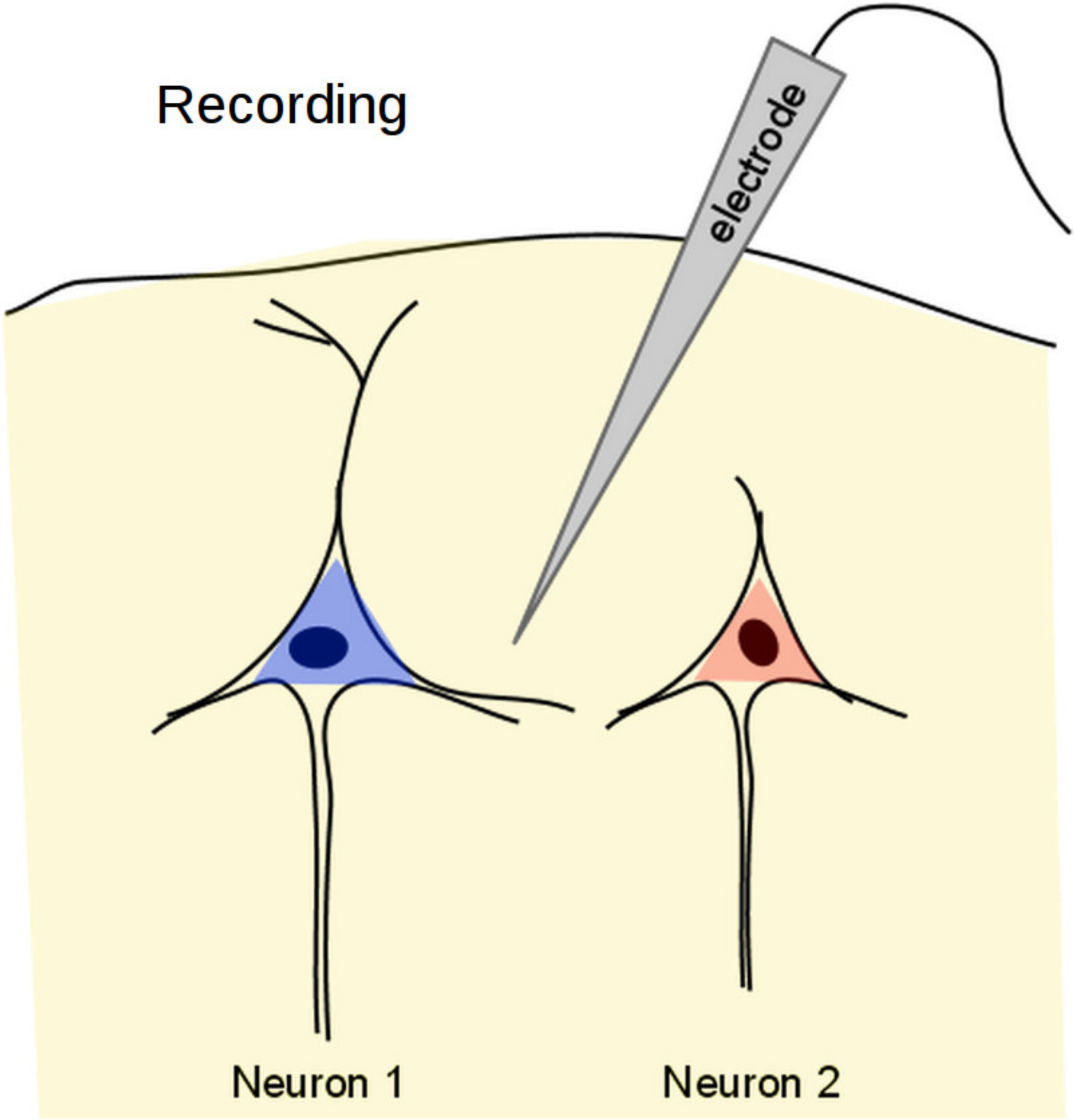
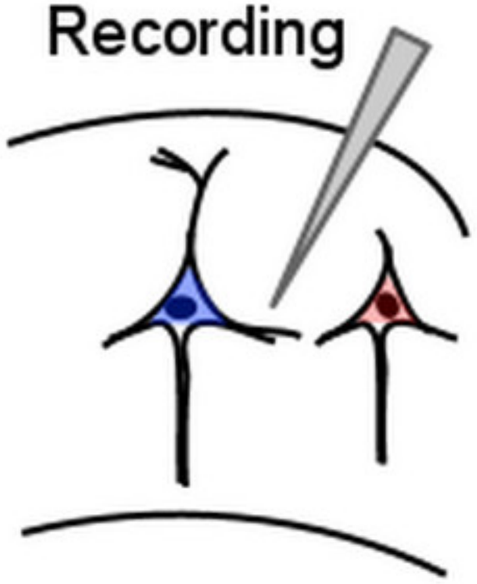
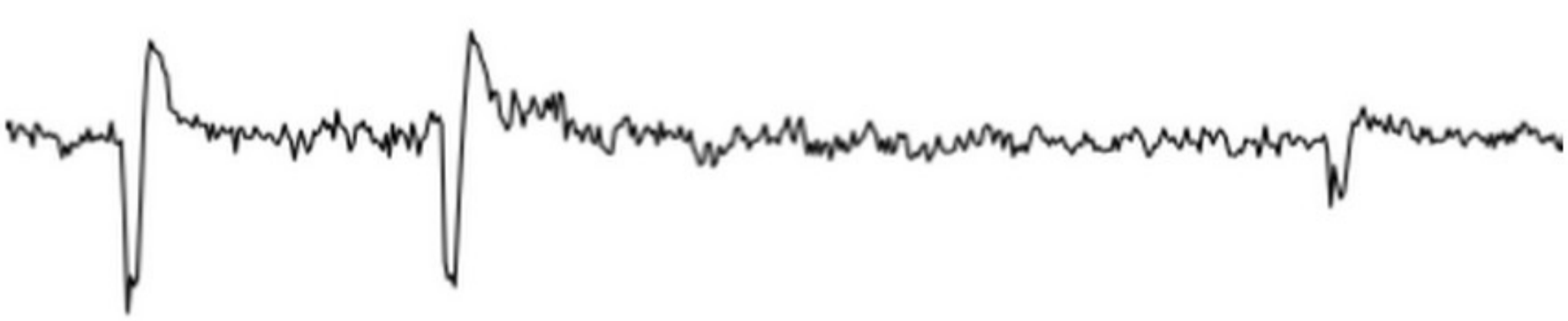
SpikeSort:
flexible spike sorting in Python

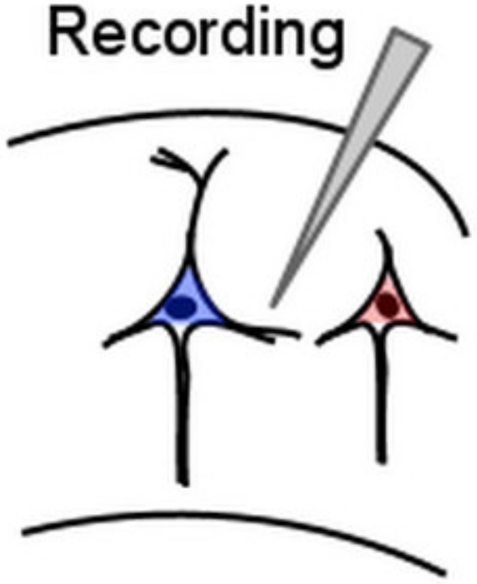
Dmytro Bielievtsov, Bartosz Telenczuk


Recording
electrode
Neuron 1
Neuron 2

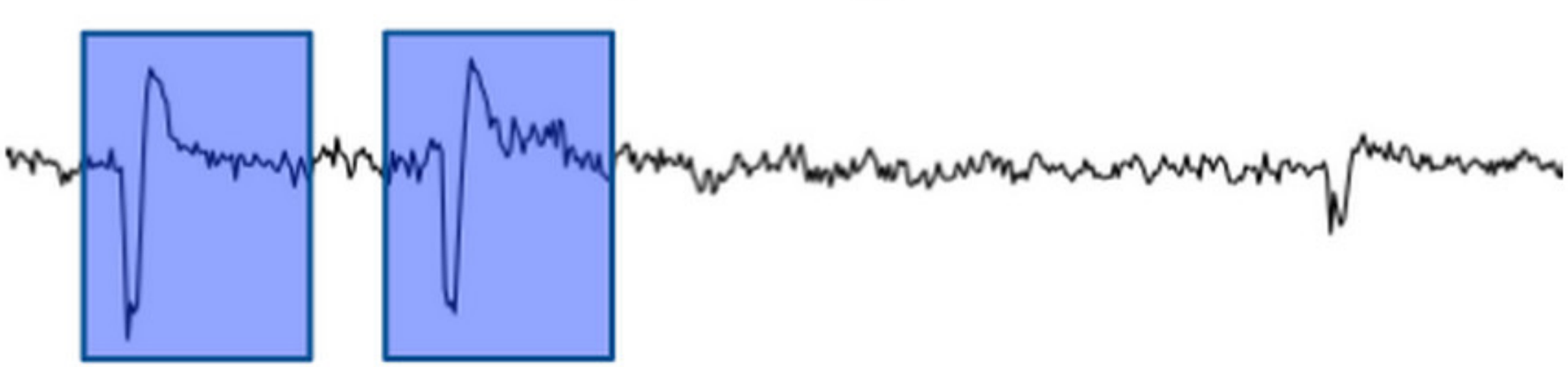


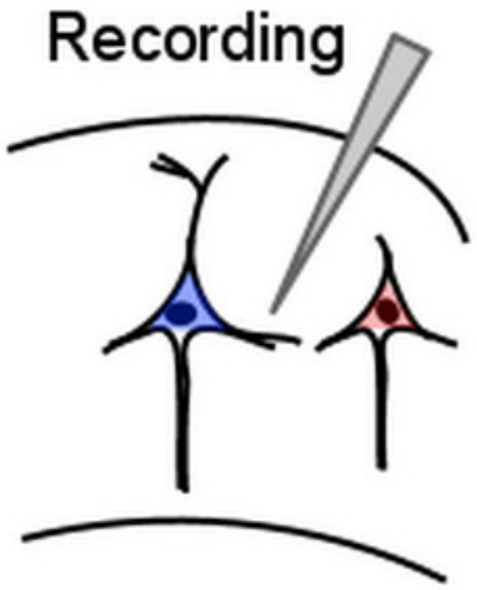
1) Filtering



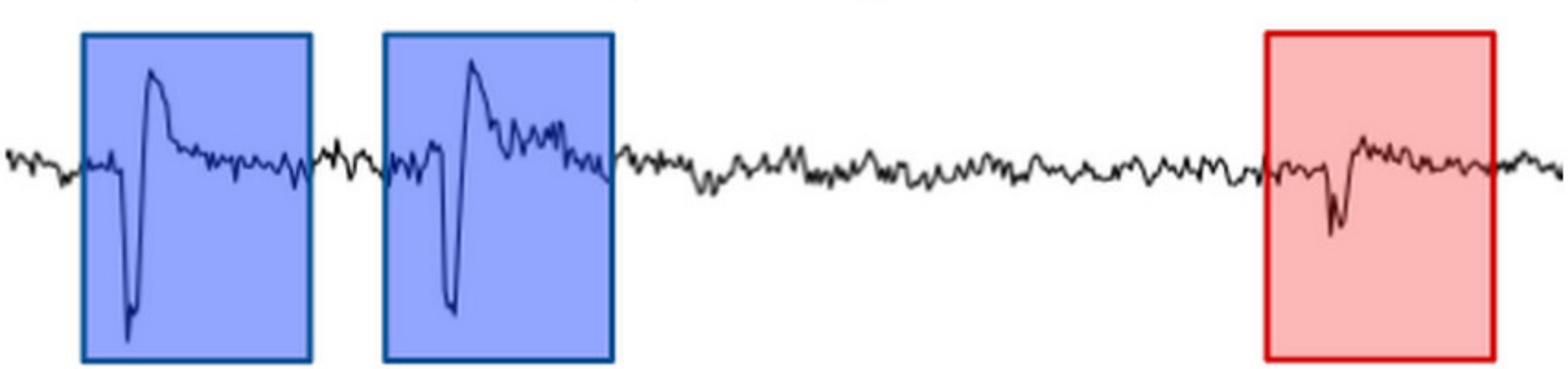


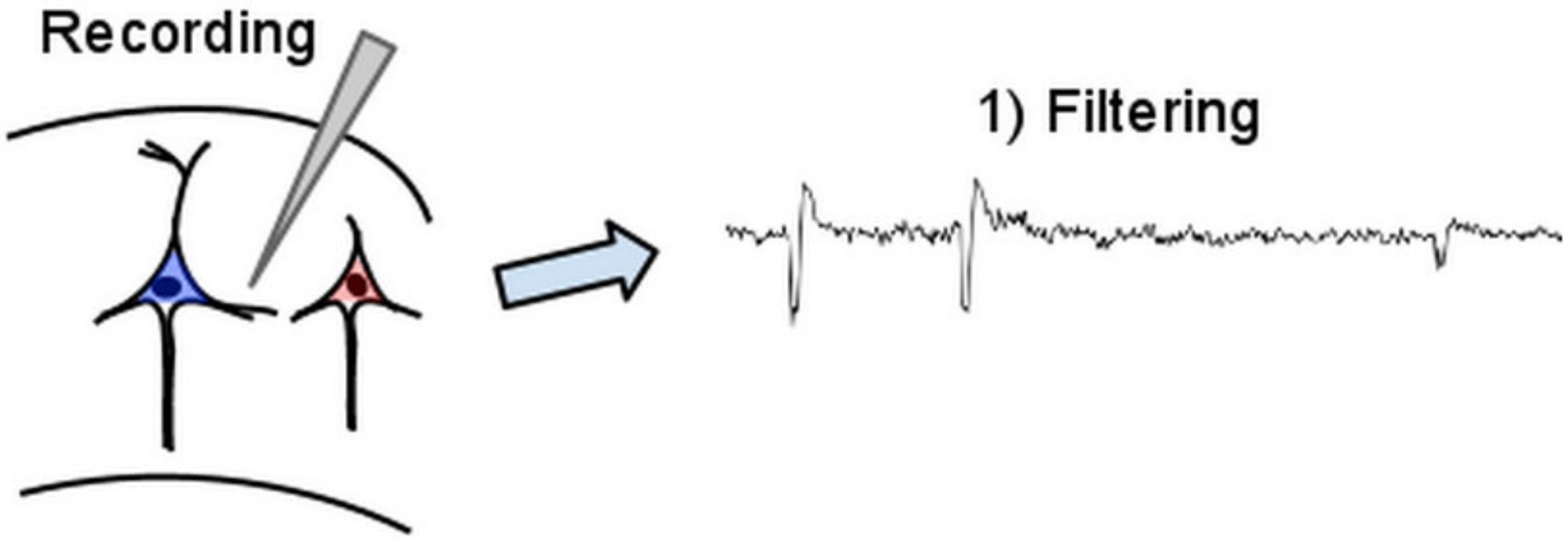
1) Filtering

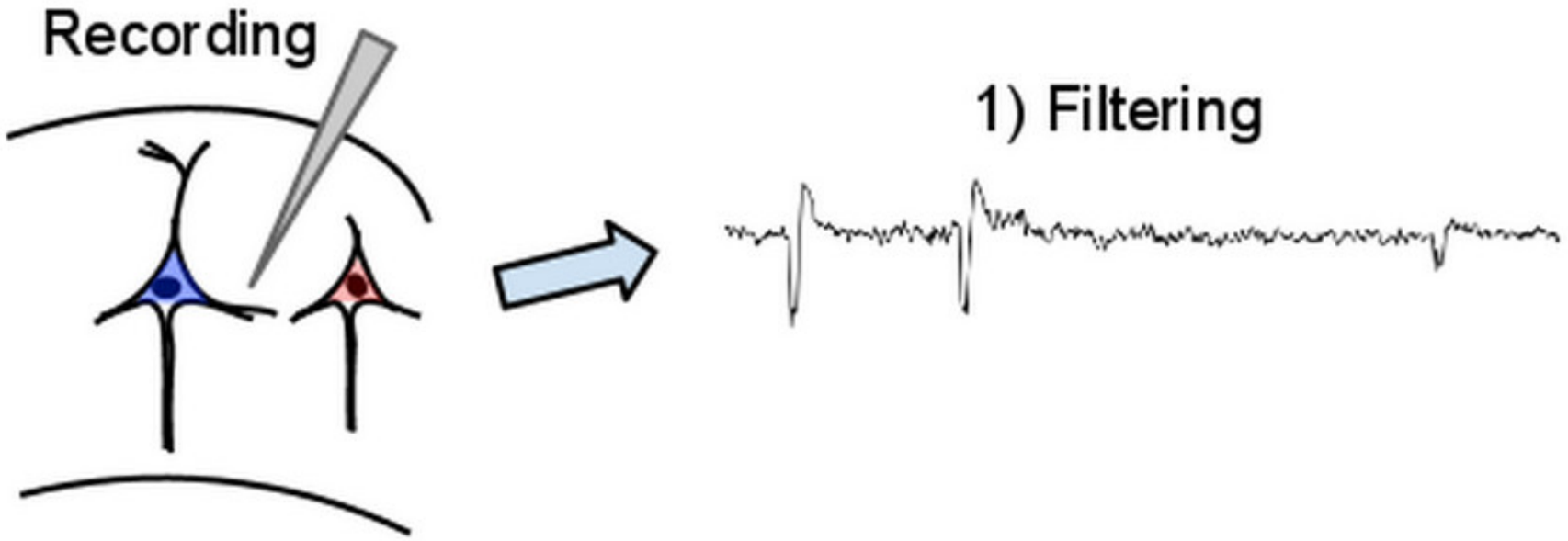




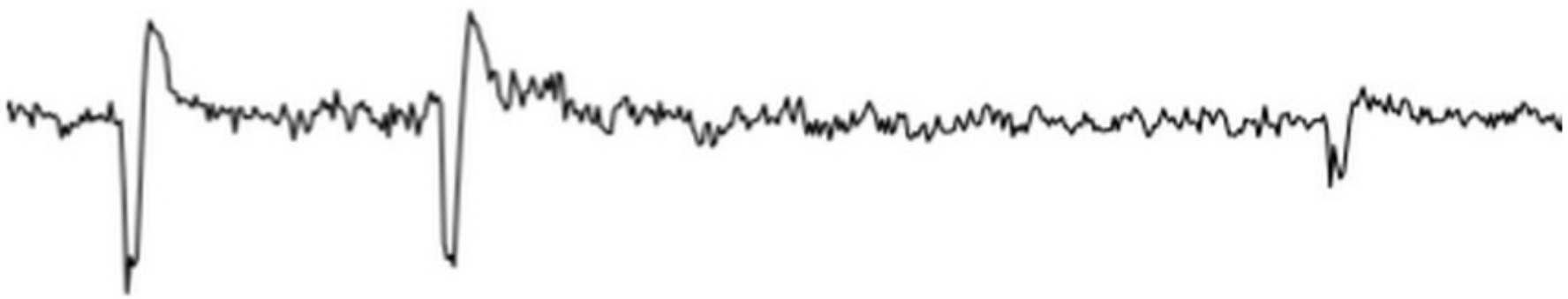
1) Filtering

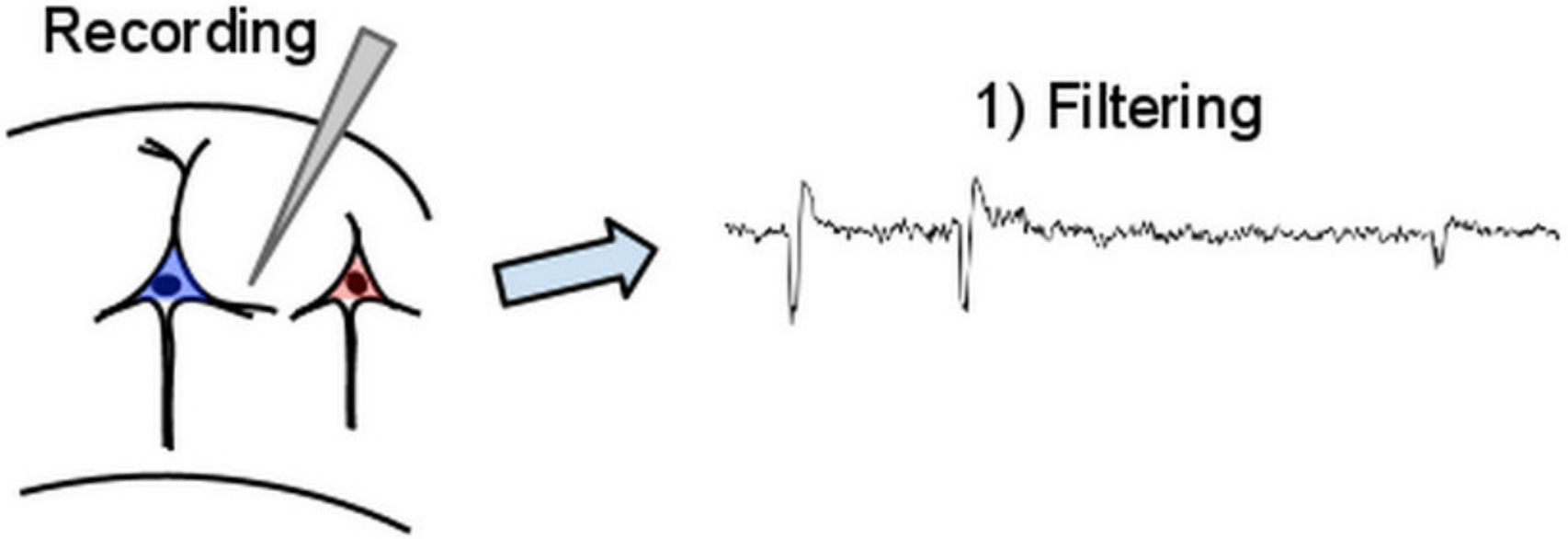



Recording
1) Filtering

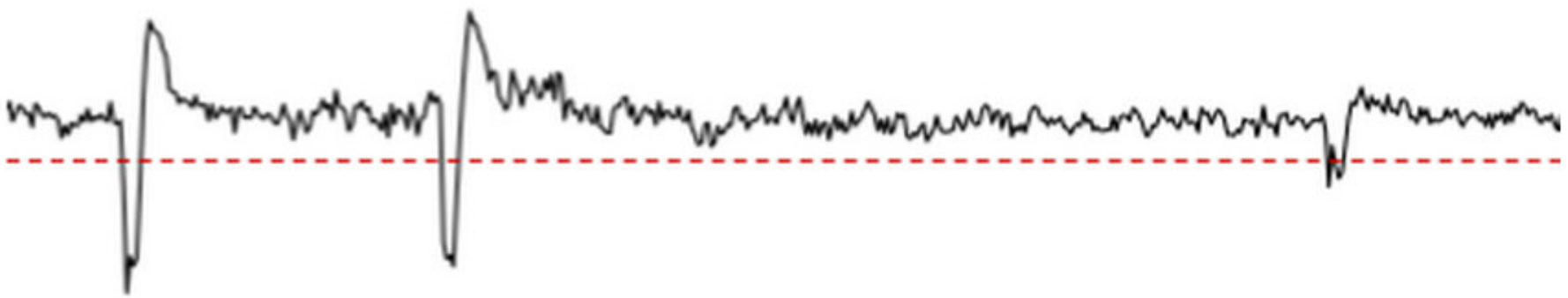


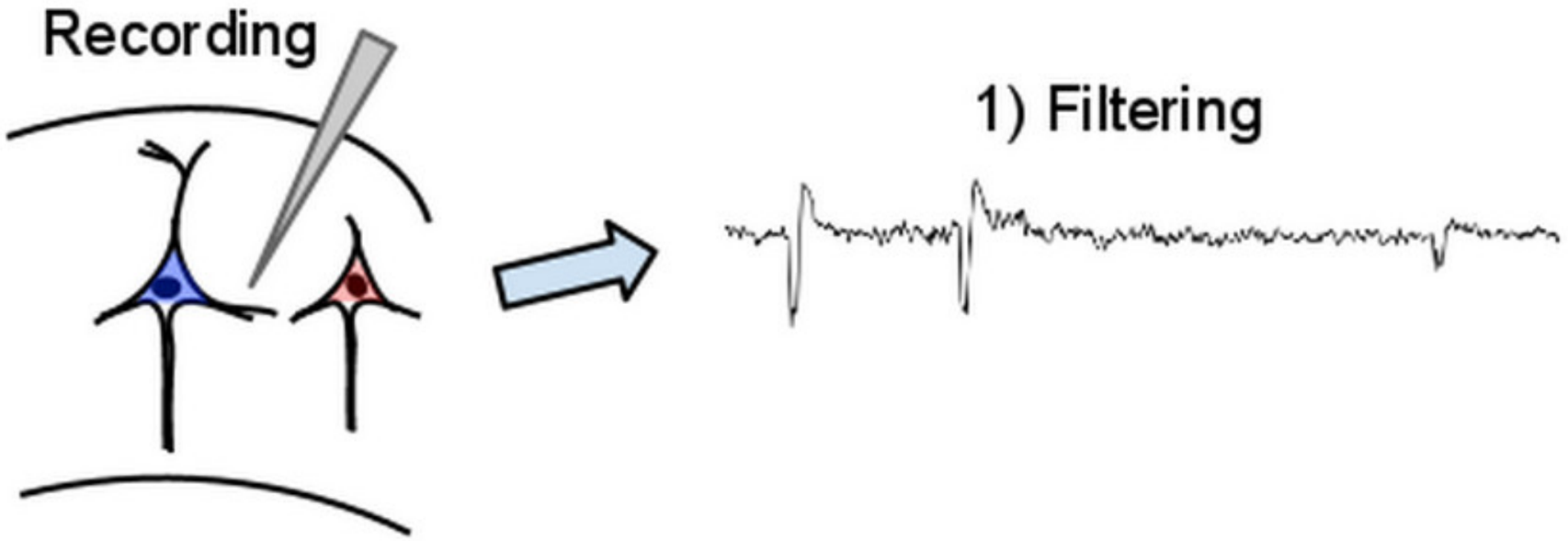
2) Detection



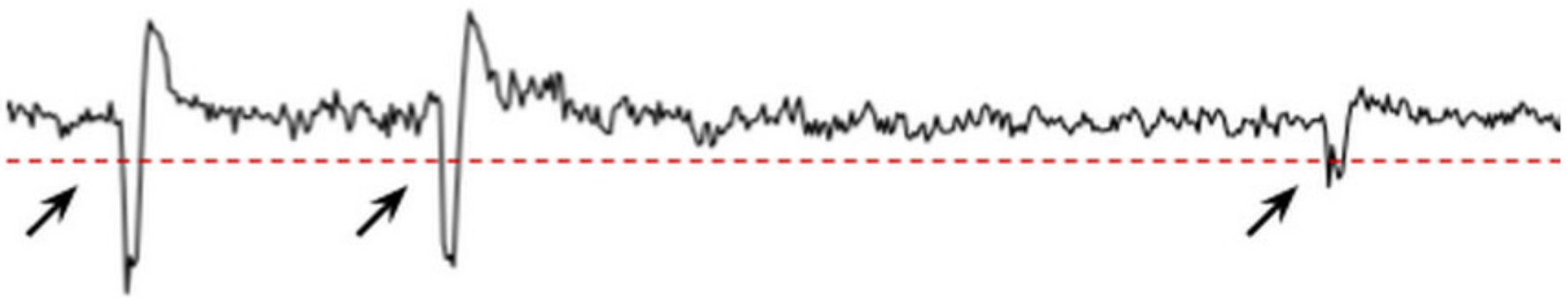


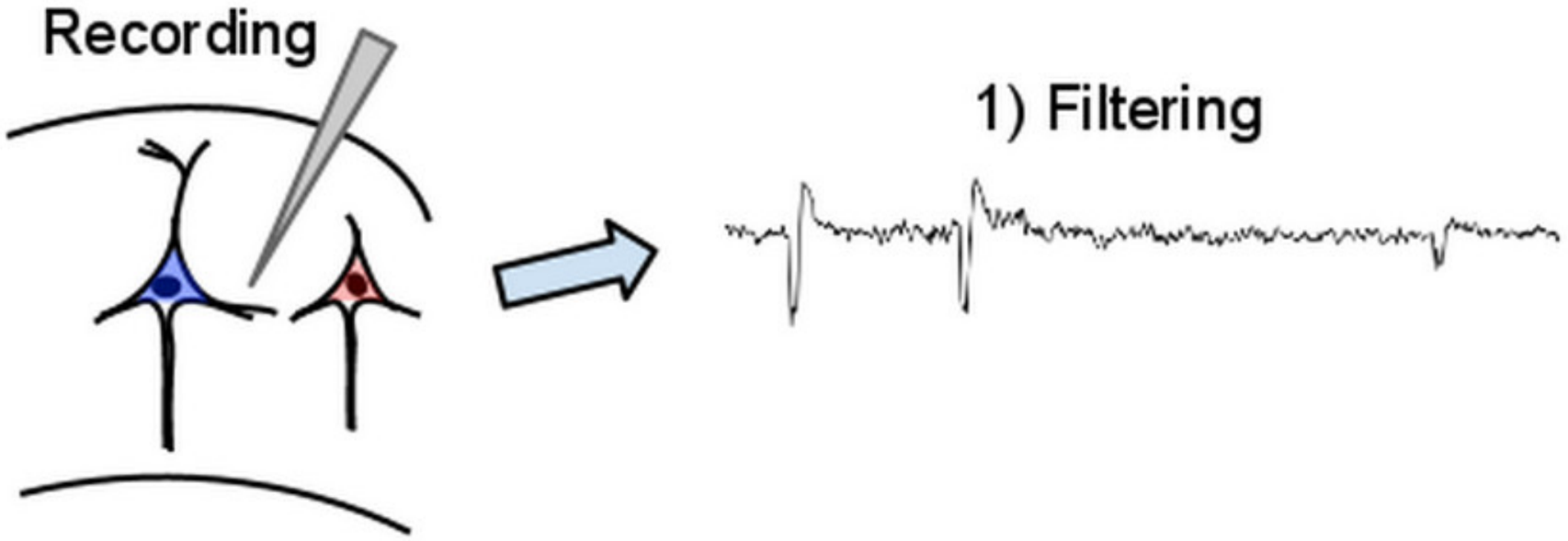
2) Detection



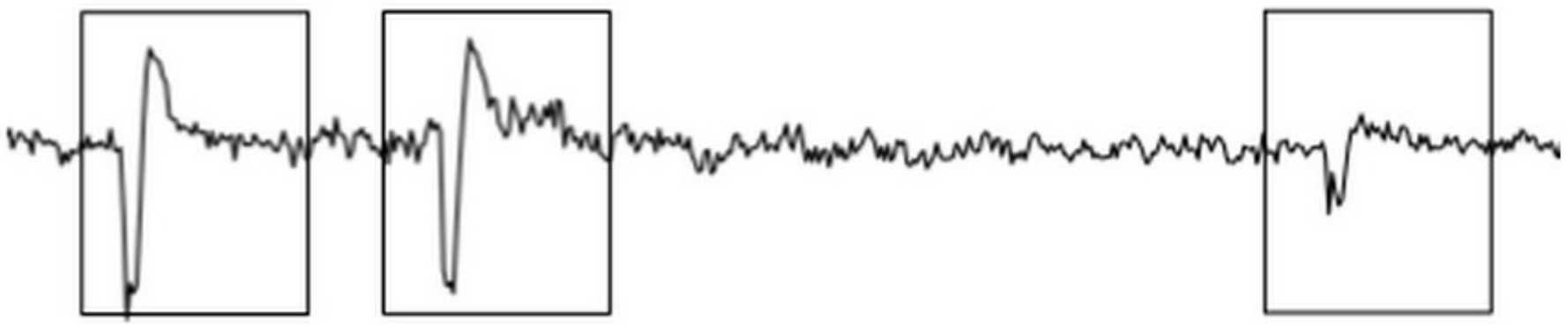


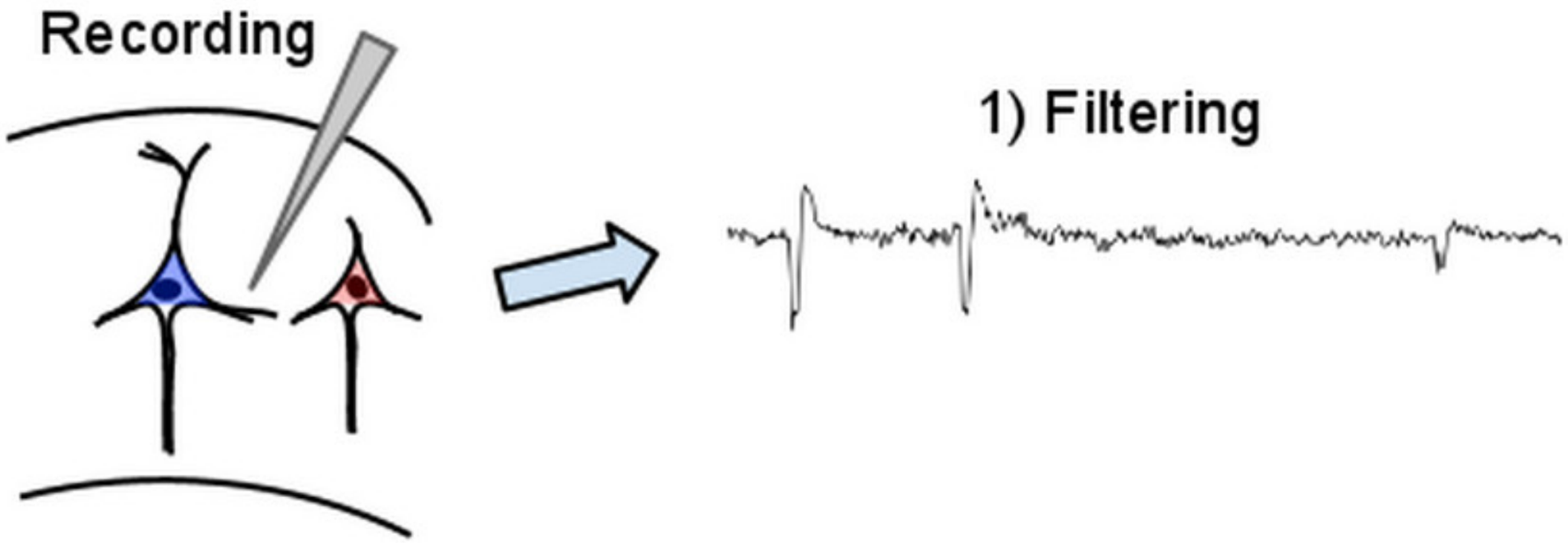
2) Detection

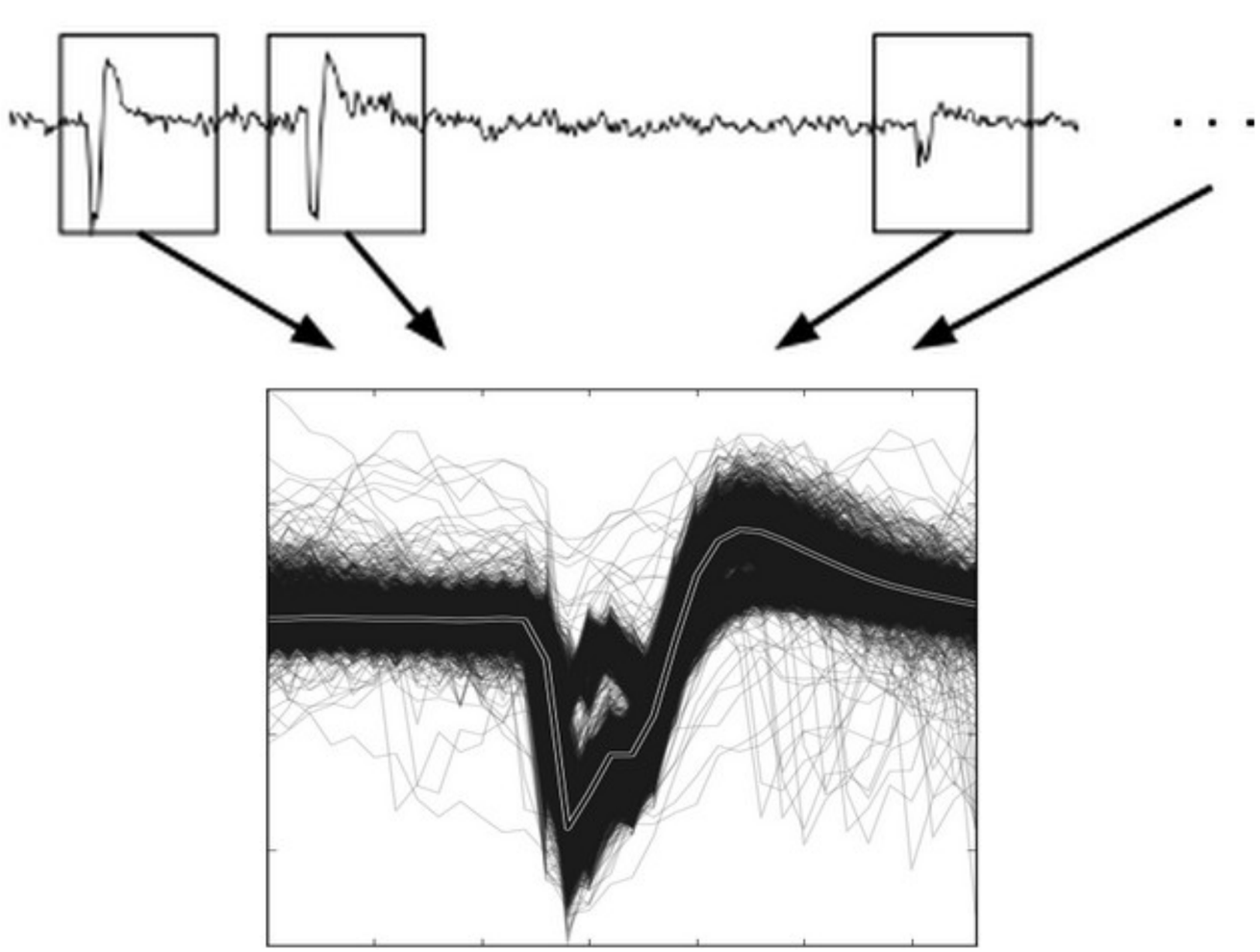


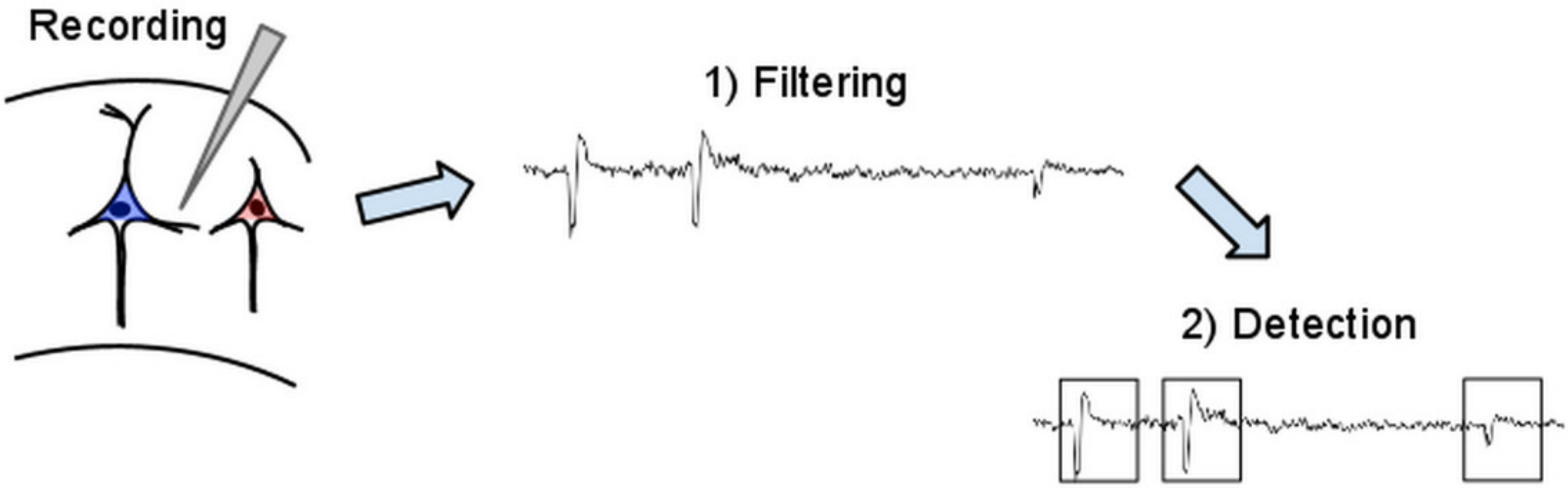


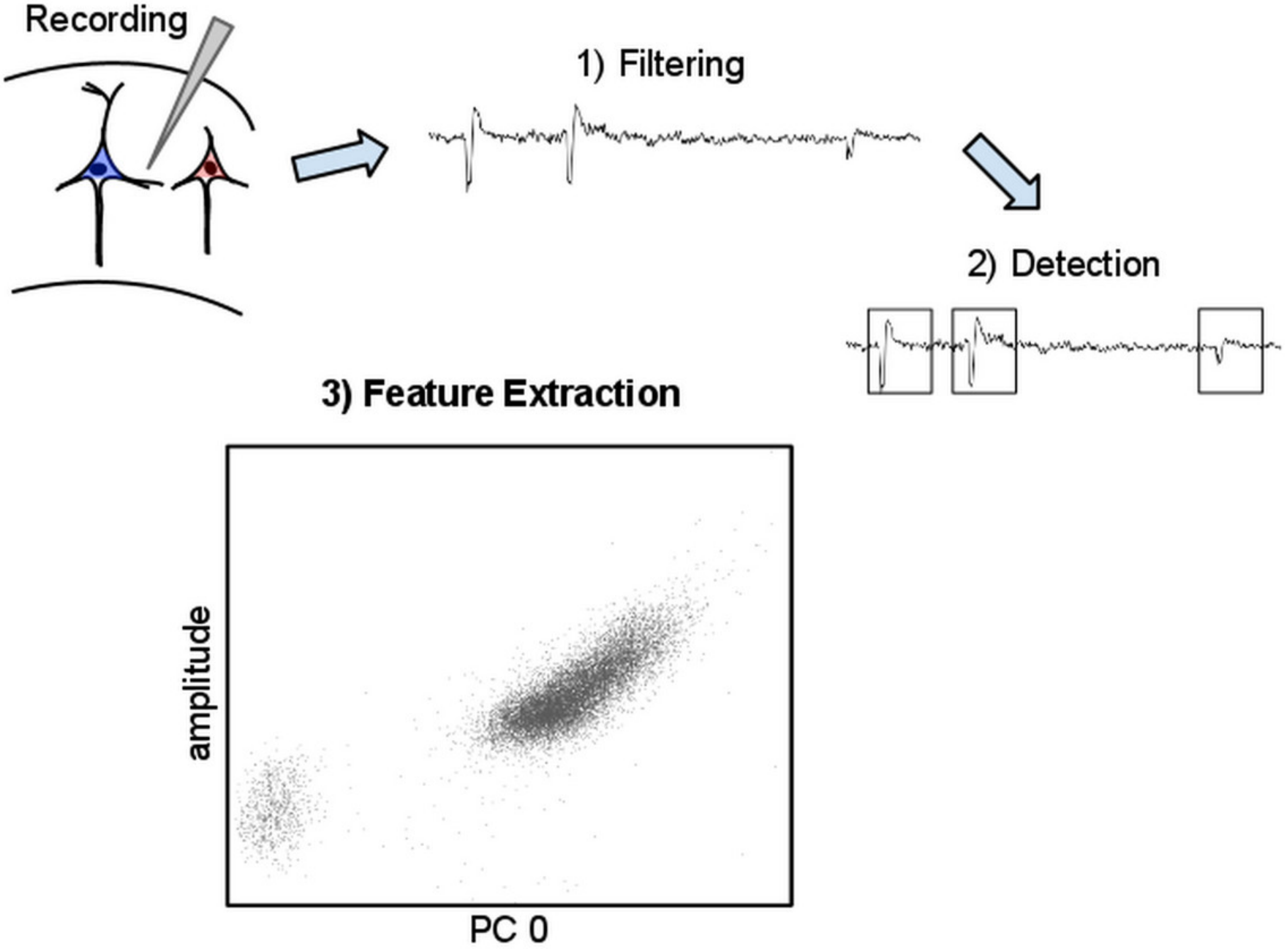
2) Detection



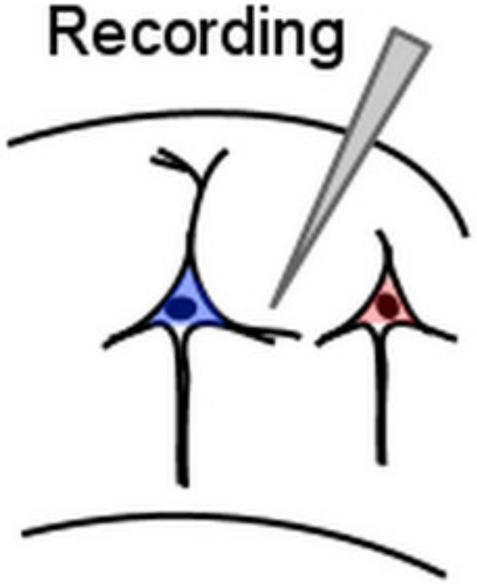

Recording
1) Filtering

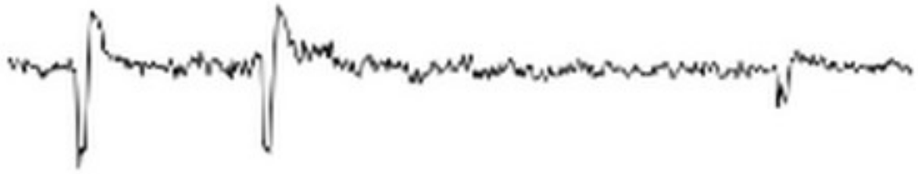

2) Detection


Recording
1) Filtering
2) Detection


Recording
1) Filtering
2) Detection
3) Feature Extraction
amplitude
PC 0

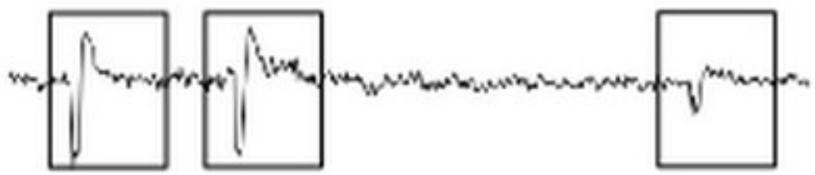
Recording



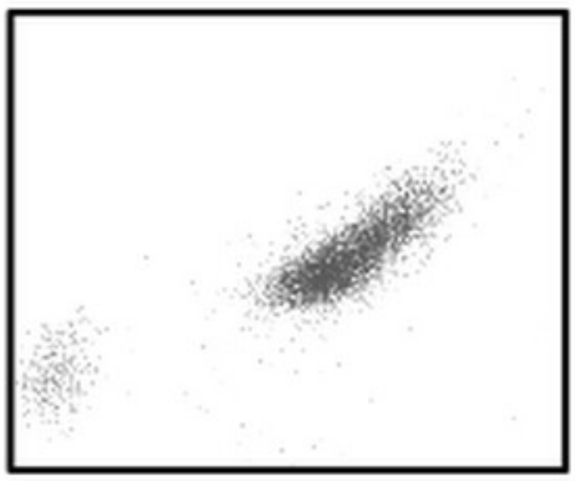


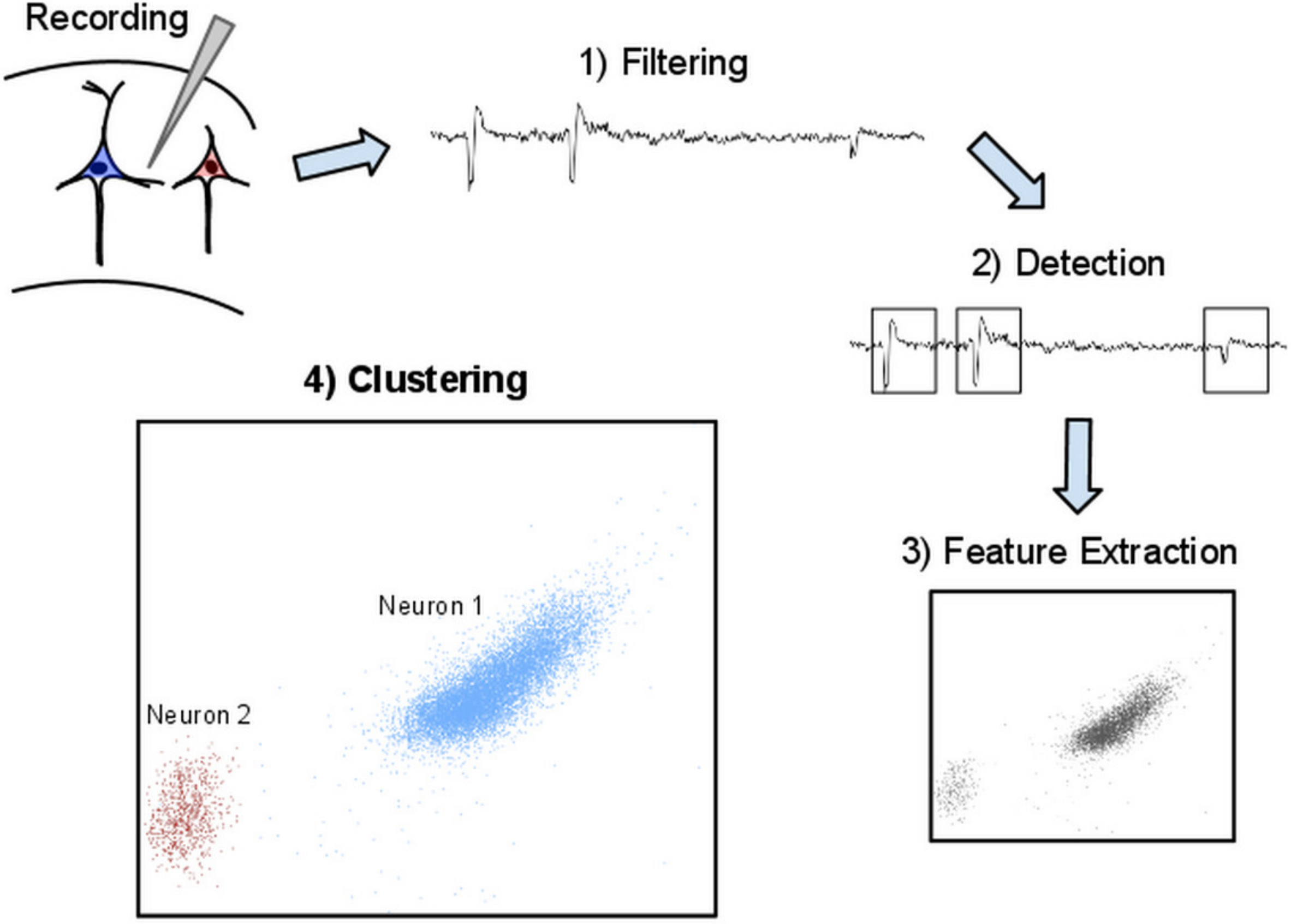
1) Filtering

2) Detection

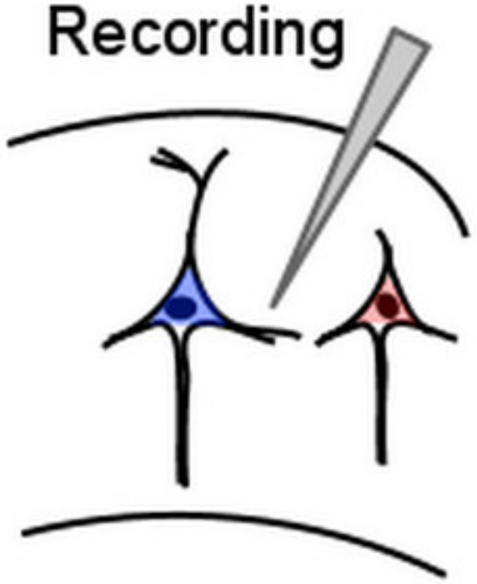


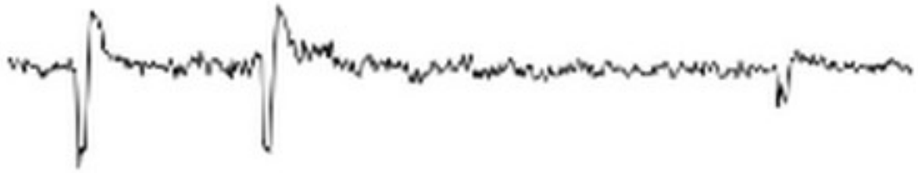
3) Feature Extraction




Recording
1) Filtering
2) Detection
3) Feature Extraction
4) Clustering
Neuron 1
Neuron 2

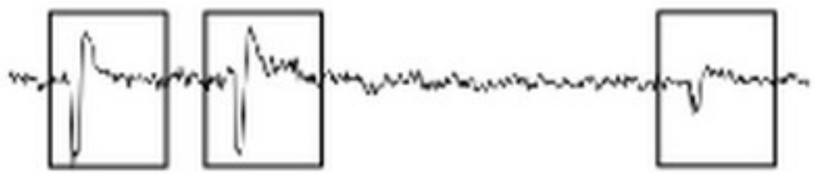
Recording



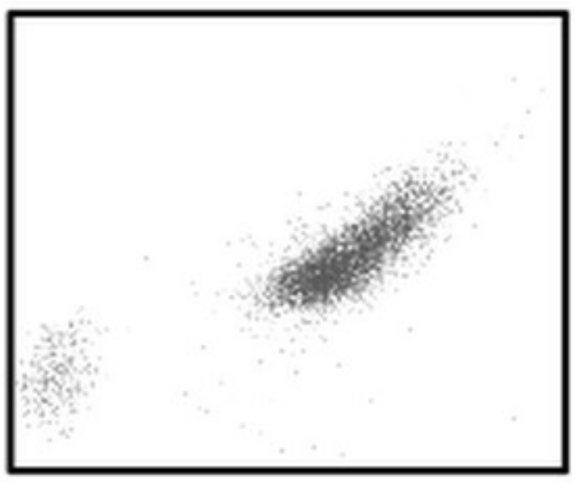


1) Filtering

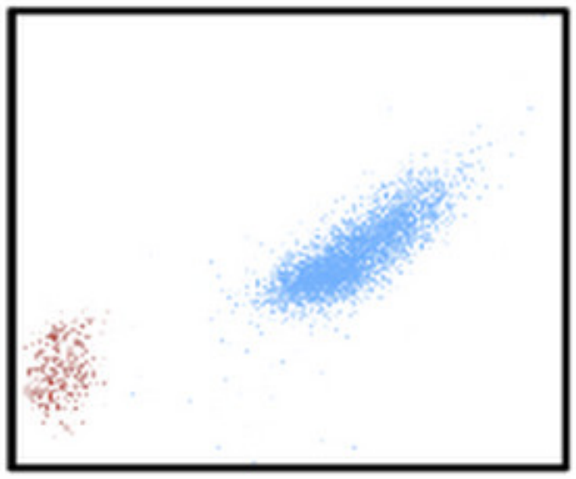
2) Detection



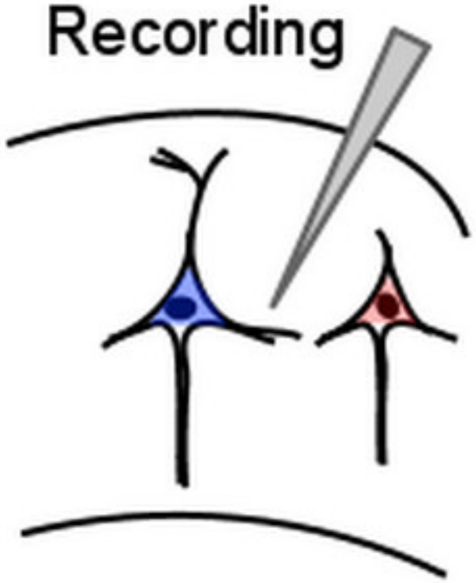
3) Feature Extraction



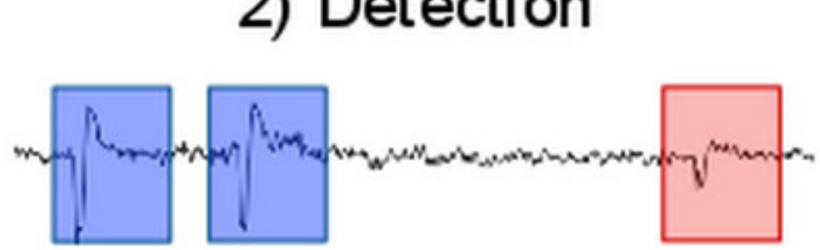
4) Clustering



Recording

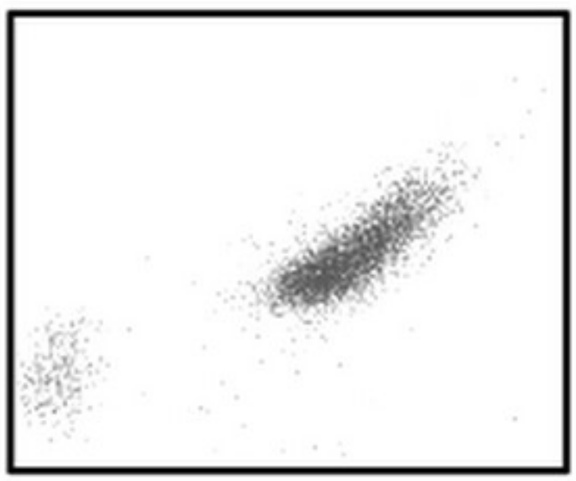


1) Filtering

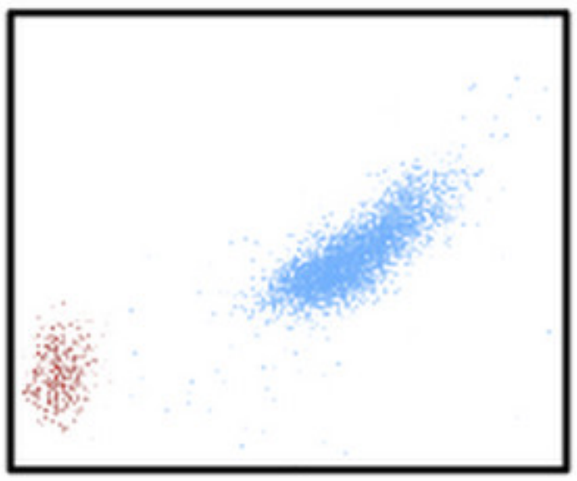


2) Detection

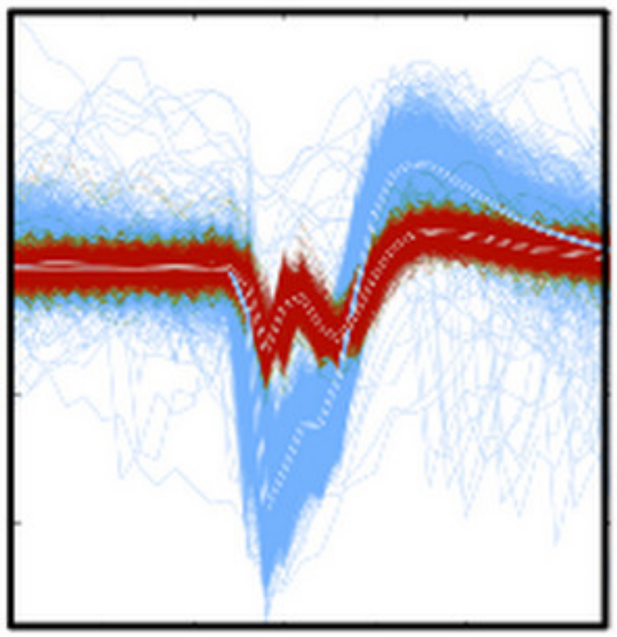
3) Feature Extraction



4) Clustering



Result



modular

modular

customizable

customizable

modular

shell-based

modular

customizable

shell-based

interactive

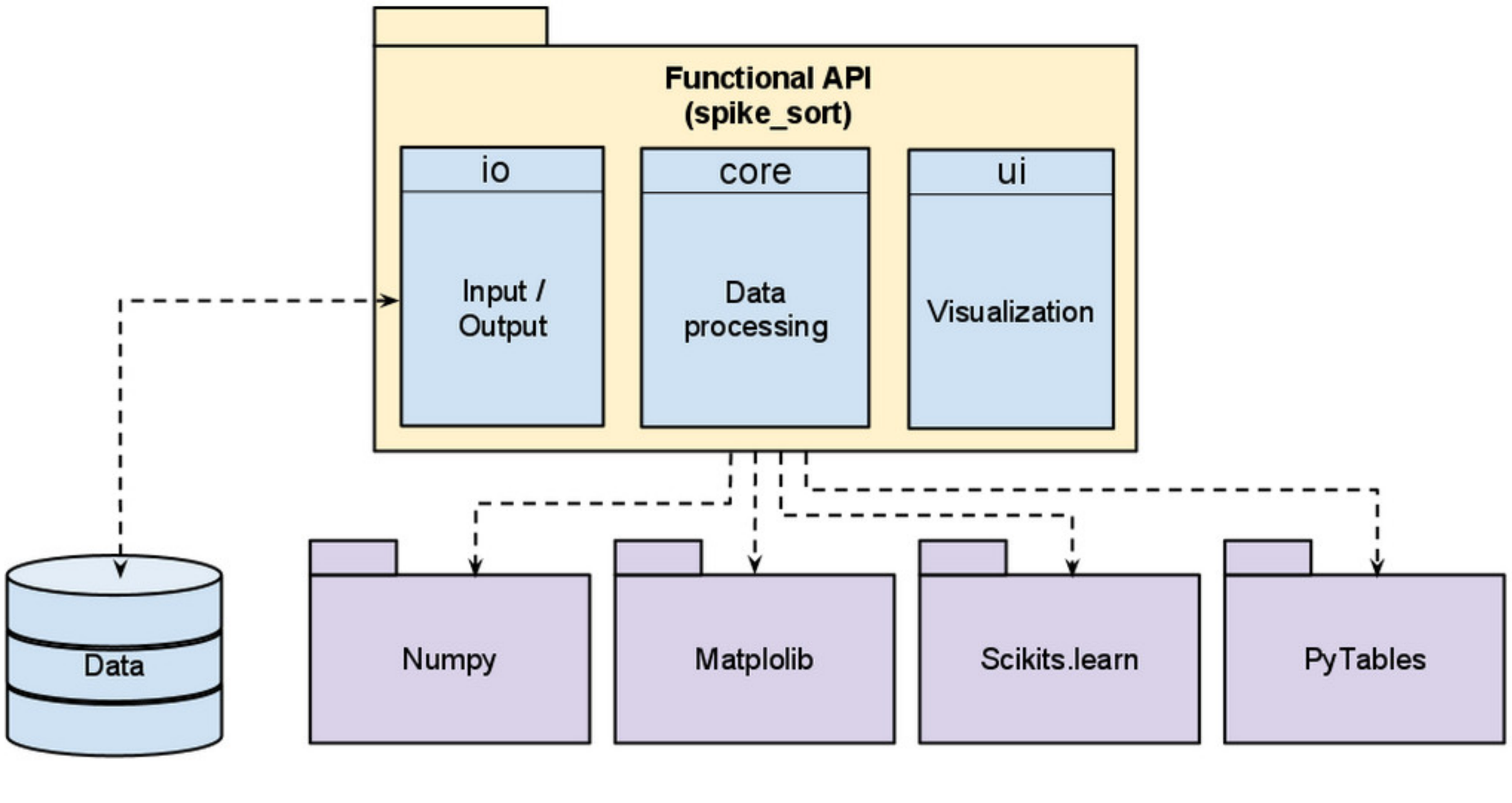
modular

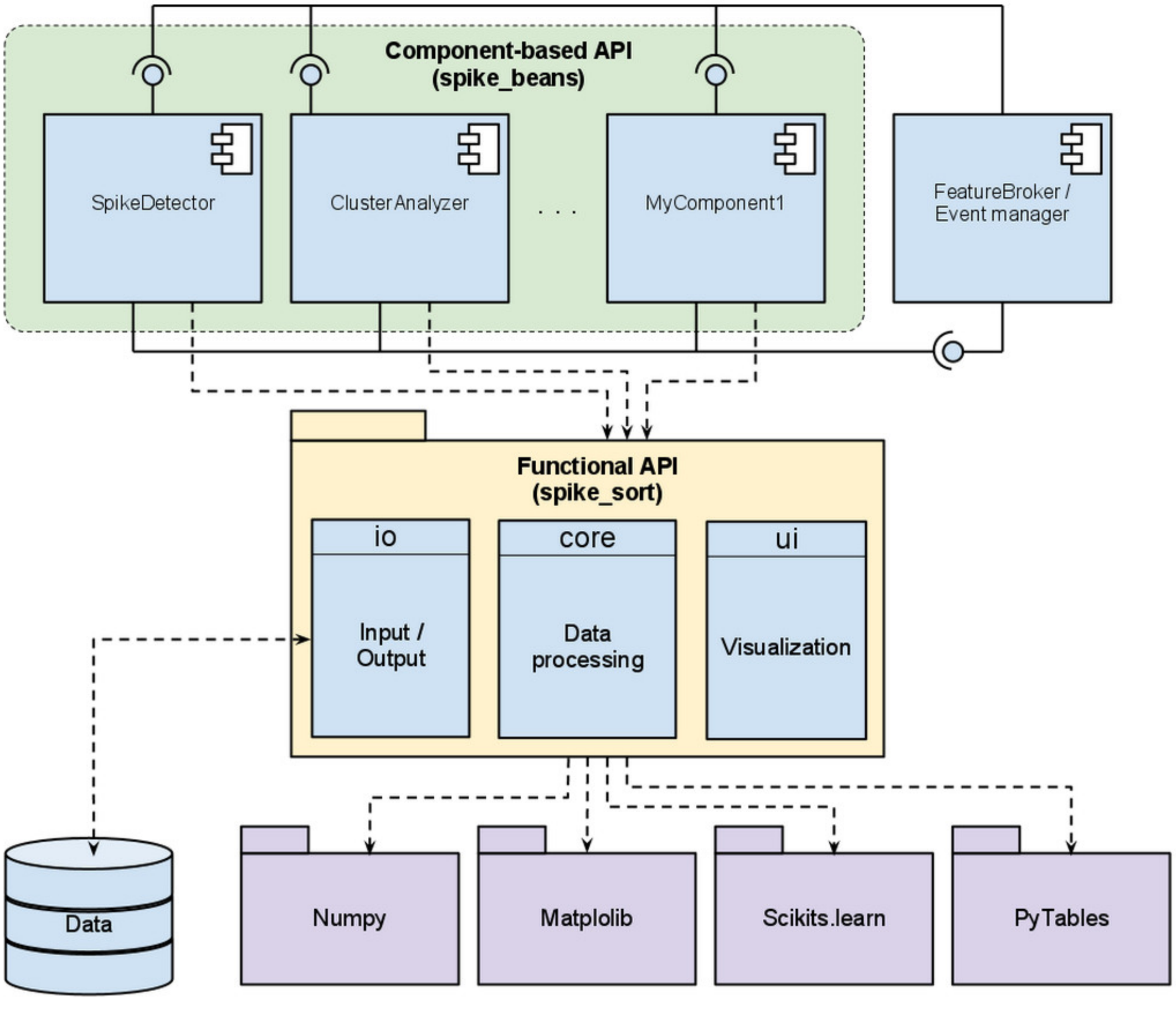
customizable

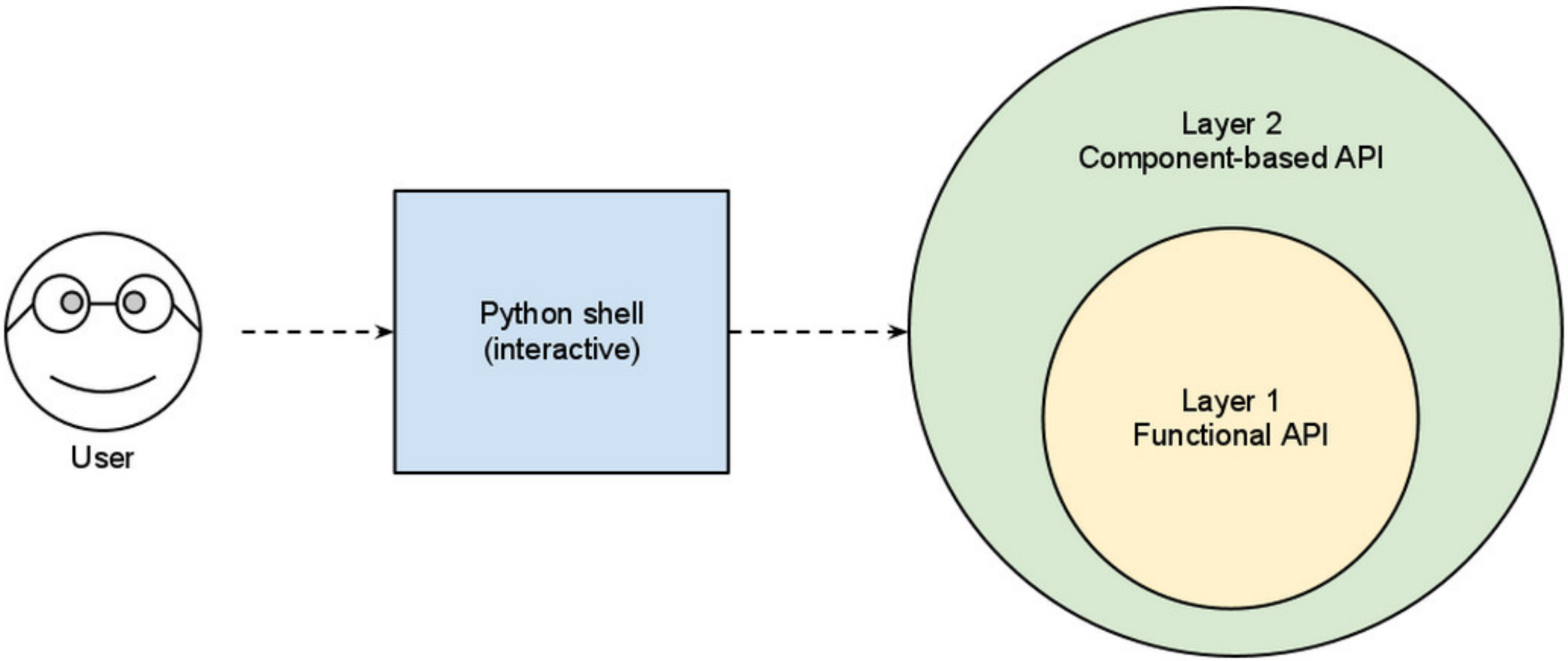
shell-based

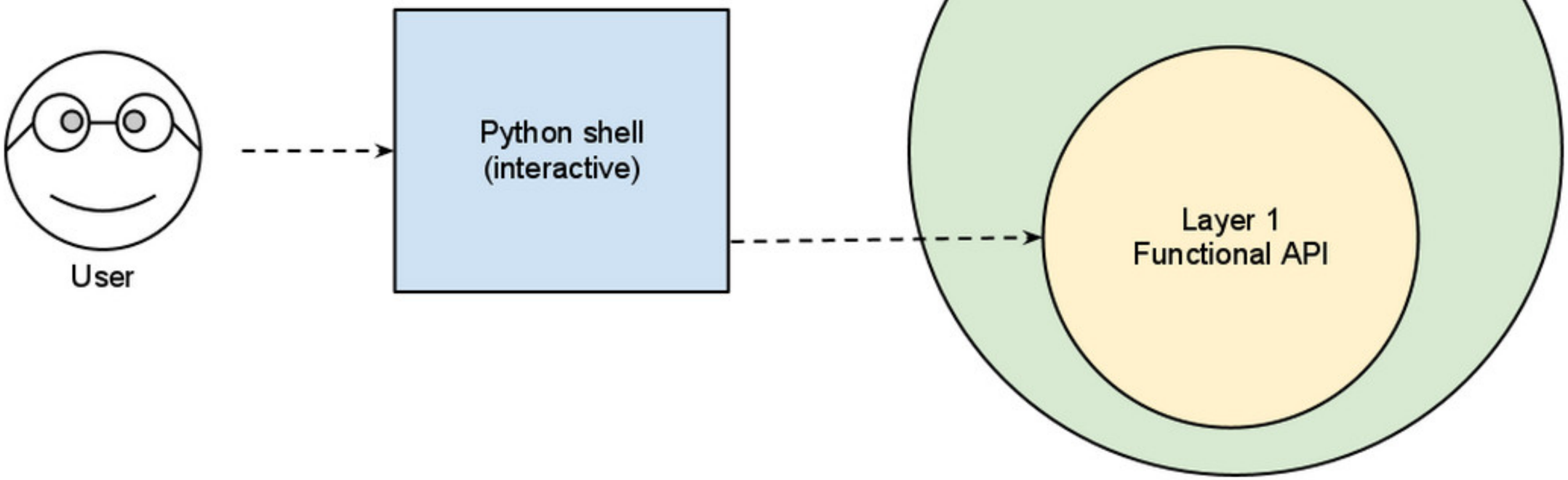
```
$ python -i
```

interactive


Functional API
(spike_sort)
io
Input /
Output
core
Data
processing
ui
Visualization
Data
Numpy
Matplolib
Scikits.learn
PyTables


Component-based API
(spike_beans)
SpikeDetector
ClusterAnalyzer
. . .
MyComponent1
FeatureBroker /
Event manager
Functional API
(spike_sort)
io
Input /
Output
core
Data
processing
ui
Visualization
Data
Numpy
Matplolib
Scikits.learn
PyTables


User
Python shell
(interactive)
Layer 2
Component-based API
Layer 1
Functional API


User
Python shell
(interactive)
Layer 2
Component-based API
Layer 1
Functional API

Implemented components

Input / Output:

- BakerlabSource
- PyTablesSource
- ExportCells

Data processing:

- *SpikeDetector*
- *SpikeExtractor*
- *FeatureExtractor*
- *ClusterAnalyzer*

Visualization:

- *SpikeBrowser*
- *PlotFeatures*
- *PlotSpikes*
- *Legend*

Програми Місця Система

23 °C вт 23 сер, 17:35 dmytro США

dmytro@dmytro-R580: ~/work/SpikeSort/exam

Файл Правка Вигляд Знайти Термінал Довідка

```
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>> browser.show()
>>> features.show()
>>> waveShapes.show()
>>> legend.show()
>>>
```

Figure 3

Cell 4

Cell 3

Cell 2

Cell 1

Figure 2

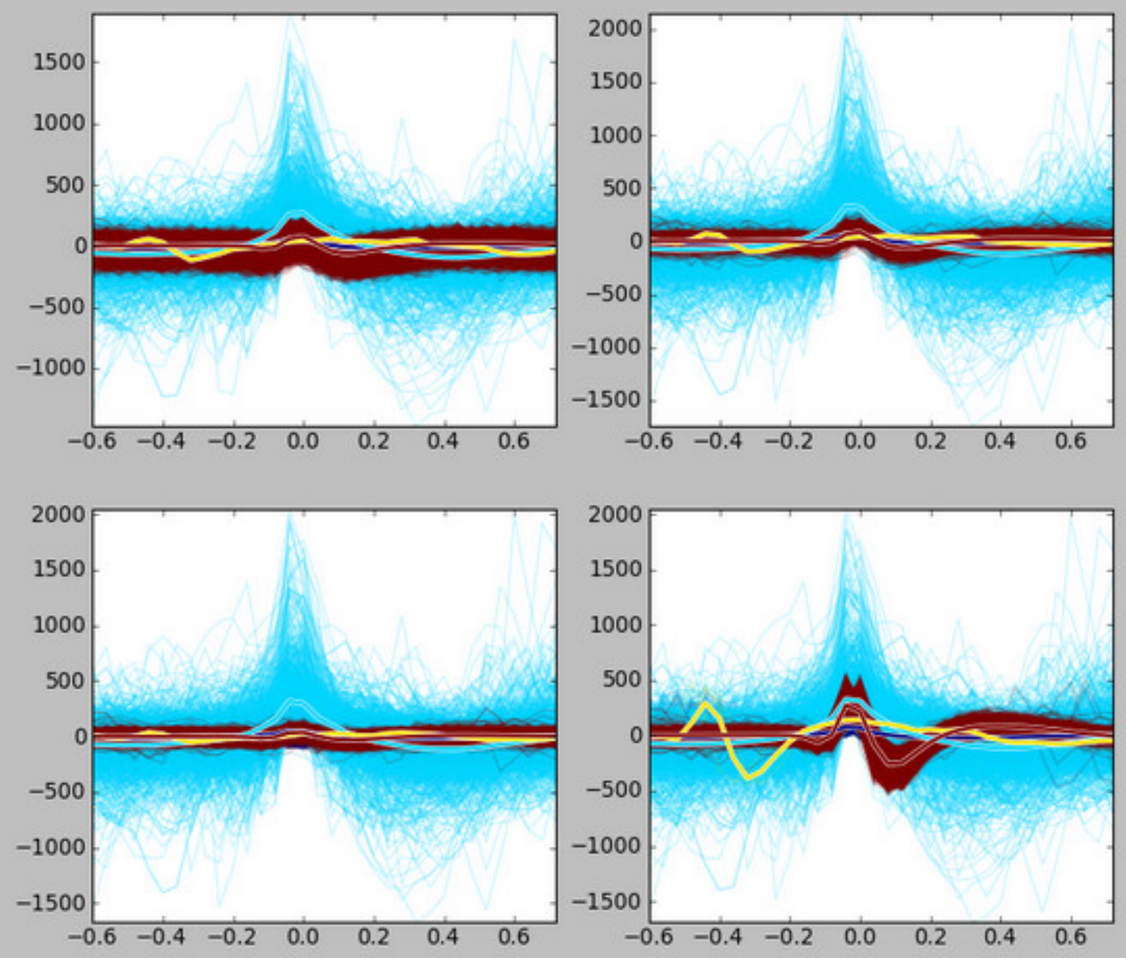
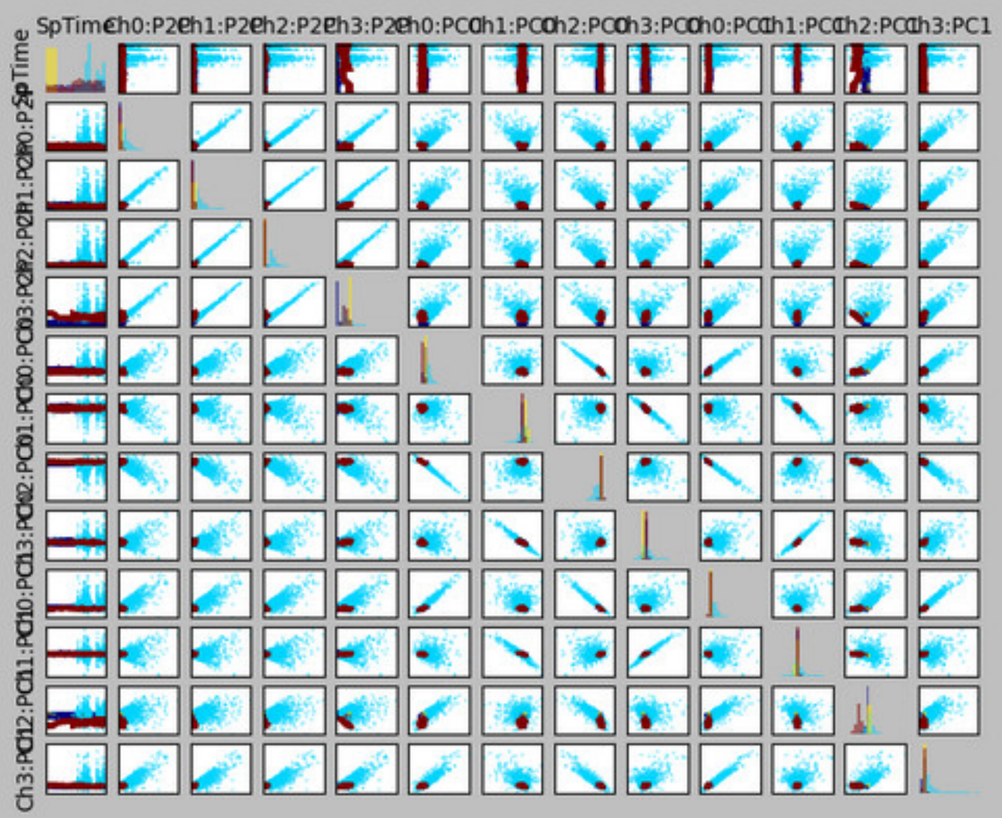
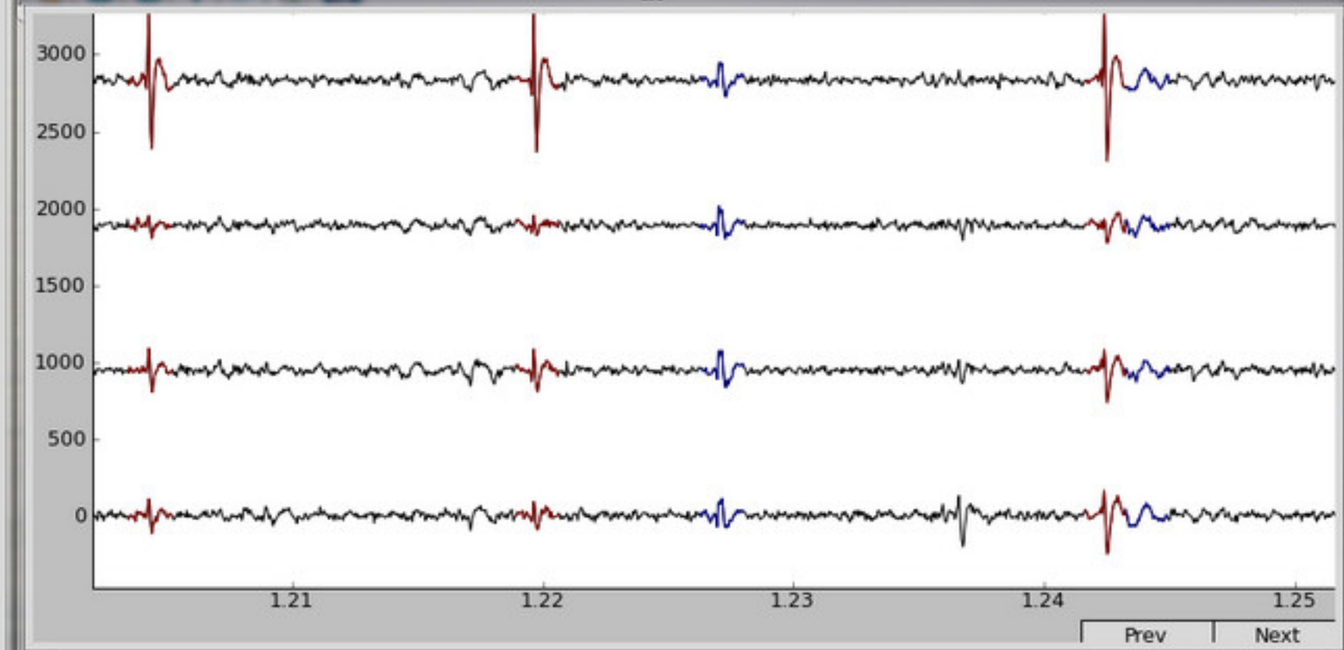


Figure 1



tk



Програми Місця Система

23 °C вт 23 сер, 17:37 dmytro США

dmytro@dmytro-R580: ~/work/SpikeSort/exam

Файл Правка Вигляд Знайти Термінал Довідка

```
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>> browser.show()
>>> features.show()
>>> waveShapes.show()
>>> legend.show()
>>> clusterAnalyzer.delete_cells(2)
>>>
```

Figure 3

Cell 4

Cell 3

Cell 1

Cell 0

Figure 1

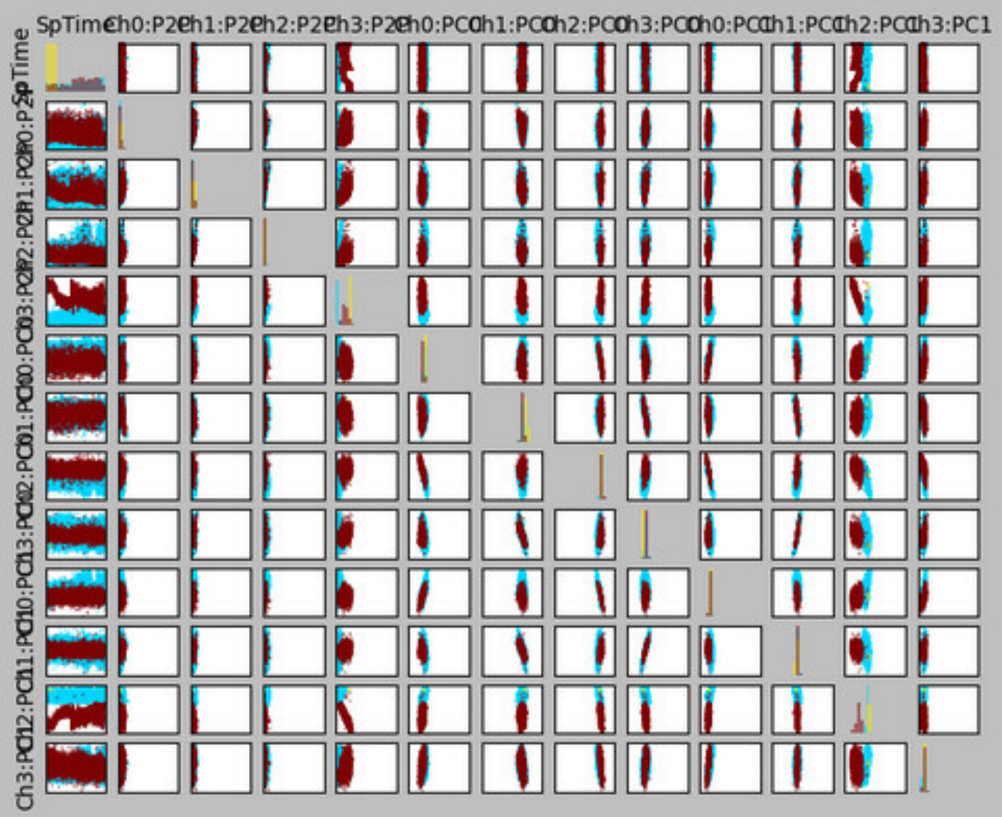
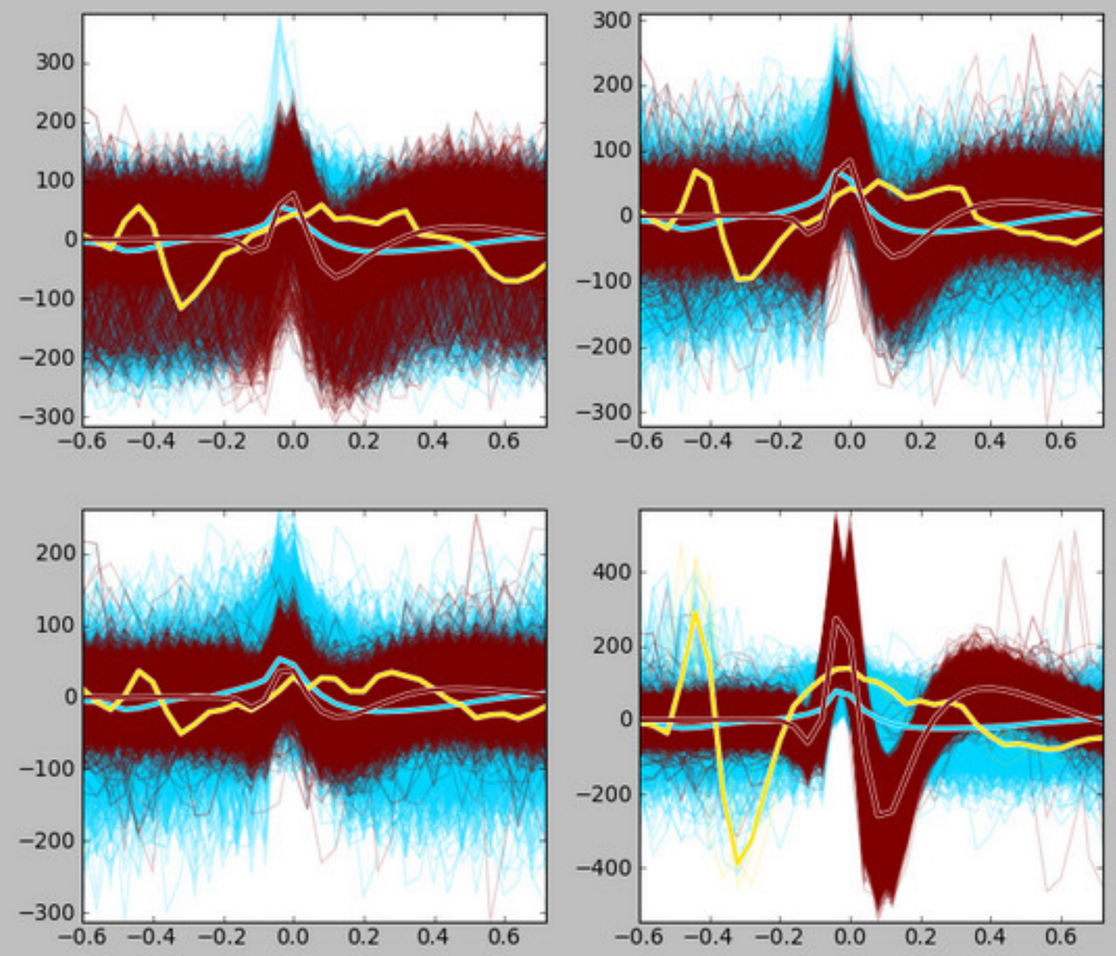
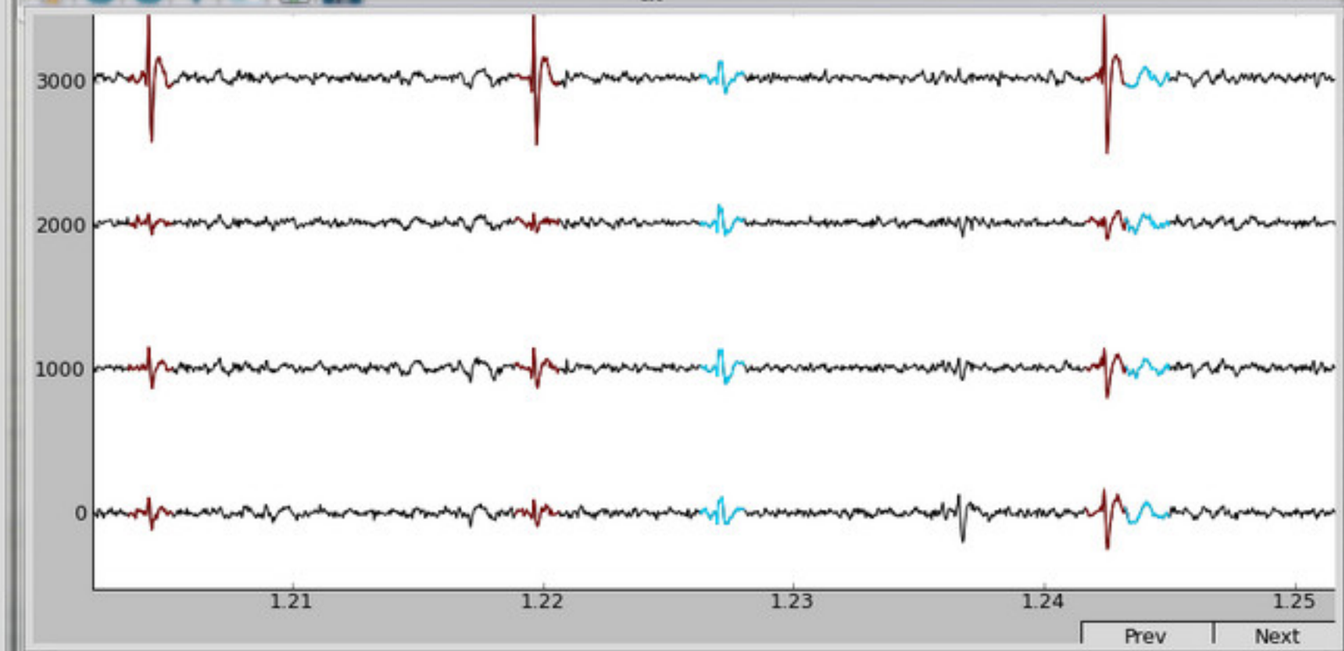


Figure 2



tk



Програми Місця Система

23 °C вт 23 сер, 17:39 dmytro США

dmytro@dmytro-R580: ~/work/SpikeSort/exam

Файл Правка Вигляд Знайти Термінал Довідка

```
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>>
>>> browser.show()
>>> features.show()
>>> waveShapes.show()
>>> legend.show()
>>> clusterAnalyzer.delete_cells(2)
>>> clusterAnalyzer.delete_cells(1,3)
>>>
```

Figure 3

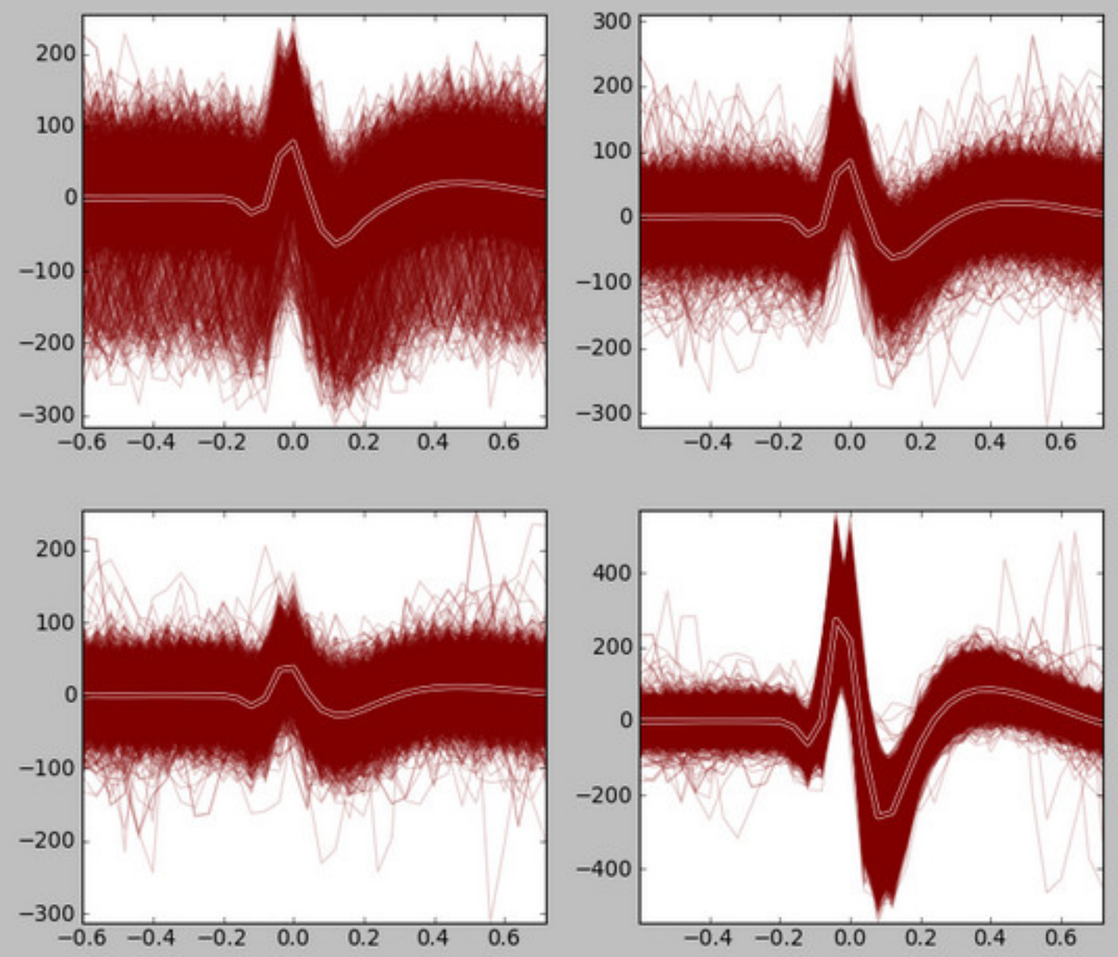
Cell 4

Cell 0

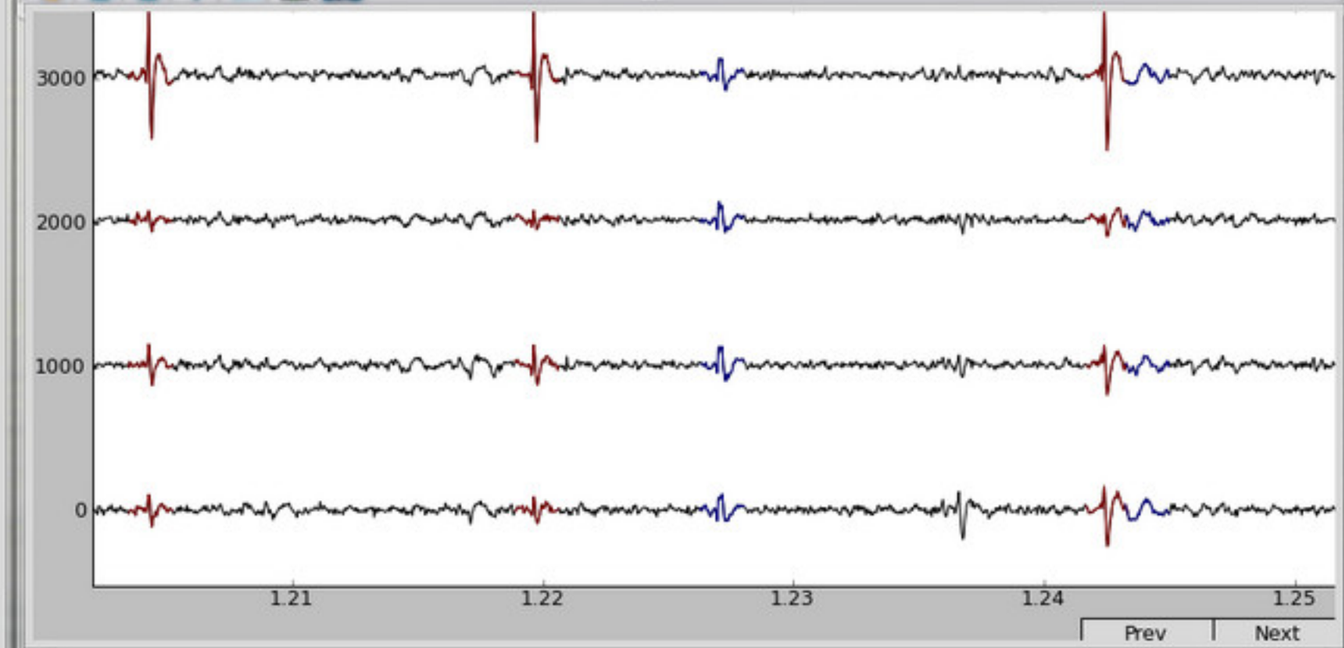
Figure 1

SpTime Ch0:P2 Ch1:P2 Ch2:P2 Ch3:P2 Ch0:PC0 Ch1:PC0 Ch2:PC0 Ch3:PC0 Ch0:PC1 Ch1:PC1 Ch2:PC1 Ch3:PC1

Figure 2



tk



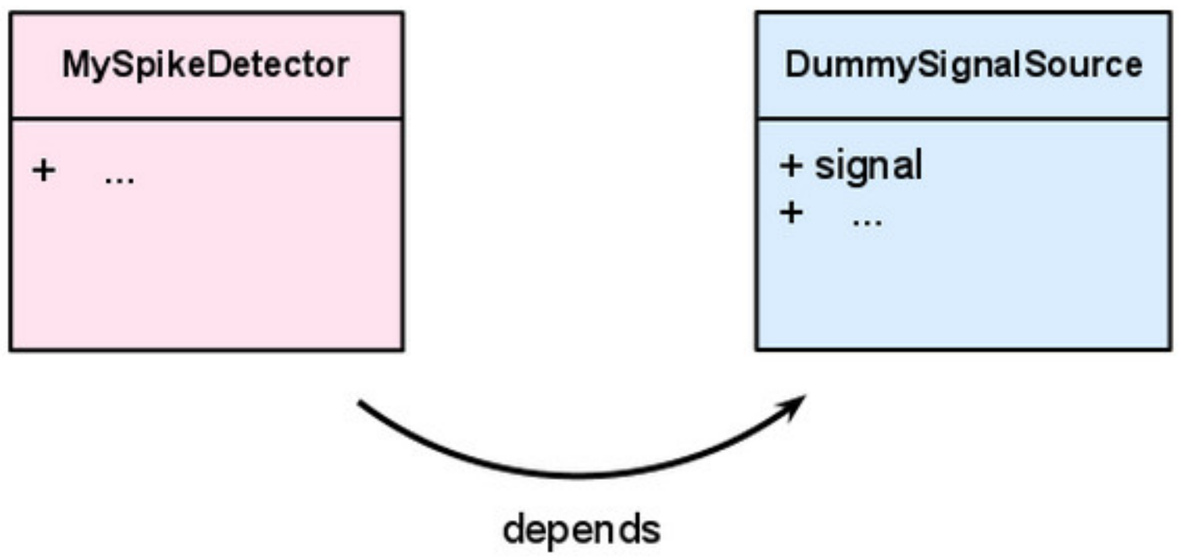
Data inputs:

Clustering:

Visualization:

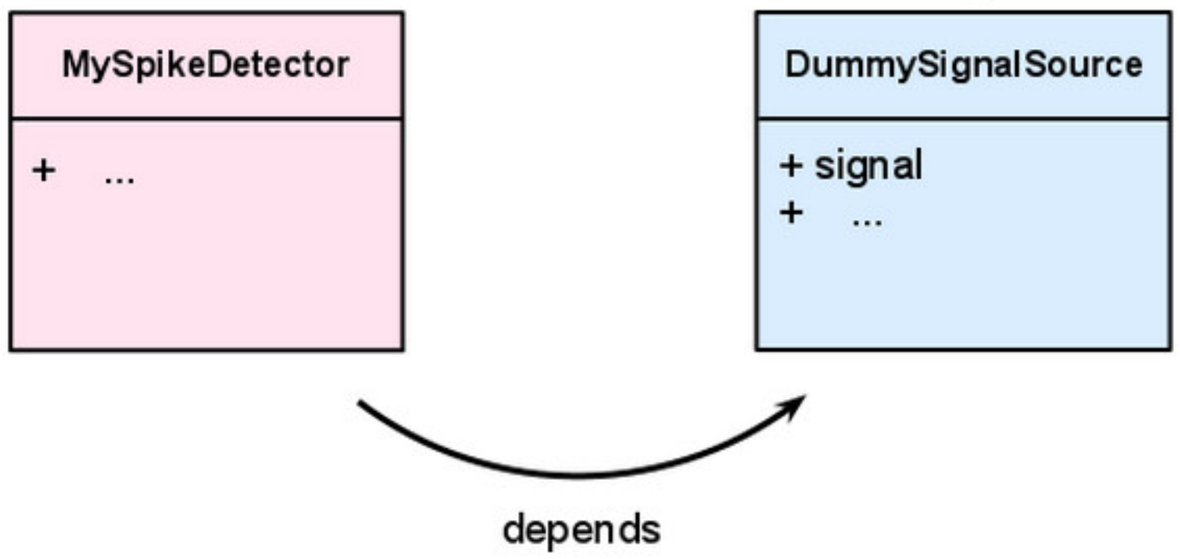
Thanks for your attention!

SFB 618 (project B4)


MySpikeDetector
+ ...
DummySignalSource
+ signal
+ ...
depends

```
class DummySignalSource(base.Component):

    def __init__(self):
        signal = None
```

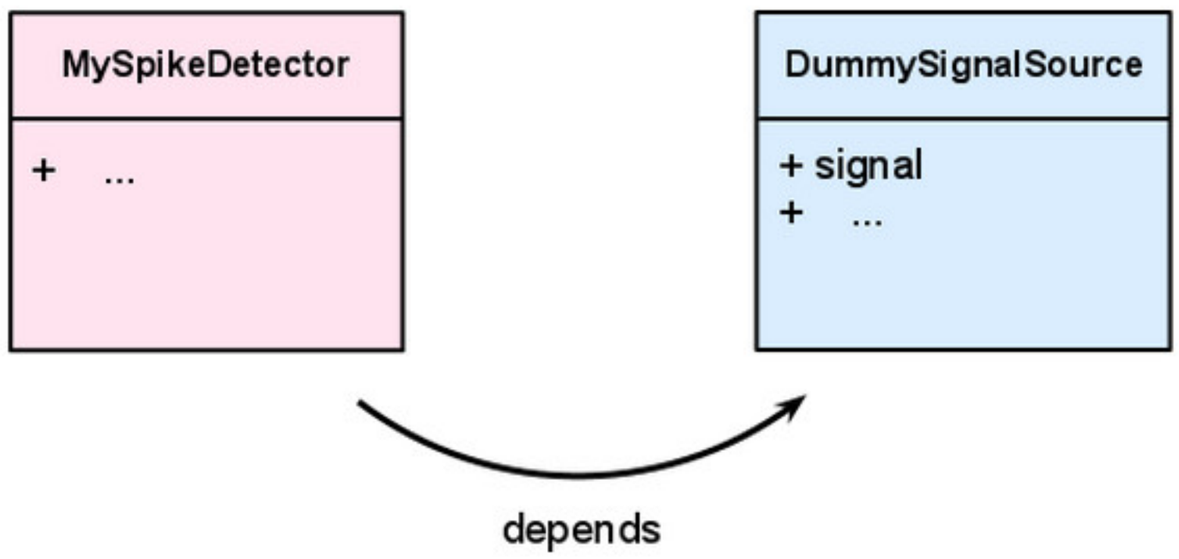
```
class MySpikeDetector(base.Component):

    signalSource = base.RequiredFeature("SignalSource",
                                        base.HasAttributes("signal"))

base.features.Provide("SignalSource", DummySignalSource())

detector = MySpikeDetector()

detector.update()
```



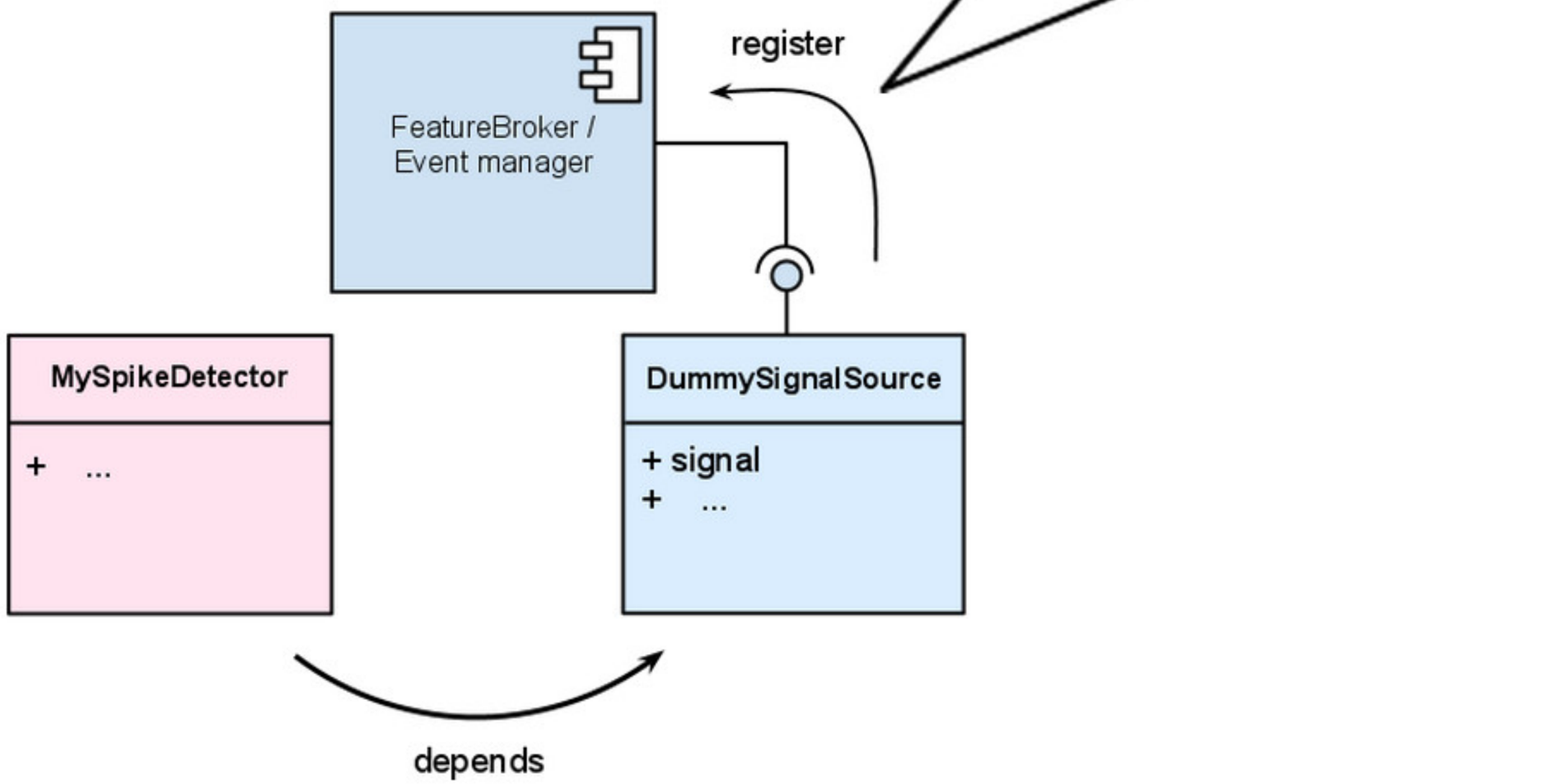
```
class DummySignalSource(base.Component):

    def __init__(self):
        signal = None
```

```
class MySpikeDetector(base.Component):

    signalSource = base.RequiredFeature("SignalSource",
                                        base.HasAttributes("signal"))
```

```
base.features.Provide("SignalSource", DummySignalSource())

detector = MySpikeDetector()

detector.update()
```



```
class DummySignalSource(base.Component):

    def __init__(self):
        signal = None

class MySpikeDetector(base.Component):

    signalSource = base.RequiredFeature("SignalSource",
                                        base.HasAttributes("signal"))

base.features.Provide("SignalSource", DummySignalSource())

detector = MySpikeDetector()

detector.update()
```



```
class DummySignalSource(base.Component):

    def __init__(self):
        signal = None

class MySpikeDetector(base.Component):

    signalSource = base.RequiredFeature("SignalSource",
                                        base.HasAttributes("signal"))

base.features.Provide("SignalSource", DummySignalSource())

detector = MySpikeDetector()

detector.update()
```