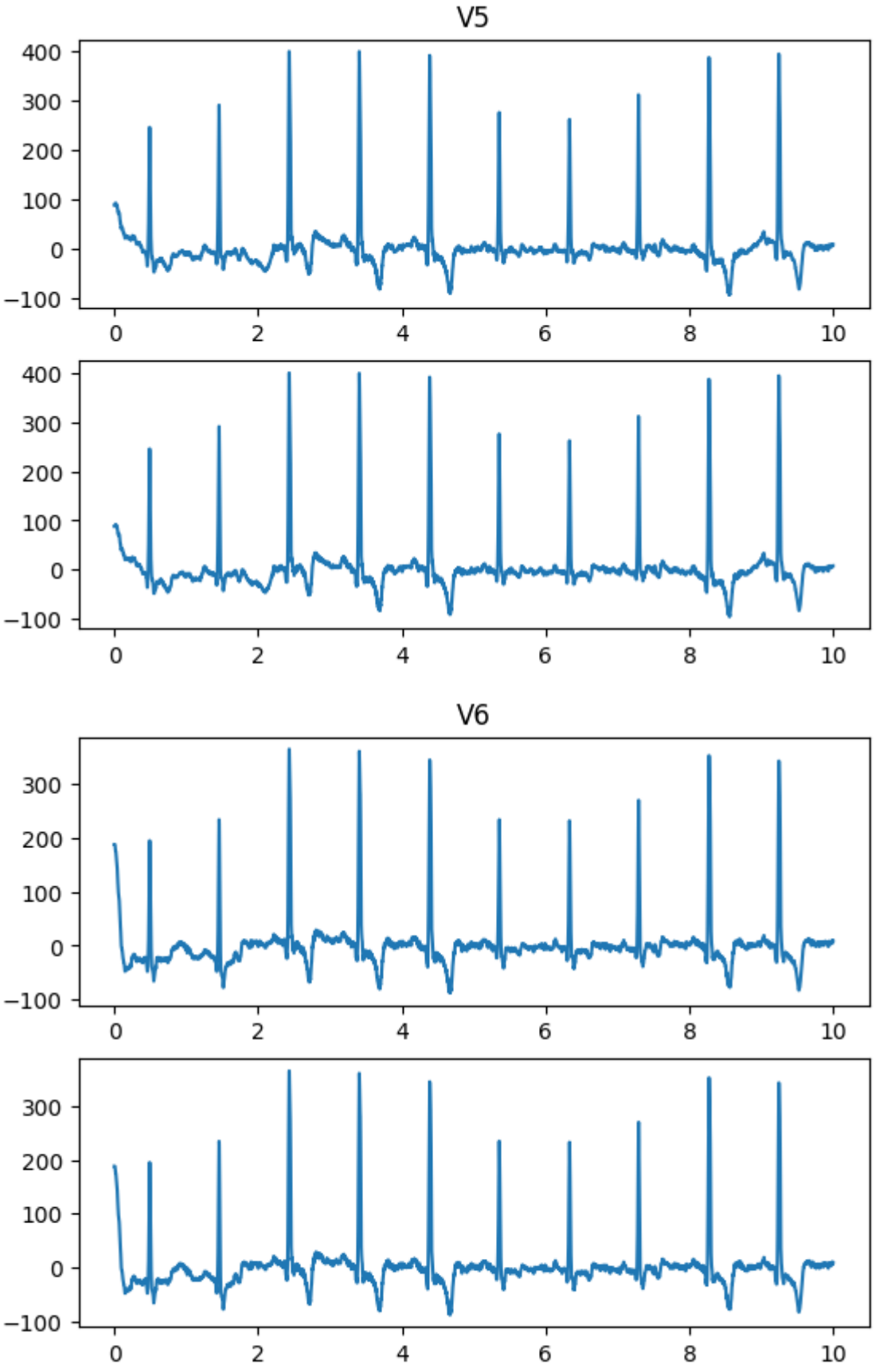

lead I
200
0
−200
0
2
4
6
8
10
200
0
−200
0
2
4
6
8
10
lead II
200
0
−200
−400
−600
0
2
4
6
8
10
200
0
−200
−400
−600
0
2
4
6
8
10


lead II
400
200
0
−200
0
2
4
6
8
10
aVR

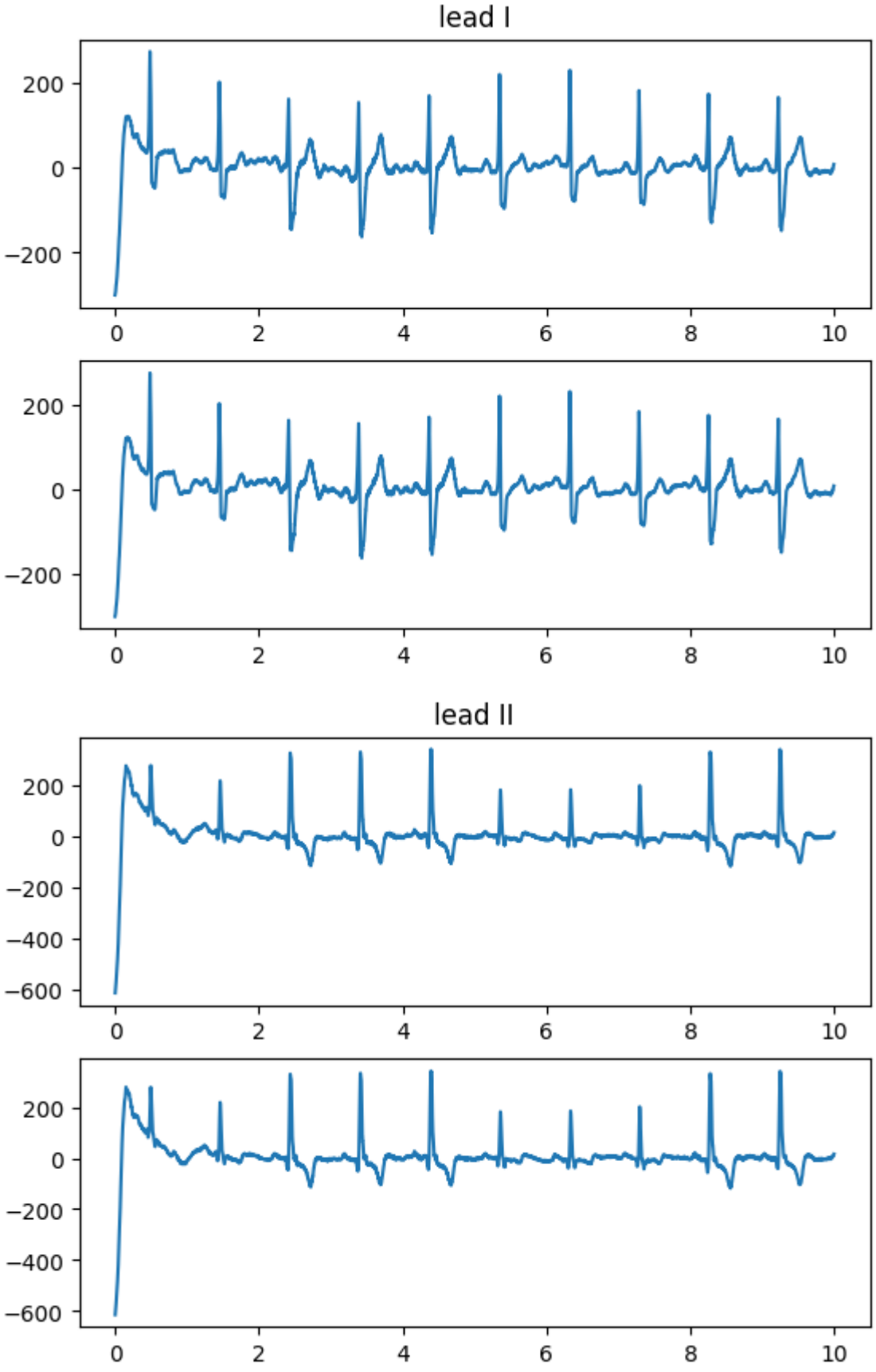

aVL
400
200
0
−200
−400
0
2
4
6
8
10
400
200
0
−200
−400
0
2
4
6
8
10
aVF
200
100
0
−100
−200
−300
0
2
4
6
8
10
200
100
0
−100
−200
−300
0
2
4
6
8
10

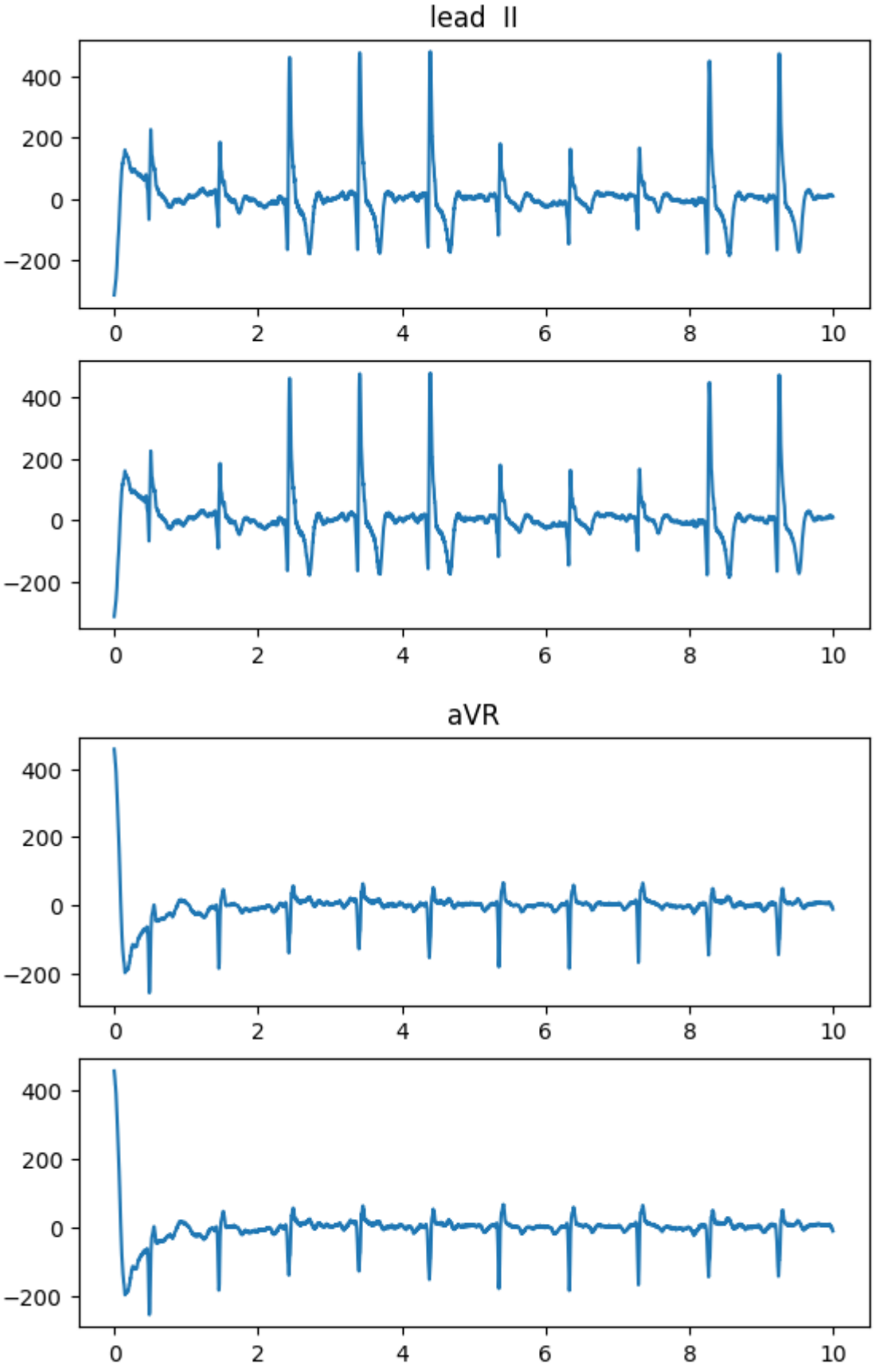

V1
150
100
50
0
−50
0
2
4
6
8
10
V2
100
0
−100

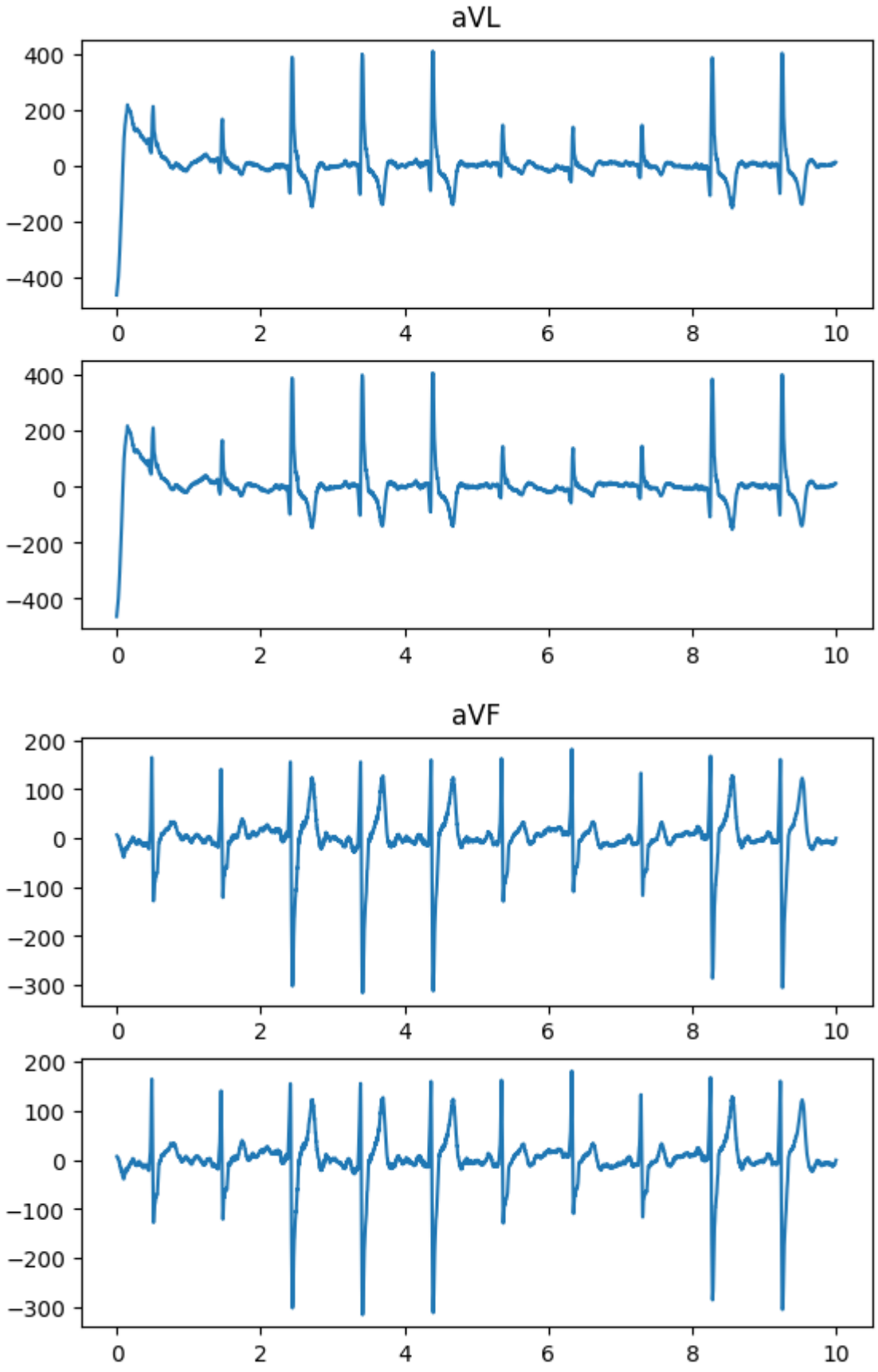

V3
300
200
100
0
0
2
4
6
8
10
300
200
100
0
0
2
4
6
8
10
V4
400
300
200
100
0
0
2
4
6
8
10
400
300
200
100
0
0
2
4
6
8
10

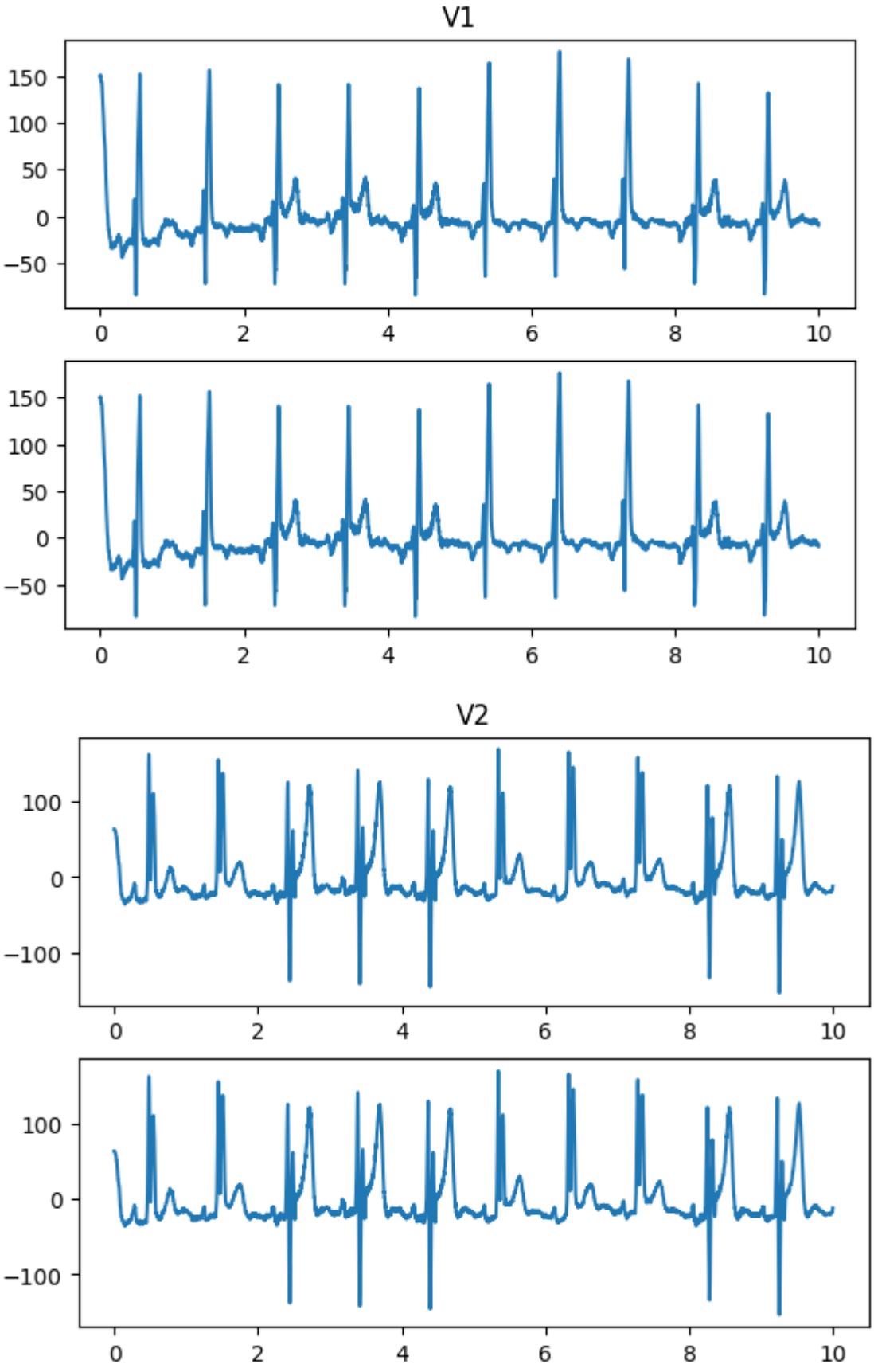


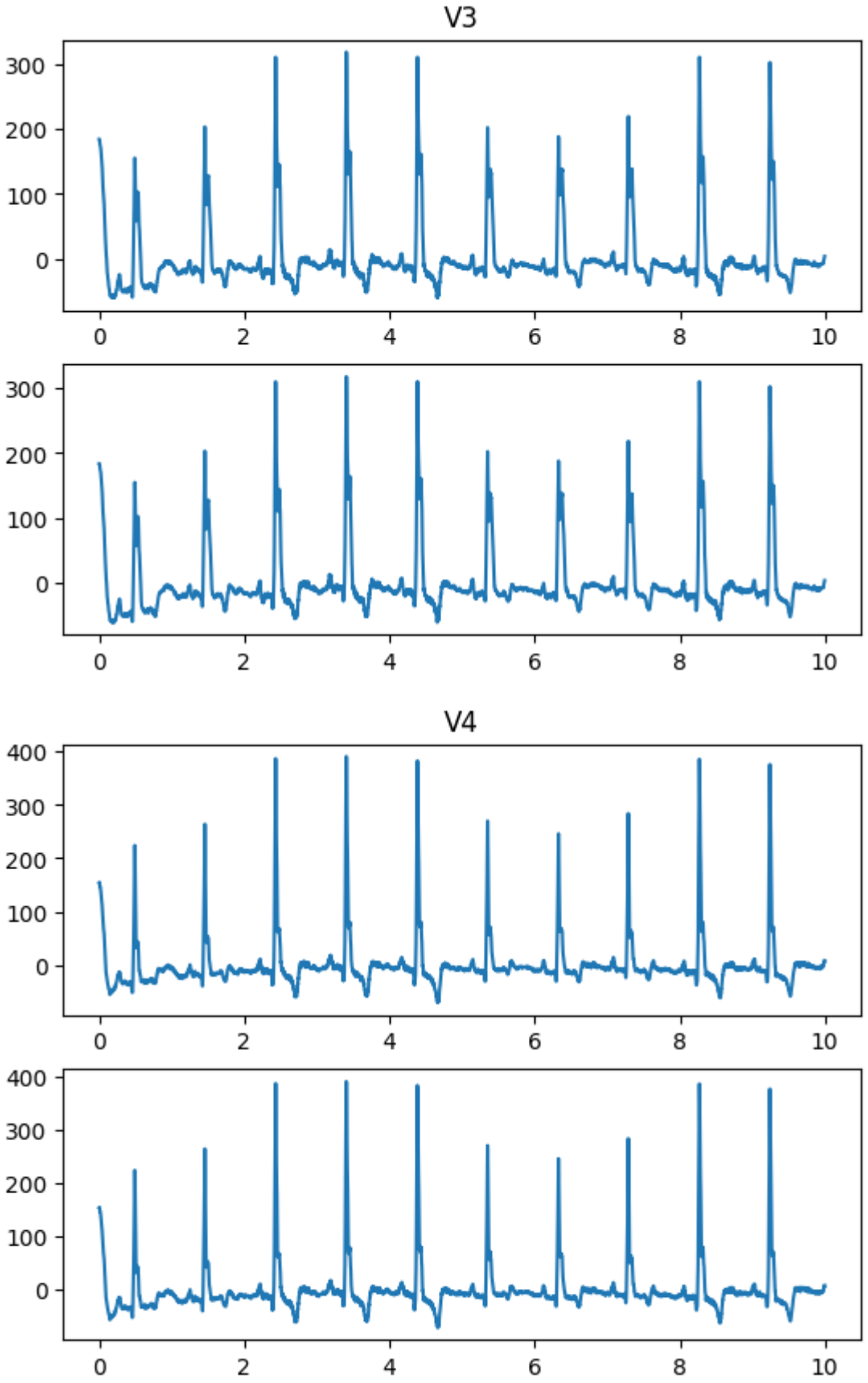
In [ ]:
```
for derivation in header[1::]:
    derivationNocthPlot(dataset, derivation, 60)
```

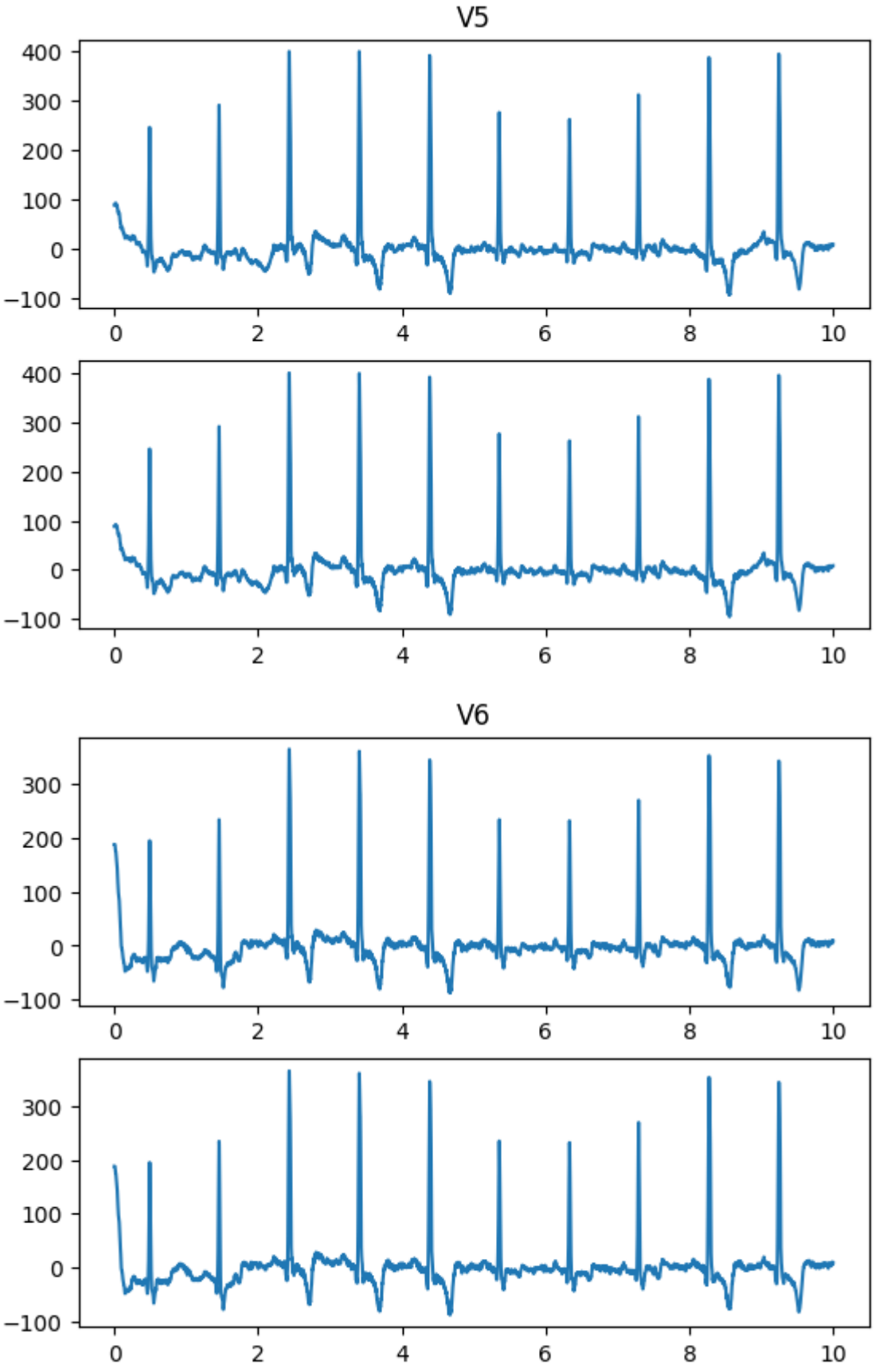

lead I
200
0
−200
0
2
4
6
8
10
200
0
−200
0
2
4
6
8
10
lead II
200
0
−200
−400
−600
0
2
4
6
8
10
200
0
−200
−400
−600
0
2
4
6
8
10


lead II
400
200
0
−200
0
2
4
6
8
10
aVR


aVL
400
200
0
−200
−400
0
2
4
6
8
10
400
200
0
−200
−400
0
2
4
6
8
10
aVF
200
100
0
−100
−200
−300
0
2
4
6
8
10
200
100
0
−100
−200
−300
0
2
4
6
8
10

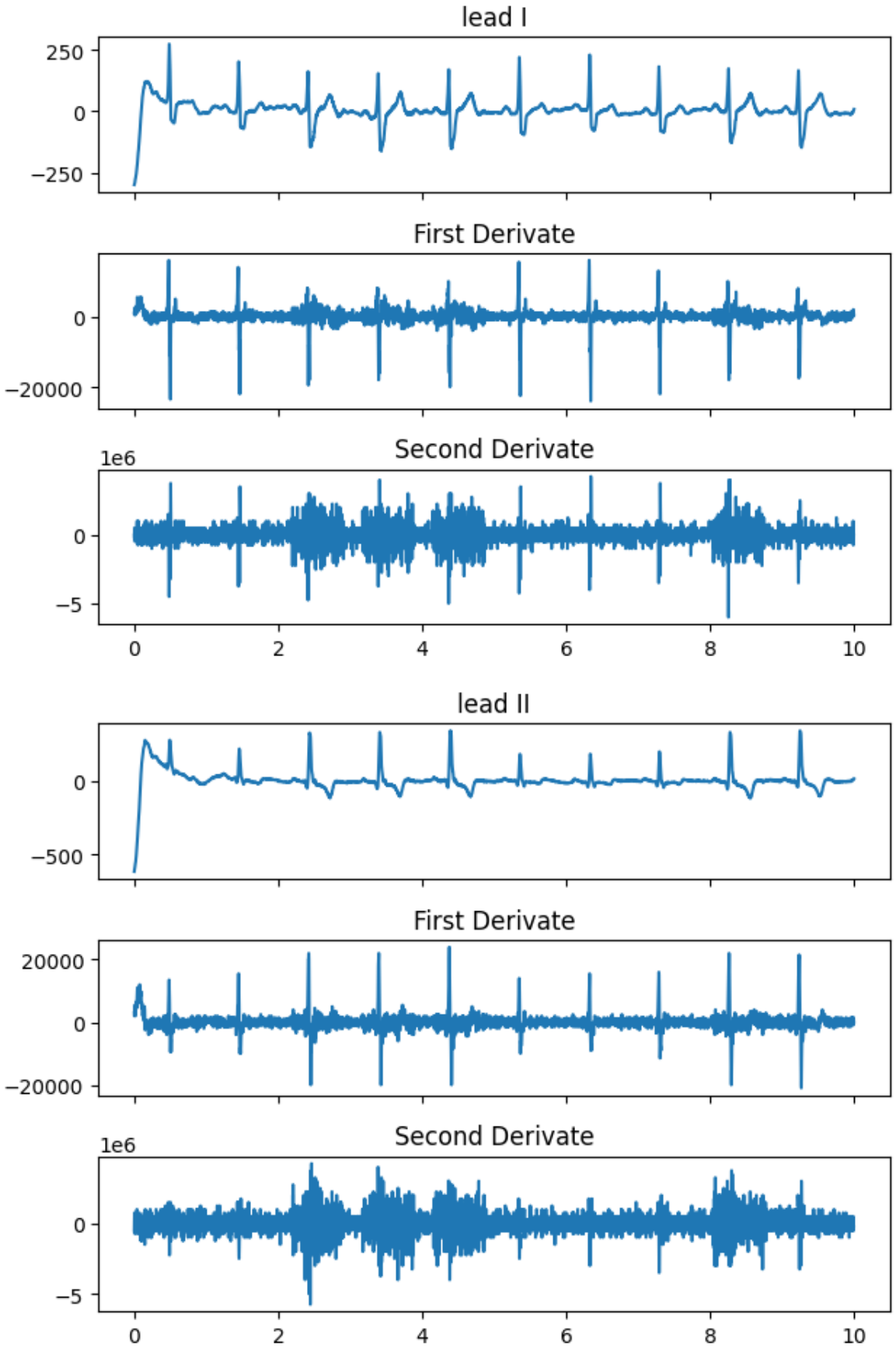

V1
150
100
50
0
−50
0
2
4
6
8
10
V2
100
0
−100

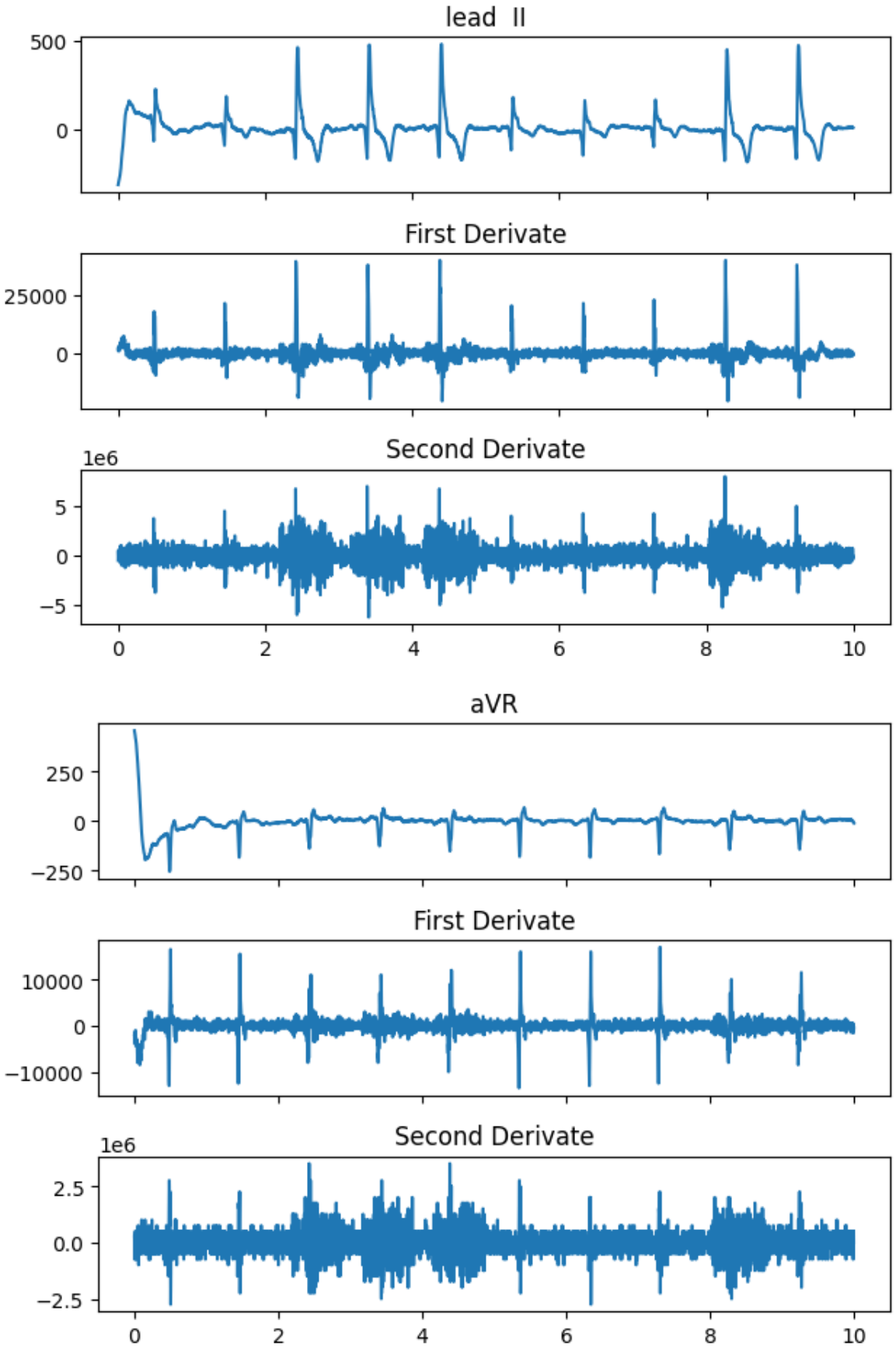

V3
300
200
100
0
0
2
4
6
8
10
300
200
100
0
0
2
4
6
8
10
V4
400
300
200
100
0
0
2
4
6
8
10
400
300
200
100
0
0
2
4
6
8
10

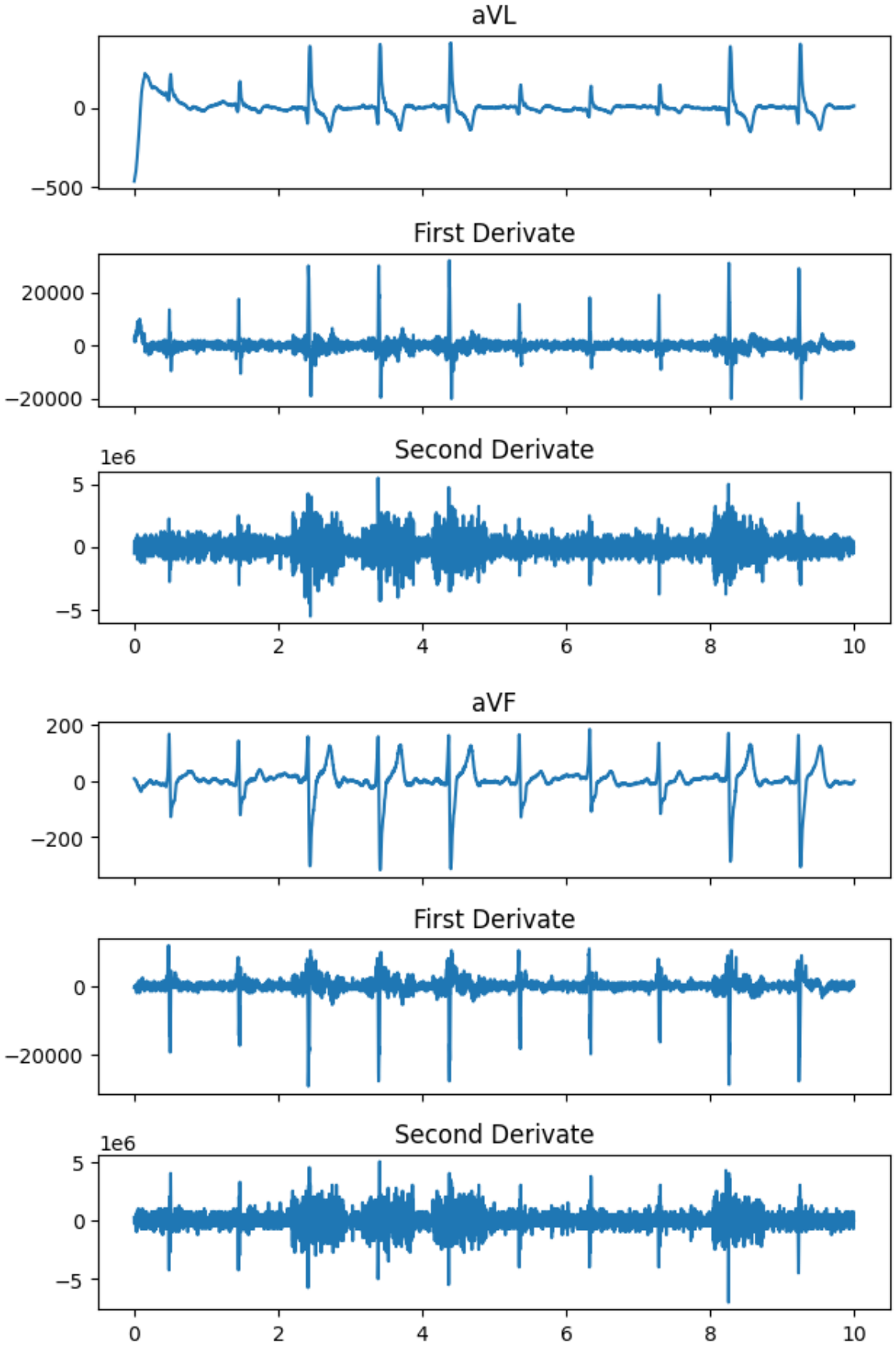


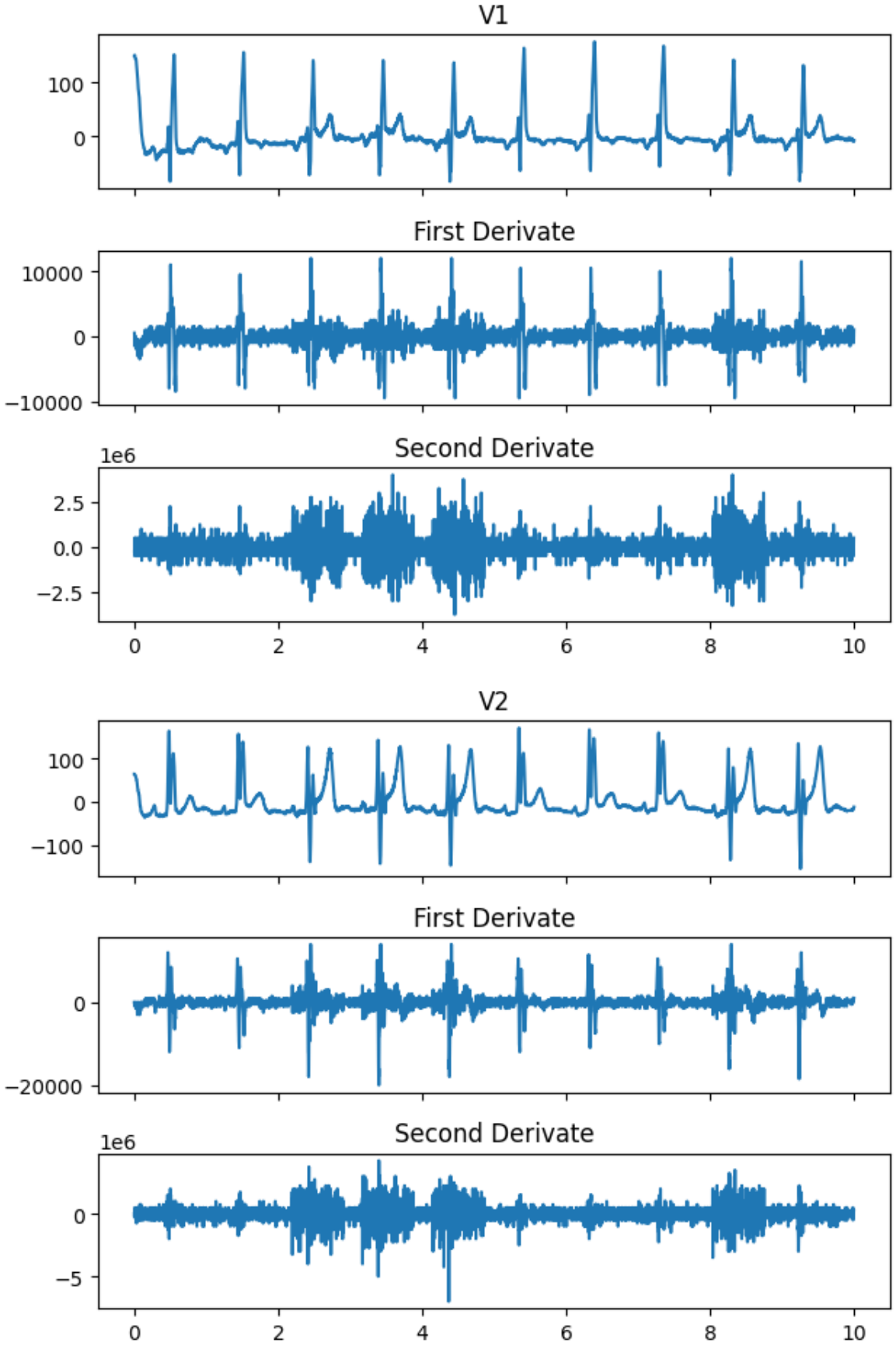
Derivadas das derivações

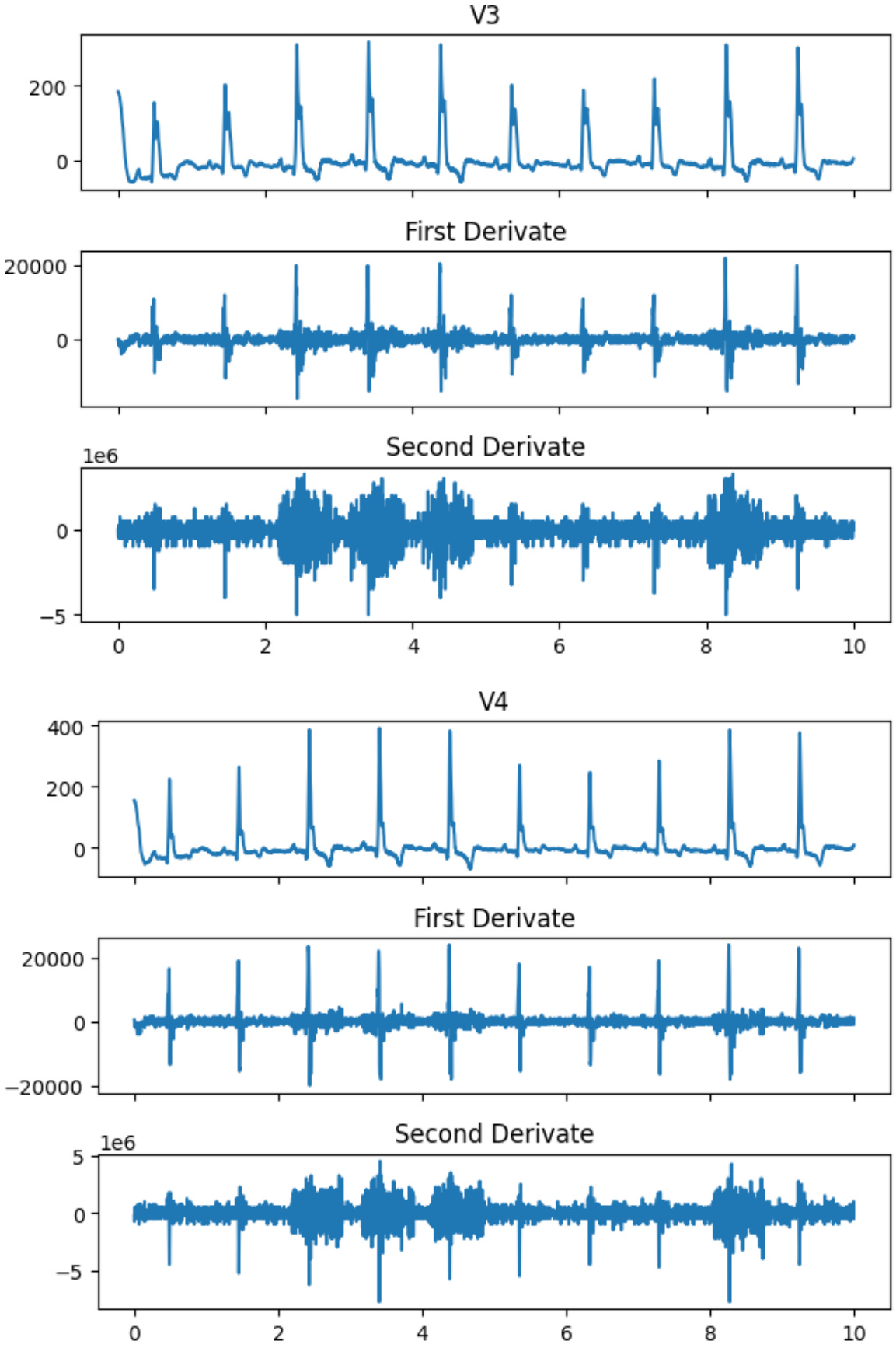
In [ ]:
```
for derivation in header[1::]:
    derivationDerivatesPlot(dataset, derivation)
```

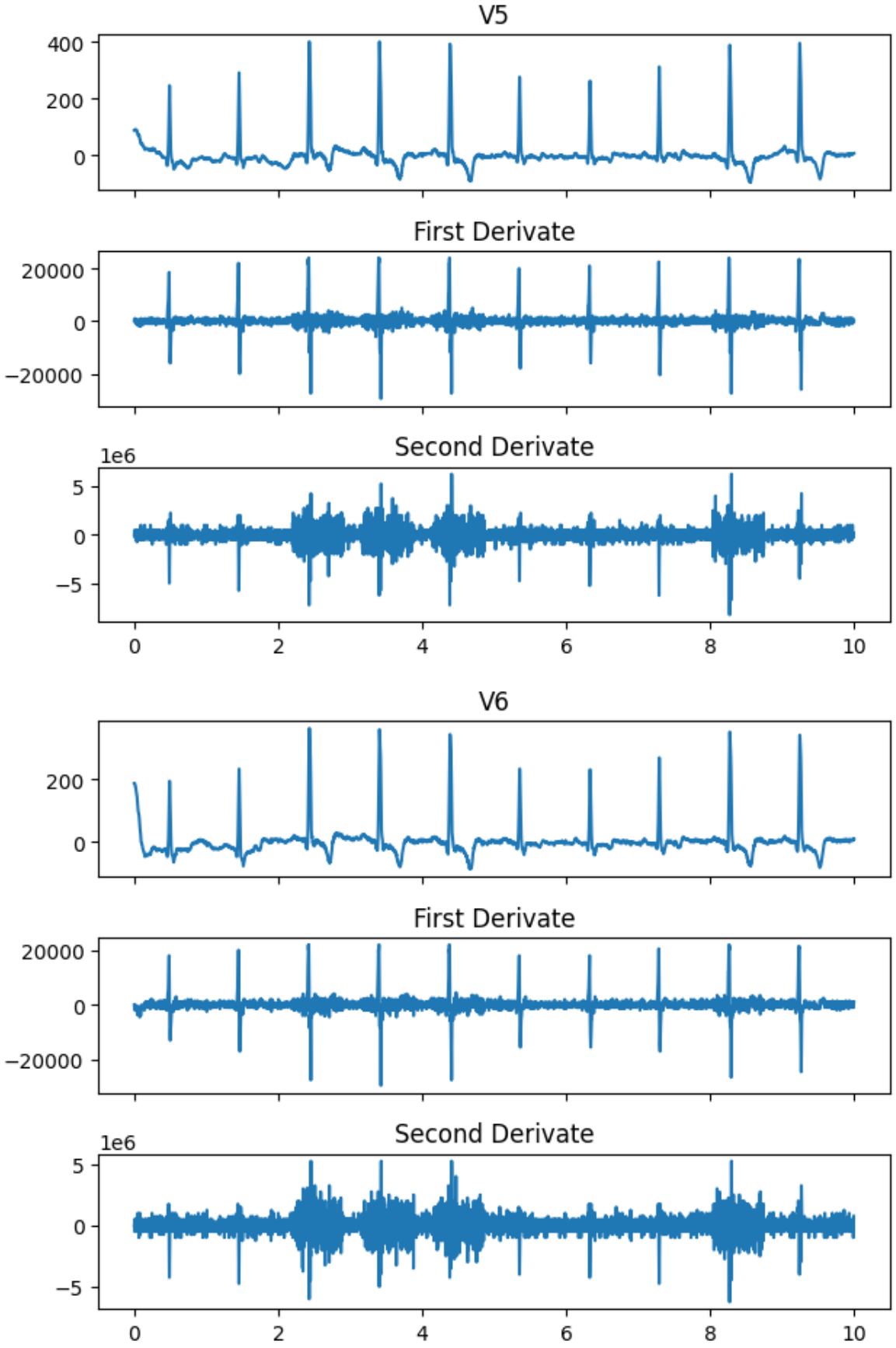

lead I
250
0
−250
First Derivate
0
−20000
Second Derivate
1e6
0
−5
0
2
4
6
8
10
lead II
0
−500
First Derivate
20000
0
−20000
Second Derivate
1e6
0
−5
0
2
4
6
8
10


lead II
500
0
First Derivate
25000
0
Second Derivate
1e6
5
0
−5
0
2
4
6
8
10
aVR
250
0
−250
First Derivate
10000
0
−10000
Second Derivate
1e6
2.5
0.0
−2.5
0
2
4
6
8
10


aVL
0
−500
First Derivate
20000
0
−20000
Second Derivate
1e6
5
0
−5
0
2
4
6
8
10
aVF
200
0
−200
First Derivate
0
−20000
Second Derivate
1e6
5
0
−5
0
2
4
6
8
10

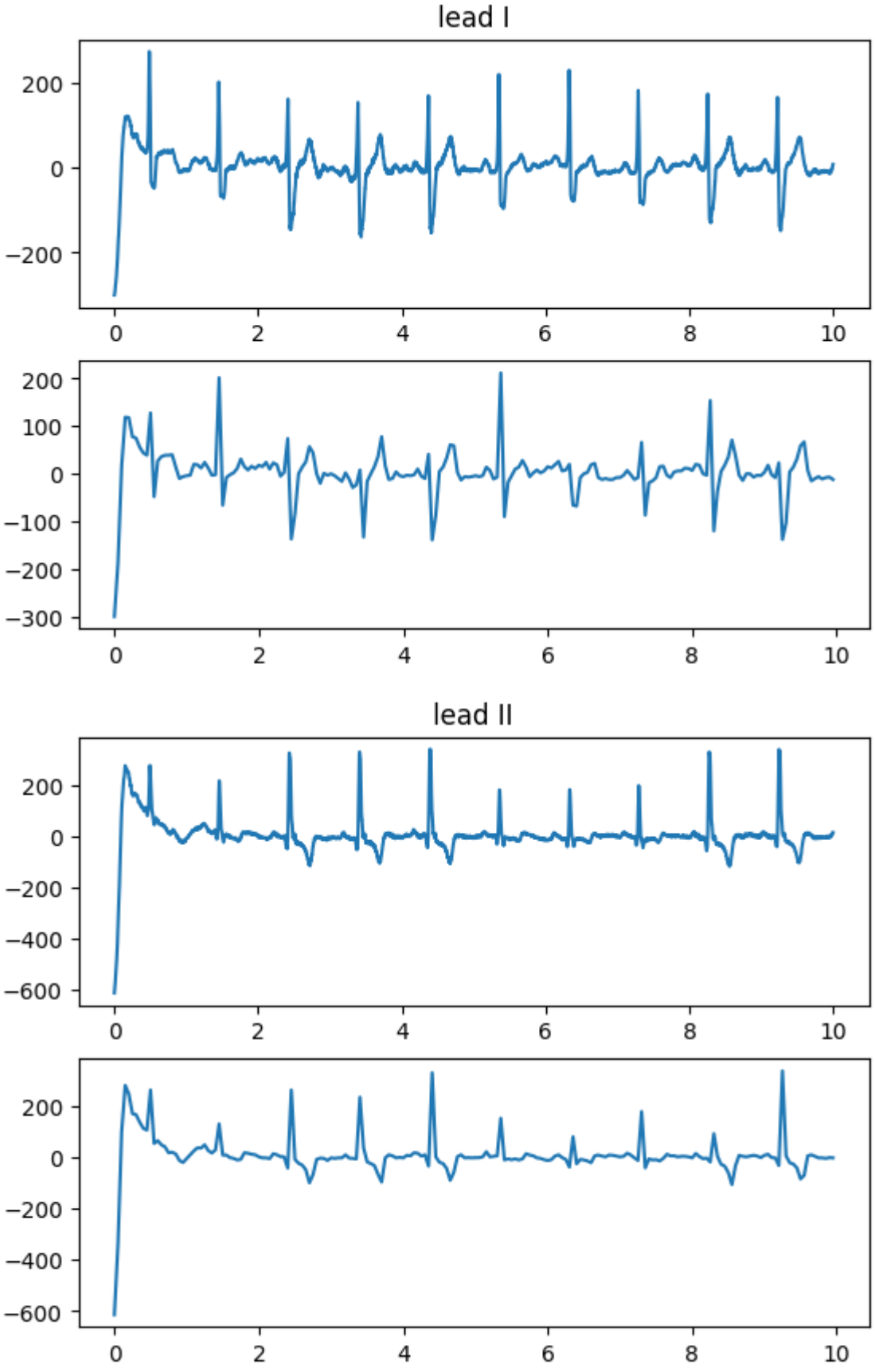

V1
100
0
First Derivate
10000
0
−10000
Second Derivate
1e6
2.5
0.0
−2.5
0
2
4
6
8
10
V2
100
0
−100
First Derivate
0
−20000
Second Derivate
1e6
0
−5
0
2
4
6
8
10

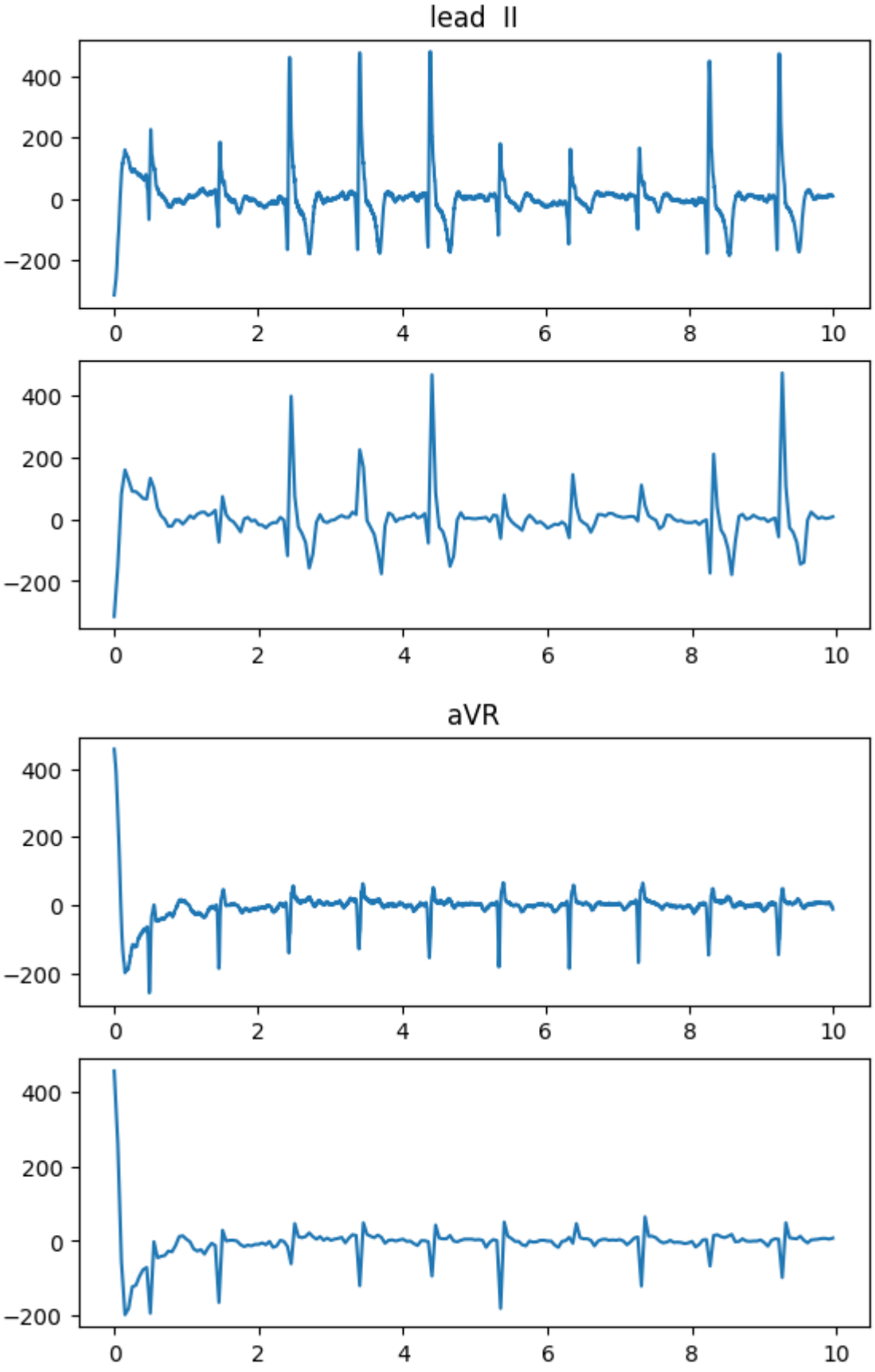

V3
200
0
First Derivate
20000
0
Second Derivate
1e6
0
−5
0
2
4
6
8
10
V4
400
200
0
First Derivate
20000
0
−20000
Second Derivate
1e6
5
0
−5
0
2
4
6
8
10

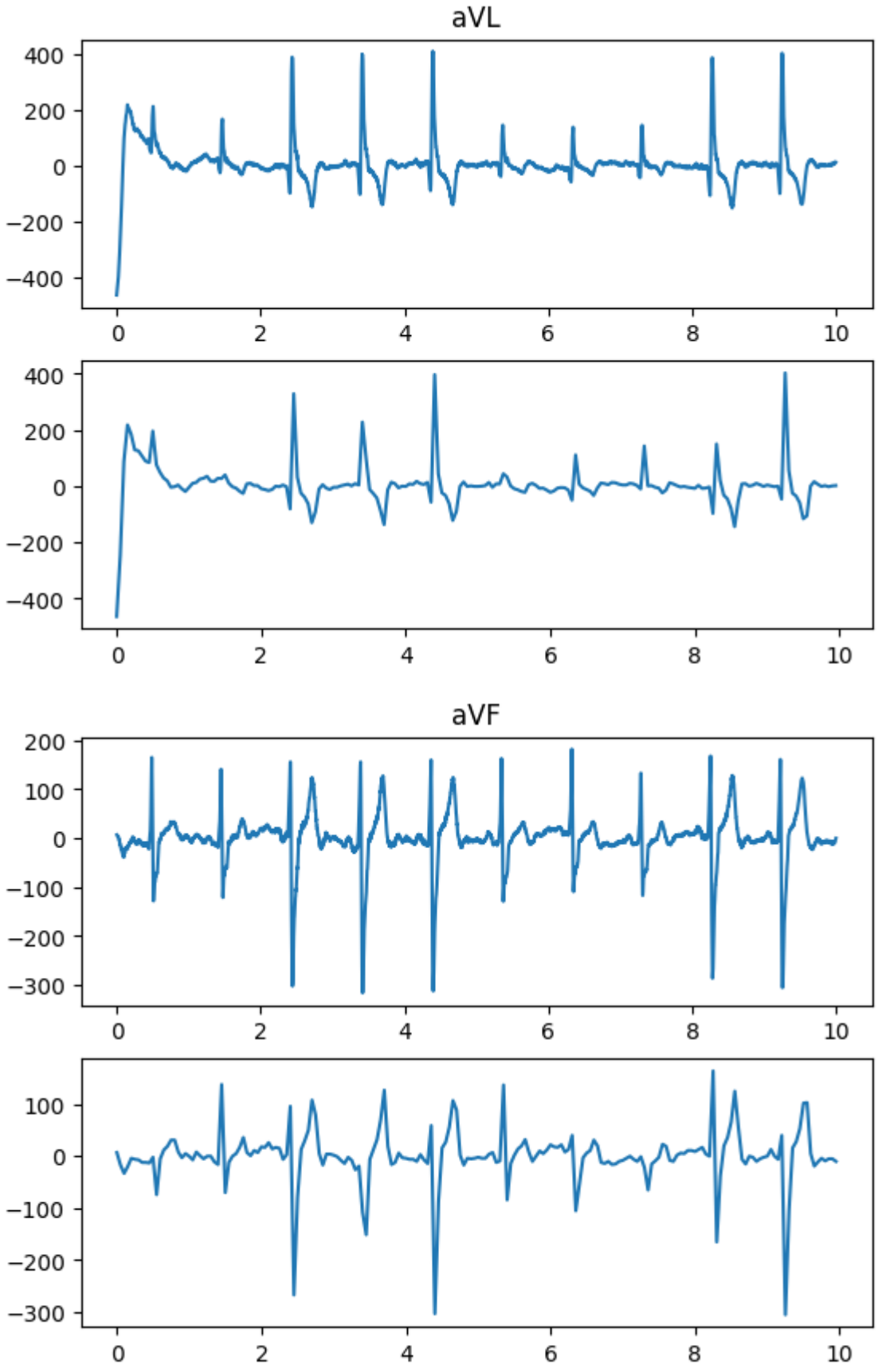


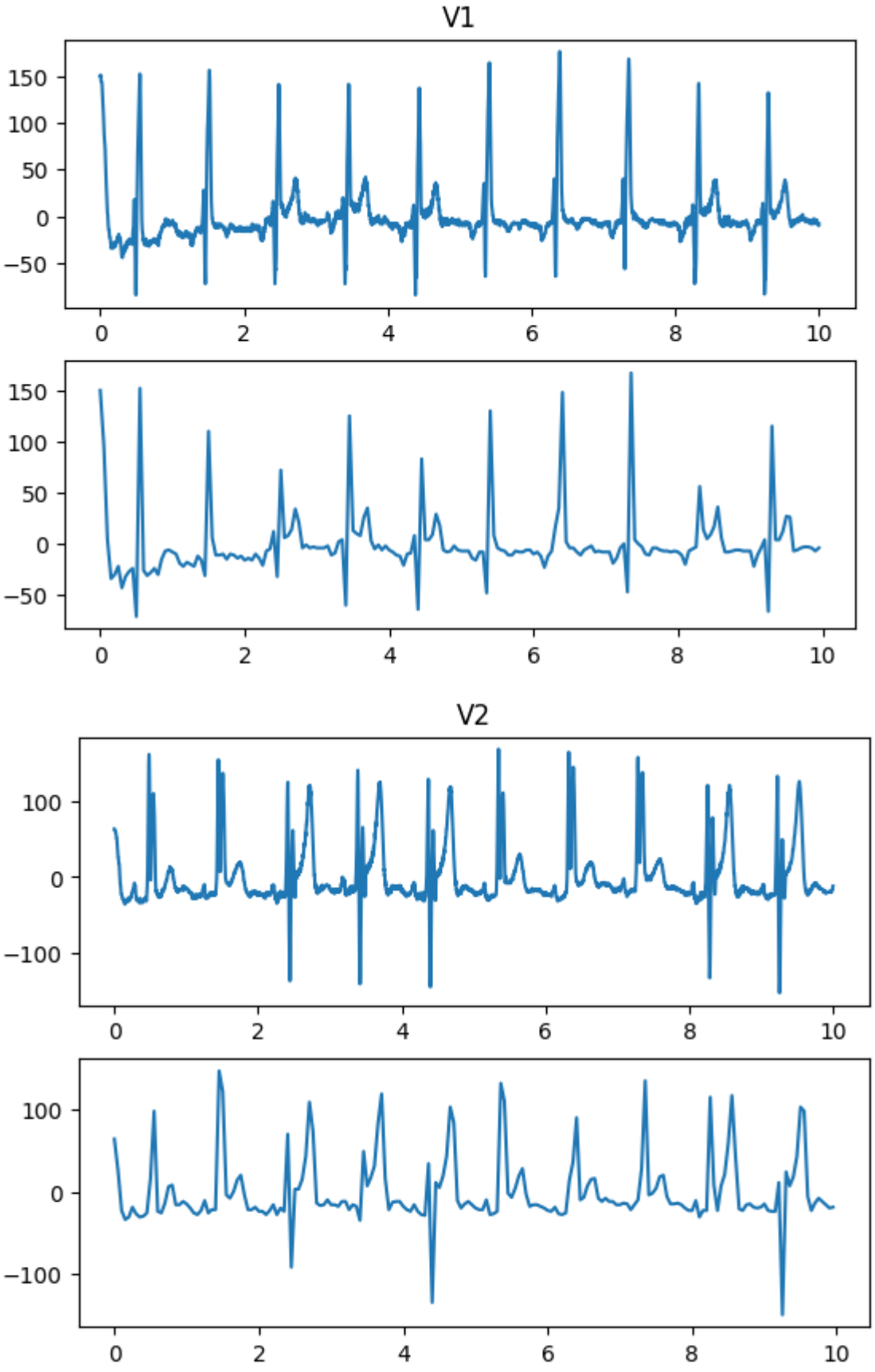
SubAmostragem das derivações com fator 25

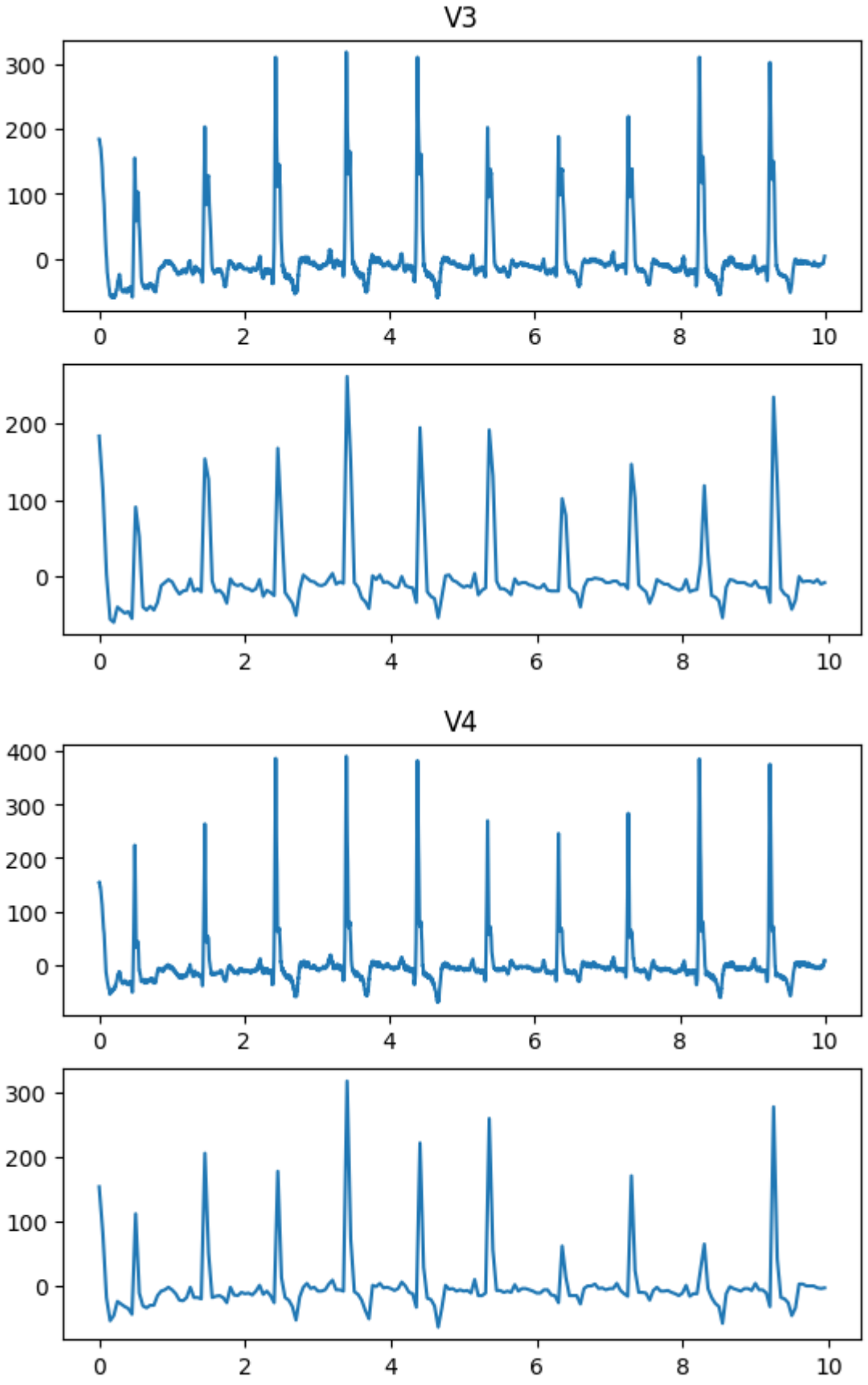
In [ ]:
```
for derivation in header[1::]:
    subSamplingPlot(dataset, derivation, 25)
```

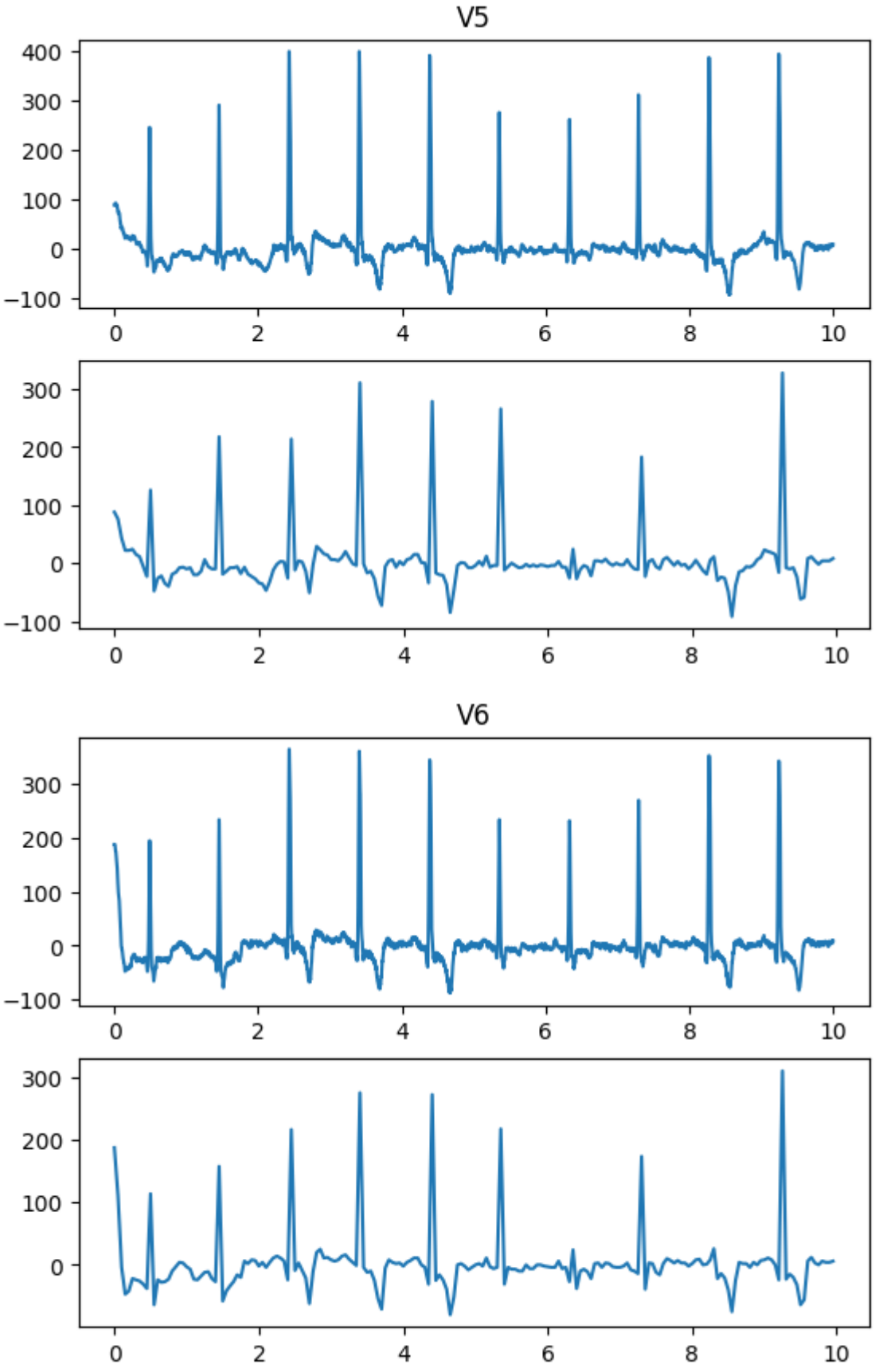

lead I
lead II


lead II
400
200
0
−200
0
2
4
6
8
10
aVR


aVL
400
200
0
−200
−400
0
2
4
6
8
10
400
200
0
−200
−400
0
2
4
6
8
10
aVF
200
100
0
−100
−200
−300
0
2
4
6
8
10
100
0
−100
−200
−300
0
2
4
6
8
10


V1
150
100
50
0
−50
0
2
4
6
8
10
V2
100
0
−100


V3
300
200
100
0
0
2
4
6
8
10
200
100
0
0
2
4
6
8
10
V4
400
300
200
100
0
0
2
4
6
8
10
300
200
100
0
0
2
4
6
8
10



Variâncias das derivações

In [ ]:
```
dataset[header[1::]].var()
```

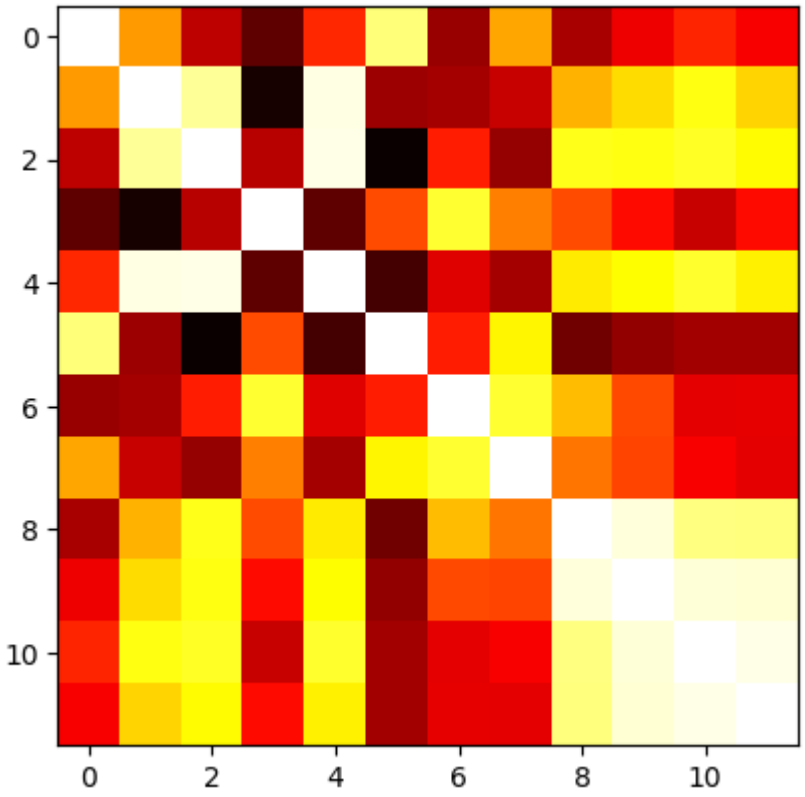
Out[ ]:
```
lead I      2236.146980
lead II     5285.052101
lead  II    5978.085145
aVR         2257.175647
aVL         5077.952501
aVF         2792.517731
V1          1043.175813
V2          1674.741153
V3          2591.492531
V4          2471.526765
V5          2624.402227
V6          2360.755338
dtype: float64
```

Matriz de Correlação

In [ ]:
```
correlationMatrix = dataset[header[1::]].corr()

print(correlationMatrix)

plt.imshow(correlationMatrix, cmap='hot', interpolation='nearest')
```

```
            lead I   lead II  lead  II       aVR       aVL       aVF  \
lead I    1.000000  0.224436 -0.400576 -0.668139 -0.104230  0.740565
lead II   0.224436  1.000000  0.802985 -0.874982  0.945781 -0.488612
lead  II -0.400576  0.802985  1.000000 -0.414067  0.953019 -0.912348
aVR      -0.668139 -0.874982 -0.414067  1.000000 -0.670314  0.005137
aVL      -0.104230  0.945781  0.953019 -0.670314  1.000000 -0.745489
aVF       0.740565 -0.488612 -0.912348  0.005137 -0.745489  1.000000
V1       -0.504490 -0.464322 -0.128032  0.604664 -0.305731 -0.131180
V2        0.260801 -0.373356 -0.510555  0.154927 -0.468000  0.491365
V3       -0.460551  0.296334  0.560303  0.002481  0.455870 -0.616567
V4       -0.260211  0.412143  0.546663 -0.185404  0.507263 -0.517055
V5       -0.108524  0.551729  0.585137 -0.367361  0.599191 -0.477507
V6       -0.226079  0.391081  0.505985 -0.186061  0.474363 -0.472057

                V1        V2        V3        V4        V5        V6
lead I   -0.504490  0.260801 -0.460551 -0.260211 -0.108524 -0.226079
lead II  -0.464322 -0.373356  0.296334  0.412143  0.551729  0.391081
lead  II -0.128032 -0.510555  0.560303  0.546663  0.585137  0.505985
aVR       0.604664  0.154927  0.002481 -0.185404 -0.367361 -0.186061
aVL      -0.305731 -0.468000  0.455870  0.507263  0.599191  0.474363
aVF      -0.131180  0.491365 -0.616567 -0.517055 -0.477507 -0.472057
V1        1.000000  0.605843  0.323947 -0.008173 -0.289164 -0.279028
V2        0.605843  1.000000  0.125738 -0.016090 -0.229092 -0.285742
V3        0.323947  0.125738  1.000000  0.929425  0.753541  0.750063
V4       -0.008173 -0.016090  0.929425  1.000000  0.919541  0.916240
V5       -0.289164 -0.229092  0.753541  0.919541  1.000000  0.951237
V6       -0.279028 -0.285742  0.750063  0.916240  0.951237  1.000000
```

Out[ ]:
```
<matplotlib.image.AxesImage at 0x72db76aaba40>
```


0
2
4
6
8
10
0
2
4
6
8
10