Pynapple : Python Neural Analysis package

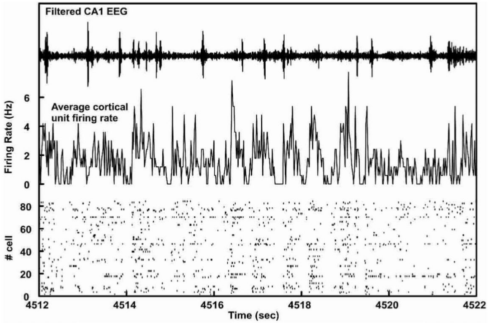
Flatiron Workshop, January 2025

Guillaume Viejo

FLATIRON
INSTITUTE

Where does pynapple comes from?

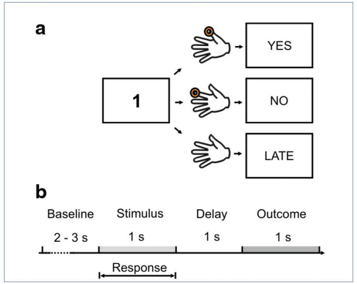
A possible classification of experiments



Hippocampal sharp wave bursts coincide with neocortical “up-state” transitions

Francesco P. Battaglia,[1] Gary R. Sutherland, and Bruce L. McNaughton[2]

Arizona Research Laboratories–Division of Neural Systems, Memory, and Aging, University of Arizona, Tucson, Arizona 85724, USA

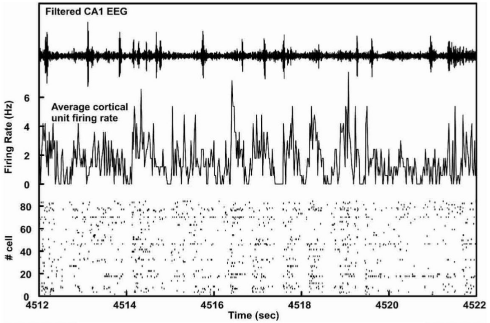


Behavioral/Cognitive

Characterization of Cortical Networks and Corticocortical Functional Connectivity Mediating Arbitrary Visuomotor Mapping

Andrea Brovelli,[1] Daniel Chicharro,[2] Jean-Michel Badier,[3,4] Huifang Wang,[3,4] and Viktor Jirsa[3,4]

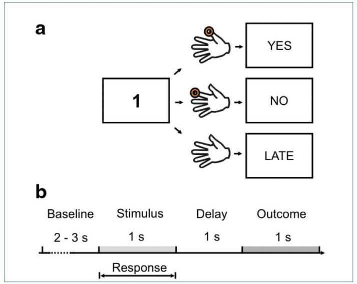
A possible classification of experiments



Hippocampal sharp wave bursts coincide with neocortical "up-state" transitions

Francesco P. Battaglia,[1] Gary R. Sutherland, and Bruce L. McNaughton[2]

Arizona Research Laboratories–Division of Neural Systems, Memory, and Aging, University of Arizona, Tucson, Arizona 85724, USA



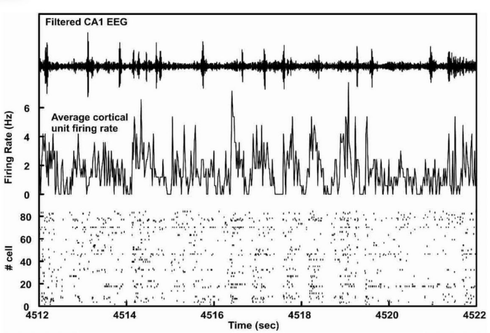
Behavioral/Cognitive

Characterization of Cortical Networks and Corticocortical Functional Connectivity Mediating Arbitrary Visuomotor Mapping

Andrea Brovelli,[1] Daniel Chicharro,[2] Jean-Michel Badier,[3,4] Huifang Wang,[3,4] and Viktor Jirsa[3,4]

Less structured ⟷ More structured

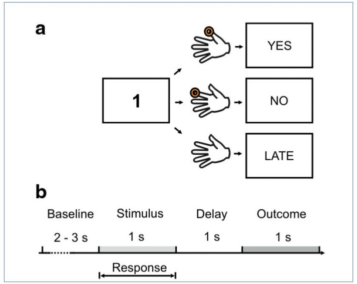
A possible classification of experiments



Hippocampal sharp wave bursts coincide with neocortical “up-state” transitions

Francesco P. Battaglia,[1] Gary R. Sutherland, and Bruce L. McNaughton[2]

Arizona Research Laboratories–Division of Neural Systems, Memory, and Aging, University of Arizona, Tucson, Arizona 85724, USA



Behavioral/Cognitive

Characterization of Cortical Networks and Corticocortical Functional Connectivity Mediating Arbitrary Visuomotor Mapping

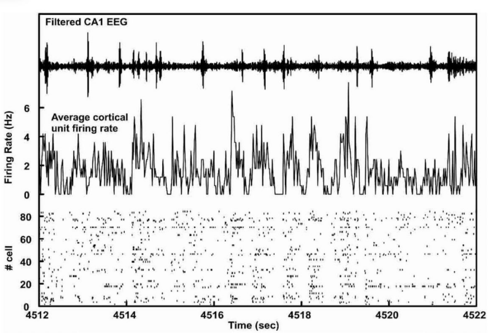
Andrea Brovelli,[1] Daniel Chicharro,[2] Jean-Michel Badier,[3,4] Huifang Wang,[3,4] and Viktor Jirsa[3,4]

Less structured ↔ More structured

→ TsToolbox (matlab)

Francesco Battaglia

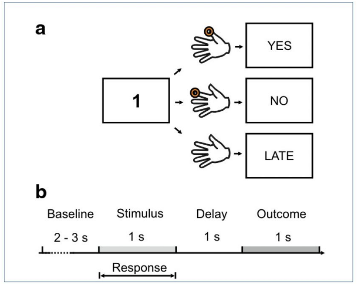
A possible classification of experiments



Hippocampal sharp wave bursts coincide with neocortical "up-state" transitions

Francesco P. Battaglia,[1] Gary R. Sutherland, and Bruce L. McNaughton[2]

Arizona Research Laboratories–Division of Neural Systems, Memory, and Aging, University of Arizona, Tucson, Arizona 85724, USA



Behavioral/Cognitive

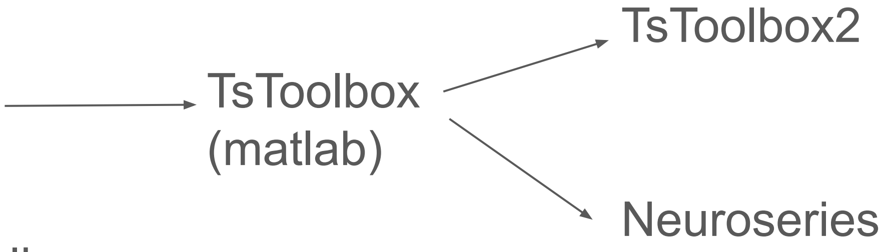
Characterization of Cortical Networks and Corticocortical Functional Connectivity Mediating Arbitrary Visuomotor Mapping

Andrea Brovelli,[1] Daniel Chicharro,[2] Jean-Michel Badier,[3,4] Huifang Wang,[3,4] and Viktor Jirsa[3,4]

Less structured

More structured

Francesco Battaglia

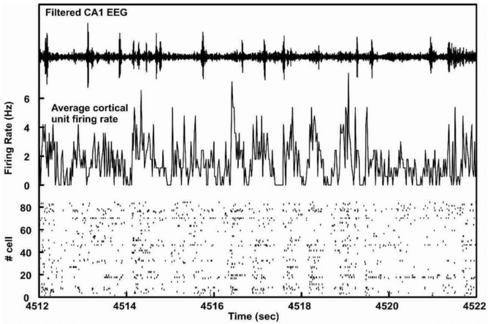


TsToolbox (matlab)

TsToolbox2

Neuroseries

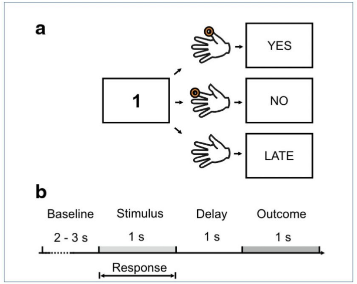
A possible classification of experiments



Hippocampal sharp wave bursts coincide with neocortical “up-state” transitions

Francesco P. Battaglia,[1] Gary R. Sutherland, and Bruce L. McNaughton[2]

Arizona Research Laboratories–Division of Neural Systems, Memory, and Aging, University of Arizona, Tucson, Arizona 85724, USA



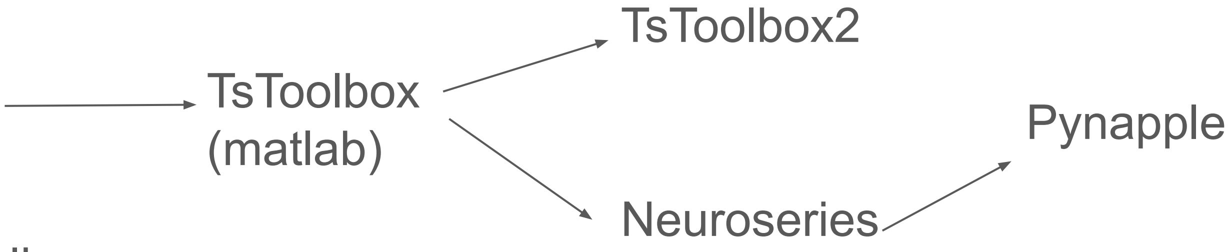
Behavioral/Cognitive

Characterization of Cortical Networks and Corticocortical Functional Connectivity Mediating Arbitrary Visuomotor Mapping

Andrea Brovelli,[1] Daniel Chicharro,[2] Jean-Michel Badier,[3,4] Huifang Wang,[3,4] and Viktor Jirsa[3,4]

Less structured ↔ More structured

Francesco Battaglia



When do I need pynapple?

Time

Preprocessing
(CaImAn, SpikeInterface, …)

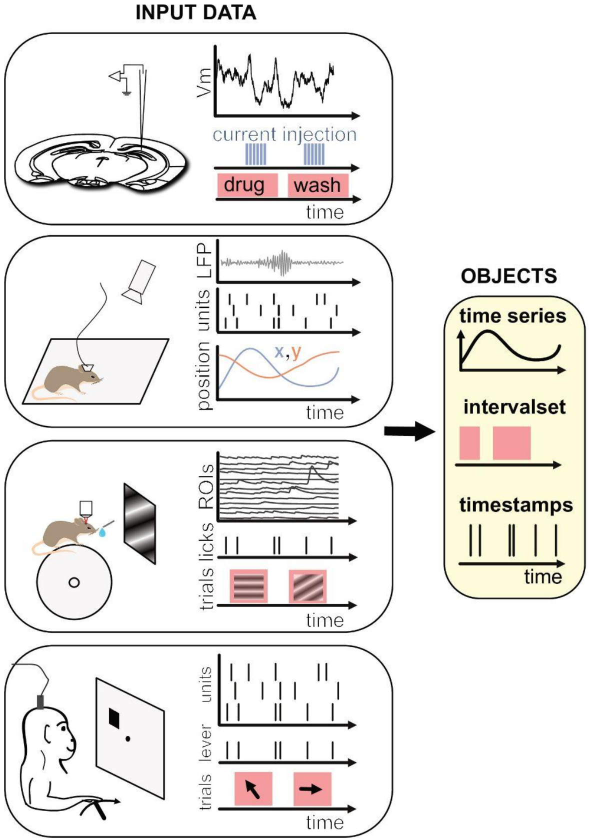
Postprocessing
(GLM, Manifold, …)

Time

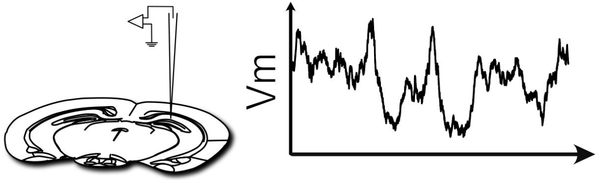
Preprocessing
(CaImAn, SpikeInterface, …)

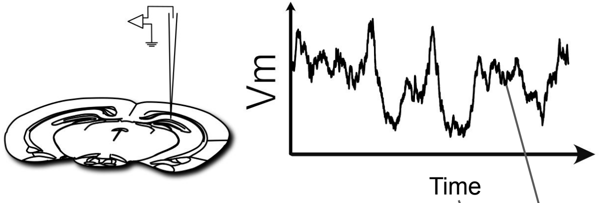
Pynapple

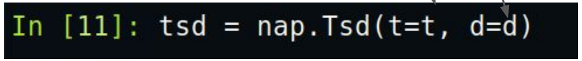
Postprocessing
(GLM, Manifold, …)



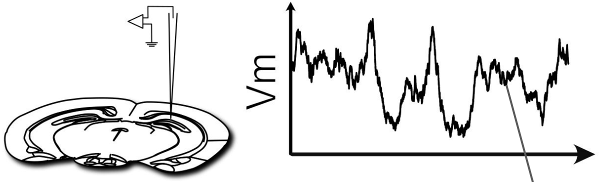
Time Series Data : Tsd, TsdFrame and TsdTensor

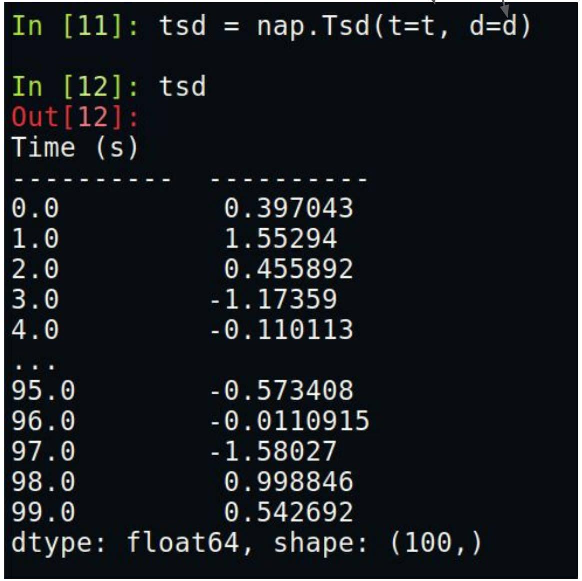

Vm



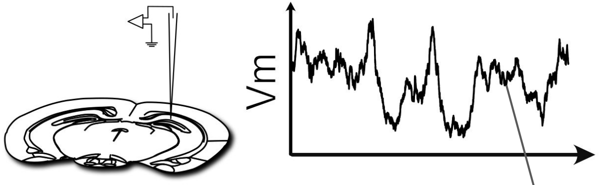


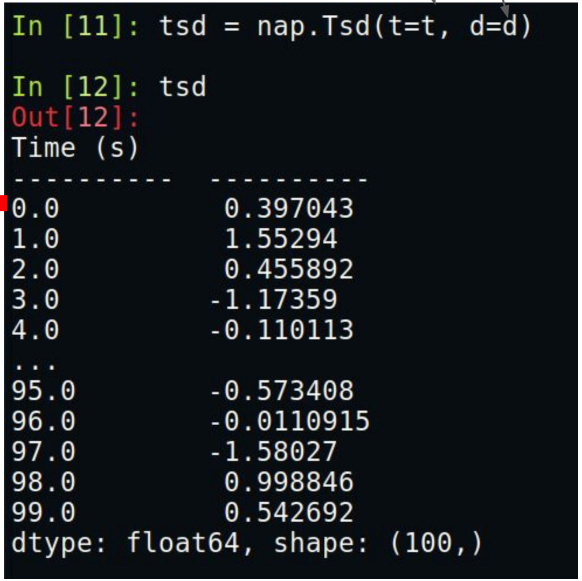
```
In [11]: tsd = nap.Tsd(t=t, d=d)
```





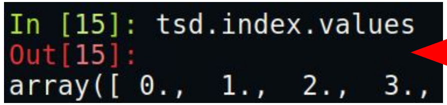
```
In [11]: tsd = nap.Tsd(t=t, d=d)

In [12]: tsd
Out[12]:
Time (s)
----------  ----------
0.0           0.397043
1.0           1.55294
2.0           0.455892
3.0          -1.17359
4.0          -0.110113
...
95.0         -0.573408
96.0         -0.0110915
97.0         -1.58027
98.0          0.998846
99.0          0.542692
dtype: float64, shape: (100,)
```



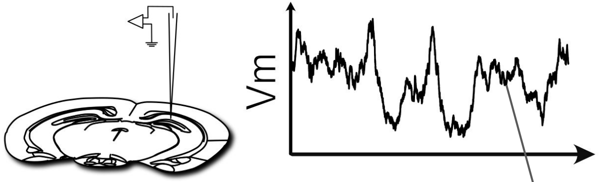


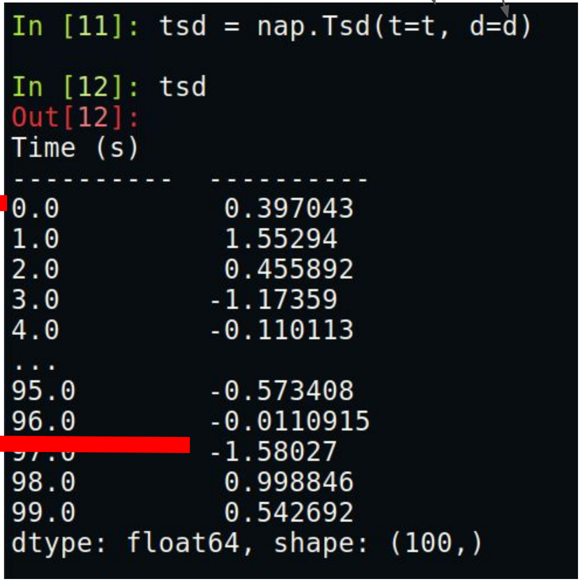
```
In [11]: tsd = nap.Tsd(t=t, d=d)

In [12]: tsd
Out[12]:
Time (s)
----------  ----------
0.0           0.397043
1.0           1.55294
2.0           0.455892
3.0          -1.17359
4.0          -0.110113
...
95.0         -0.573408
96.0         -0.0110915
97.0         -1.58027
98.0          0.998846
99.0          0.542692
dtype: float64, shape: (100,)
```

Numpy array



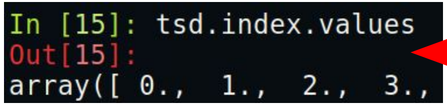
```
In [15]: tsd.index.values
Out[15]:
array([ 0.,   1.,   2.,   3.,
```



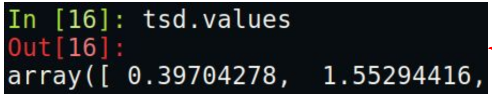


```
In [11]: tsd = nap.Tsd(t=t, d=d)

In [12]: tsd
Out[12]:
Time (s)
----------  ----------
0.0           0.397043
1.0           1.55294
2.0           0.455892
3.0          -1.17359
4.0          -0.110113
...
95.0         -0.573408
96.0         -0.0110915
97.0         -1.58027
98.0          0.998846
99.0          0.542692
dtype: float64, shape: (100,)
```

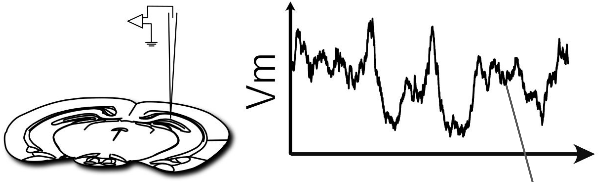
Numpy array



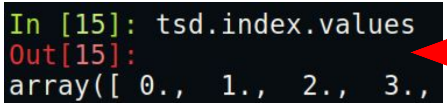
```
In [15]: tsd.index.values
Out[15]:
array([ 0.,   1.,   2.,   3.,
```

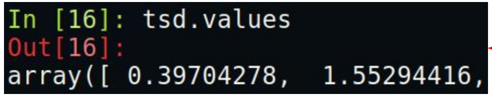


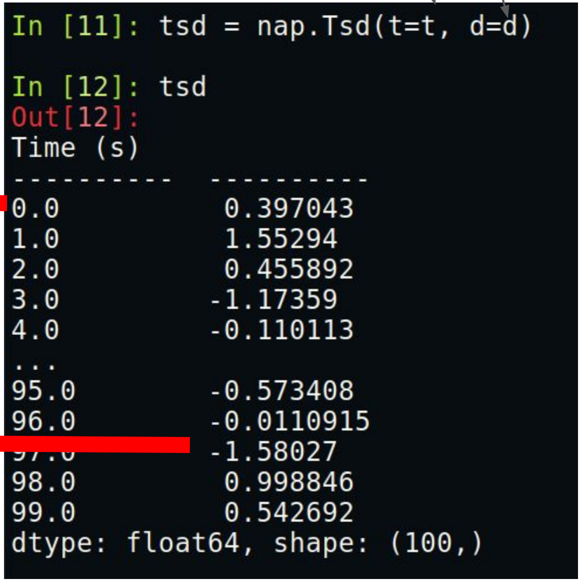
```
In [16]: tsd.values
Out[16]:
array([ 0.39704278,  1.55294416,
```



Numpy array

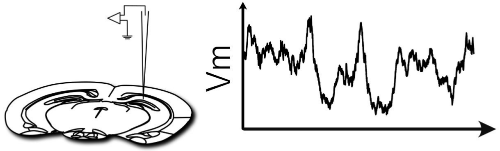

```
In [15]: tsd.index.values
Out[15]:
array([ 0.,   1.,   2.,   3.,
```


```
In [16]: tsd.values
Out[16]:
array([ 0.39704278,   1.55294416,
```

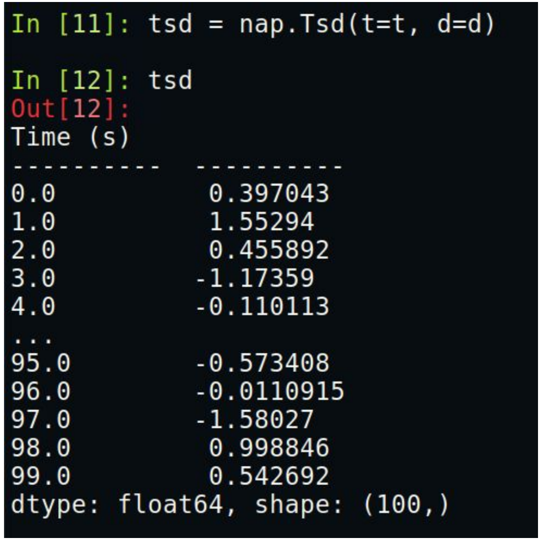

```
In [11]: tsd = nap.Tsd(t=t, d=d)

In [12]: tsd
Out[12]:
Time (s)
----------   ----------
0.0            0.397043
1.0            1.55294
2.0            0.455892
3.0           -1.17359
4.0           -0.110113
...
95.0          -0.573408
96.0          -0.0110915
97.0          -1.58027
98.0           0.998846
99.0           0.542692
dtype: float64, shape: (100,)
```

You gain

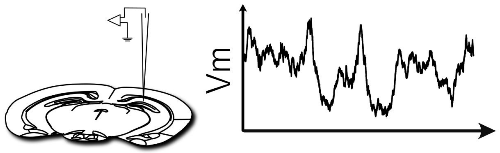
- Automatic handling of time units
- Epoch restriction
- Binning
- Time support
- …



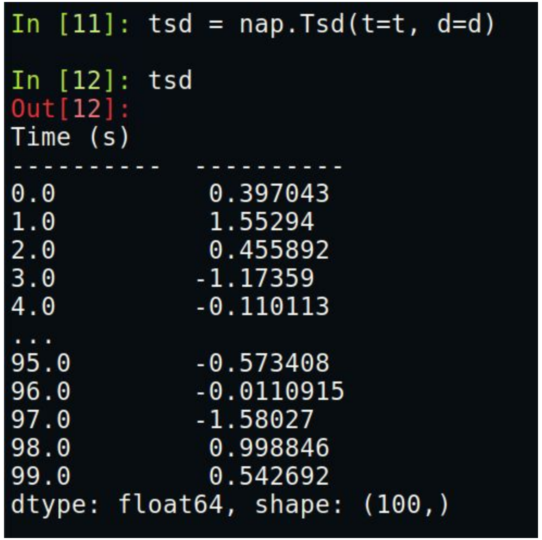
Tsd: 1-dimension



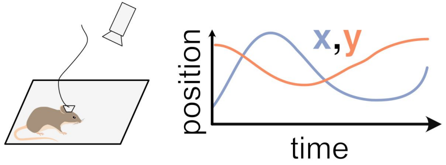
```
In [11]: tsd = nap.Tsd(t=t, d=d)

In [12]: tsd
Out[12]:
Time (s)
----------  ----------
0.0           0.397043
1.0           1.55294
2.0           0.455892
3.0          -1.17359
4.0          -0.110113
...
95.0         -0.573408
96.0         -0.0110915
97.0         -1.58027
98.0          0.998846
99.0          0.542692
dtype: float64, shape: (100,)
```



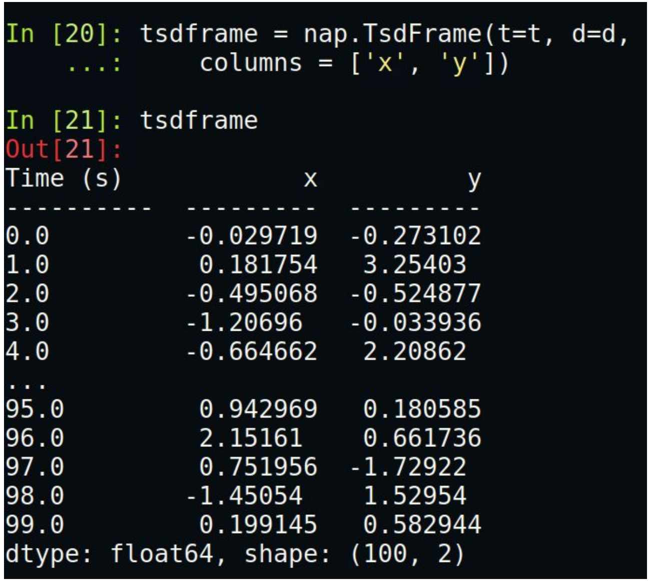
Tsd: 1-dimension



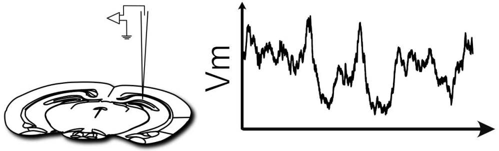
```
In [11]: tsd = nap.Tsd(t=t, d=d)

In [12]: tsd
Out[12]:
Time (s)
----------  ----------
0.0           0.397043
1.0           1.55294
2.0           0.455892
3.0          -1.17359
4.0          -0.110113
...
95.0         -0.573408
96.0         -0.0110915
97.0         -1.58027
98.0          0.998846
99.0          0.542692
dtype: float64, shape: (100,)
```

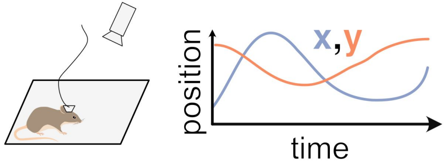


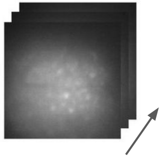
TsdFrame: 2-dimensions



```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:     columns = ['x', 'y'])

In [21]: tsdframe
Out[21]:
Time (s)            x          y
----------  ---------  ---------
0.0         -0.029719  -0.273102
1.0          0.181754   3.25403
2.0         -0.495068  -0.524877
3.0         -1.20696   -0.033936
4.0         -0.664662   2.20862
...
95.0         0.942969   0.180585
96.0         2.15161    0.661736
97.0         0.751956  -1.72922
98.0        -1.45054    1.52954
99.0         0.199145   0.582944
dtype: float64, shape: (100, 2)
```

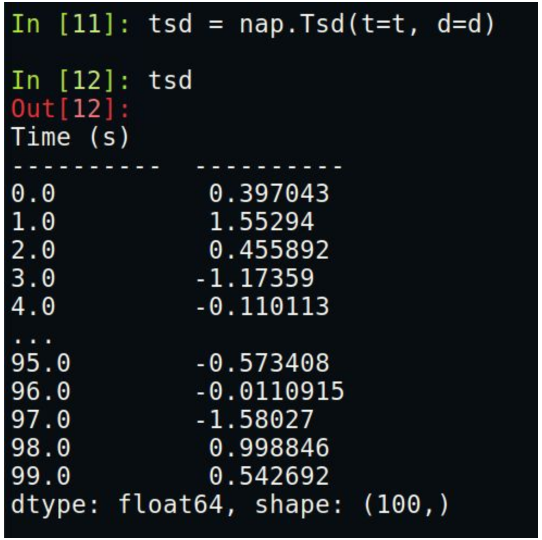






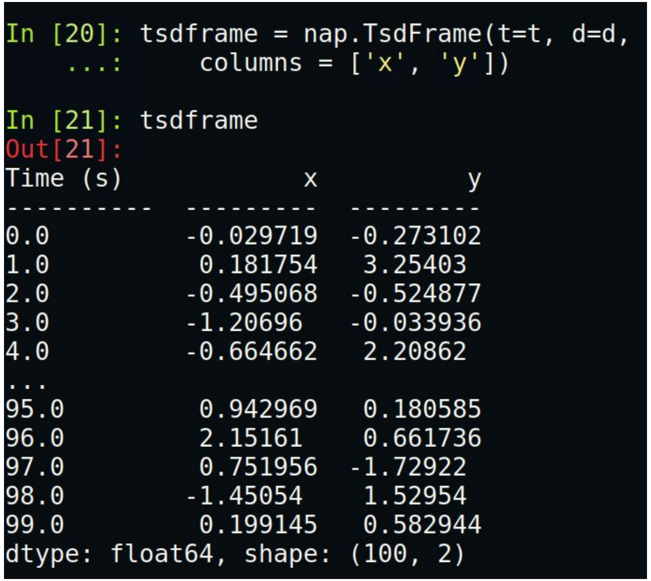
FLATIRON INSTITUTE

Tsd: 1-dimension



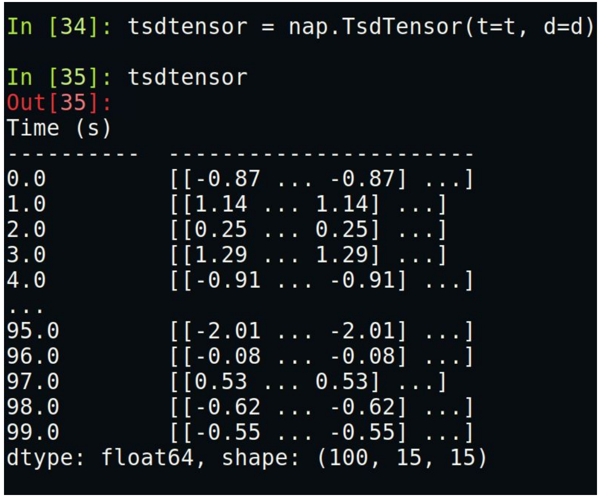
```
In [11]: tsd = nap.Tsd(t=t, d=d)

In [12]: tsd
Out[12]:
Time (s)
----------  ----------
0.0           0.397043
1.0           1.55294
2.0           0.455892
3.0          -1.17359
4.0          -0.110113
...
95.0         -0.573408
96.0         -0.0110915
97.0         -1.58027
98.0          0.998846
99.0          0.542692
dtype: float64, shape: (100,)
```

TsdFrame: 2-dimensions

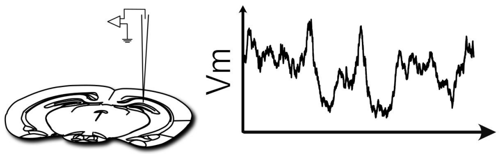


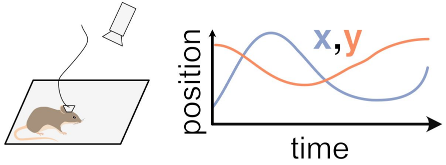
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:     columns = ['x', 'y'])

In [21]: tsdframe
Out[21]:
Time (s)            x          y
----------  ---------  ---------
0.0         -0.029719  -0.273102
1.0          0.181754   3.25403
2.0         -0.495068  -0.524877
3.0         -1.20696   -0.033936
4.0         -0.664662   2.20862
...
95.0         0.942969   0.180585
96.0         2.15161    0.661736
97.0         0.751956  -1.72922
98.0        -1.45054    1.52954
99.0         0.199145   0.582944
dtype: float64, shape: (100, 2)
```

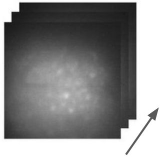
TsdTensor: n-dimensions



```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)

In [35]: tsdtensor
Out[35]:
Time (s)
----------  ---------------------
0.0         [[-0.87 ... -0.87] ...]
1.0         [[1.14 ... 1.14] ...]
2.0         [[0.25 ... 0.25] ...]
3.0         [[1.29 ... 1.29] ...]
4.0         [[-0.91 ... -0.91] ...]
...
95.0        [[-2.01 ... -2.01] ...]
96.0        [[-0.08 ... -0.08] ...]
97.0        [[0.53 ... 0.53] ...]
98.0        [[-0.62 ... -0.62] ...]
99.0        [[-0.55 ... -0.55] ...]
dtype: float64, shape: (100, 15, 15)
```



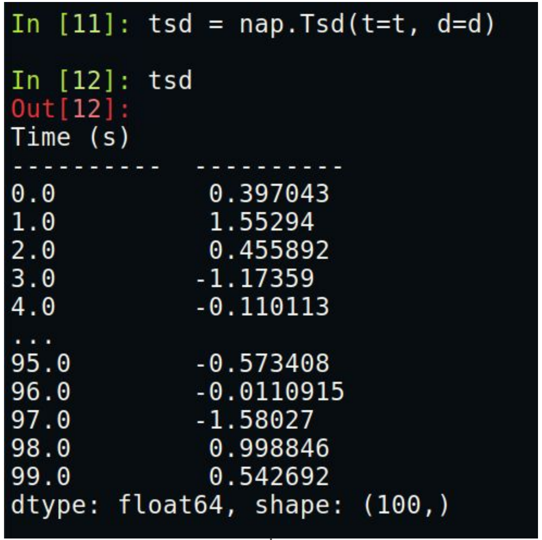




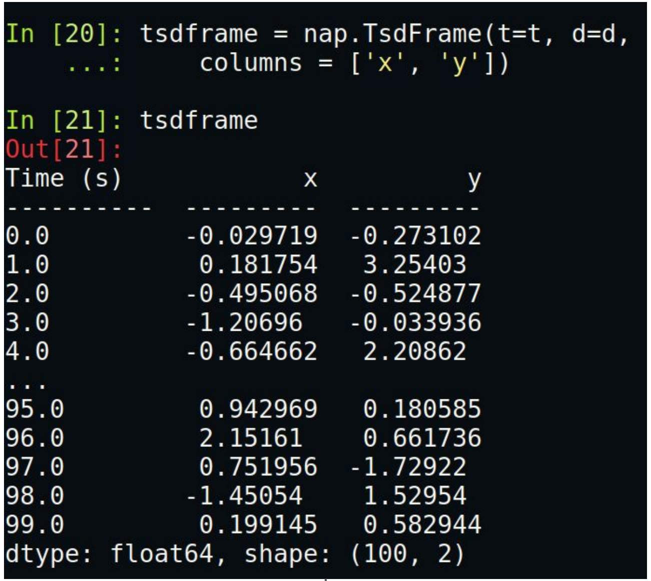
Tsd: 1-dimension

TsdFrame: 2-dimensions

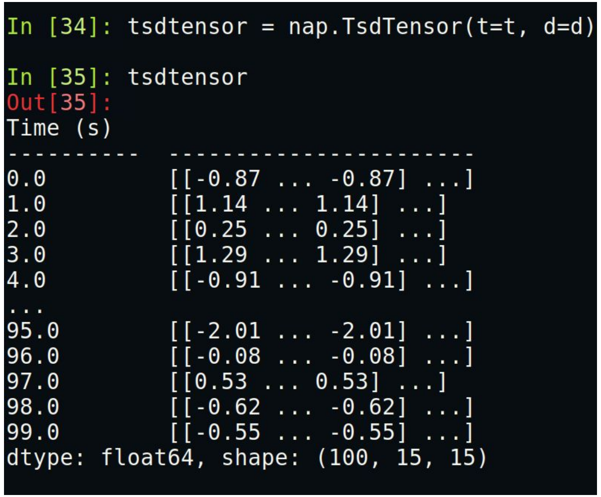
TsdTensor: n-dimensions



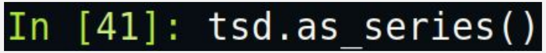
```
In [11]: tsd = nap.Tsd(t=t, d=d)

In [12]: tsd
Out[12]:
Time (s)
----------  ----------
0.0           0.397043
1.0           1.55294
2.0           0.455892
3.0          -1.17359
4.0          -0.110113
...
95.0         -0.573408
96.0         -0.0110915
97.0         -1.58027
98.0          0.998846
99.0          0.542692
dtype: float64, shape: (100,)
```



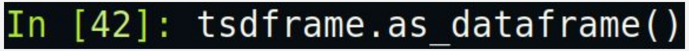
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:     columns = ['x', 'y'])

In [21]: tsdframe
Out[21]:
Time (s)            x          y
----------  ---------  ---------
0.0         -0.029719  -0.273102
1.0          0.181754   3.25403
2.0         -0.495068  -0.524877
3.0         -1.20696   -0.033936
4.0         -0.664662   2.20862
...
95.0         0.942969   0.180585
96.0         2.15161    0.661736
97.0         0.751956  -1.72922
98.0        -1.45054    1.52954
99.0         0.199145   0.582944
dtype: float64, shape: (100, 2)
```



```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)

In [35]: tsdtensor
Out[35]:
Time (s)
----------  ---------------------
0.0         [[-0.87 ... -0.87] ...]
1.0         [[1.14 ... 1.14] ...]
2.0         [[0.25 ... 0.25] ...]
3.0         [[1.29 ... 1.29] ...]
4.0         [[-0.91 ... -0.91] ...]
...
95.0        [[-2.01 ... -2.01] ...]
96.0        [[-0.08 ... -0.08] ...]
97.0        [[0.53 ... 0.53] ...]
98.0        [[-0.62 ... -0.62] ...]
99.0        [[-0.55 ... -0.55] ...]
dtype: float64, shape: (100, 15, 15)
```



```
In [41]: tsd.as_series()
```

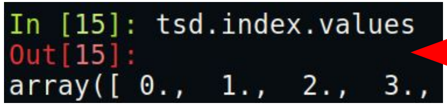


```
In [42]: tsdframe.as_dataframe()
```

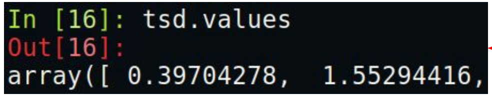
Doing math: the numpy array container approach

Numpy : NUMerical PYthon

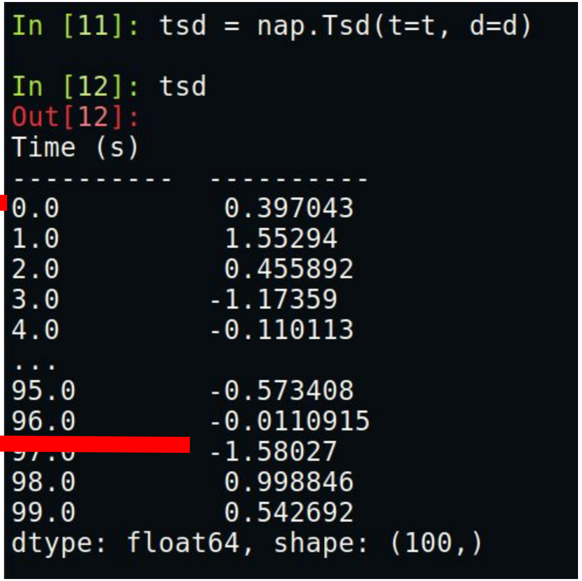
Numpy array



```
In [15]: tsd.index.values
Out[15]:
array([ 0.,   1.,   2.,   3.,
```



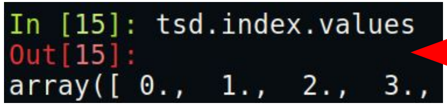
```
In [16]: tsd.values
Out[16]:
array([ 0.39704278,  1.55294416,
```



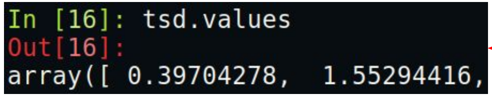
```
In [11]: tsd = nap.Tsd(t=t, d=d)

In [12]: tsd
Out[12]:
Time (s)
----------  ----------
0.0           0.397043
1.0           1.55294
2.0           0.455892
3.0          -1.17359
4.0          -0.110113
...
95.0         -0.573408
96.0         -0.0110915
97.0         -1.58027
98.0          0.998846
99.0          0.542692
dtype: float64, shape: (100,)
```

Numpy : NUMerical PYthon

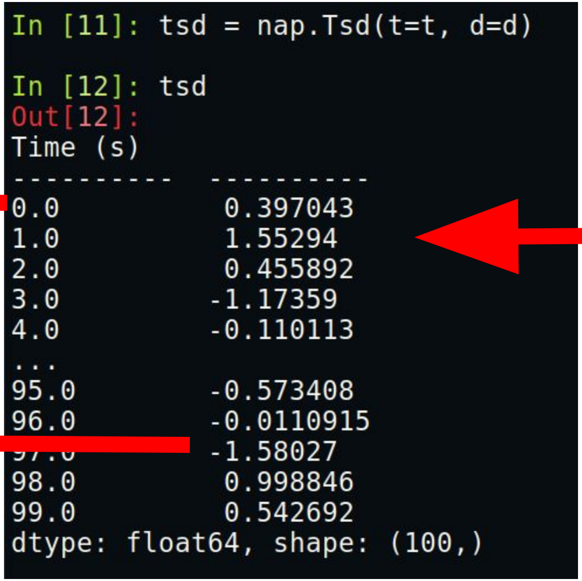
Numpy array



```
In [15]: tsd.index.values
Out[15]:
array([ 0.,   1.,   2.,   3.,
```



```
In [16]: tsd.values
Out[16]:
array([ 0.39704278,   1.55294416,
```



```
In [11]: tsd = nap.Tsd(t=t, d=d)

In [12]: tsd
Out[12]:
Time (s)
----------  ----------
0.0           0.397043
1.0           1.55294
2.0           0.455892
3.0          -1.17359
4.0          -0.110113
...
95.0         -0.573408
96.0         -0.0110915
97.0         -1.58027
98.0          0.998846
99.0          0.542692
dtype: float64, shape: (100,)
```

Numpy functions

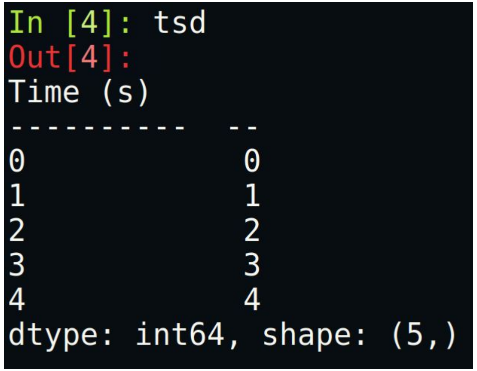
np.mean

np.add

np.min

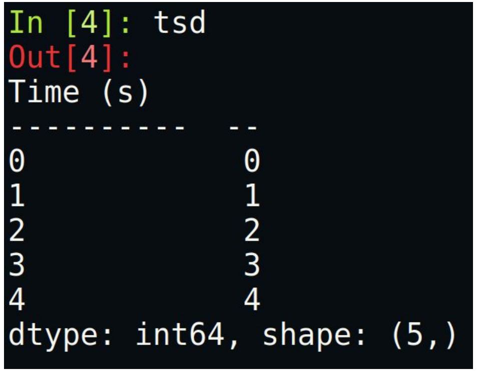
...

Arithmetical operations

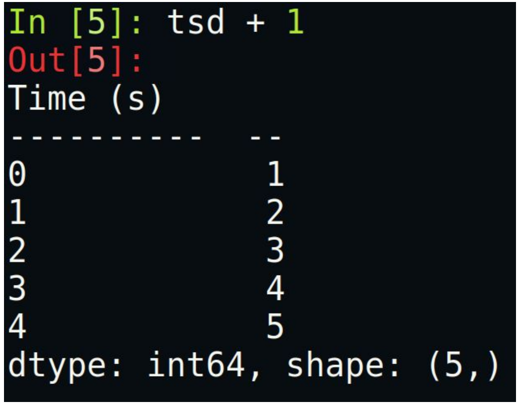


```
In [4]: tsd
Out[4]:
Time (s)
----------  --
0            0
1            1
2            2
3            3
4            4
dtype: int64, shape: (5,)
```

Arithmetical operations

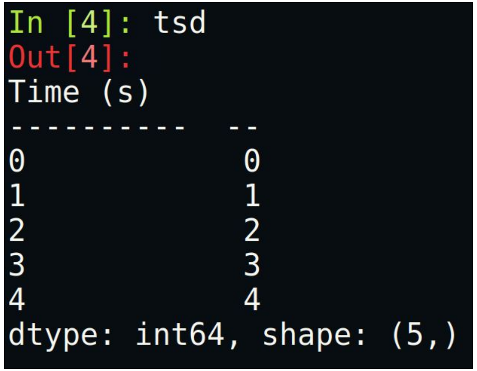


```
In [4]: tsd
Out[4]:
Time (s)
----------  --
0            0
1            1
2            2
3            3
4            4
dtype: int64, shape: (5,)
```

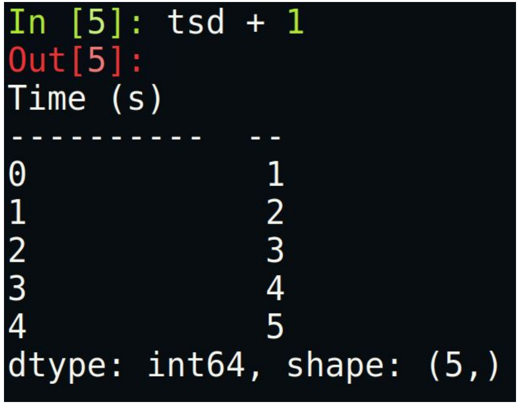


```
In [5]: tsd + 1
Out[5]:
Time (s)
----------  --
0            1
1            2
2            3
3            4
4            5
dtype: int64, shape: (5,)
```

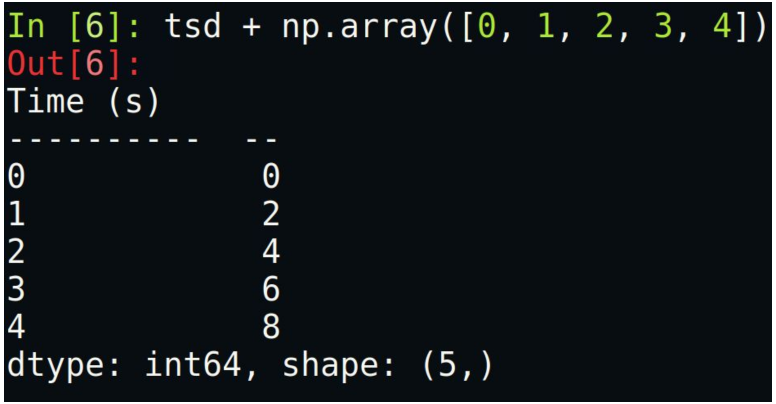
Arithmetical operations



```
In [4]: tsd
Out[4]:
Time (s)
----------  --
0            0
1            1
2            2
3            3
4            4
dtype: int64, shape: (5,)
```

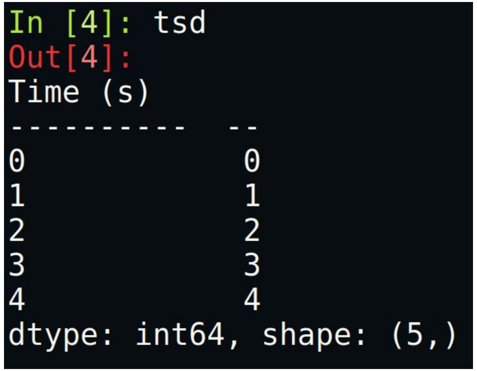


```
In [5]: tsd + 1
Out[5]:
Time (s)
----------  --
0            1
1            2
2            3
3            4
4            5
dtype: int64, shape: (5,)
```

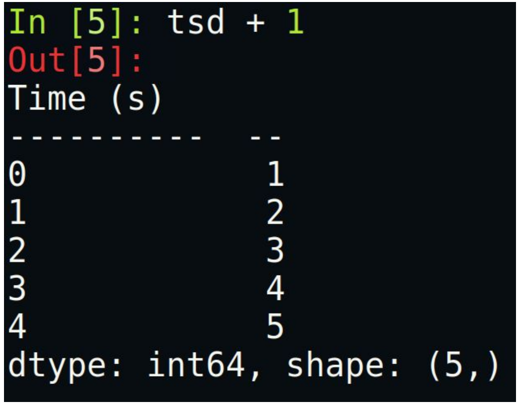


```
In [6]: tsd + np.array([0, 1, 2, 3, 4])
Out[6]:
Time (s)
----------  --
0            0
1            2
2            4
3            6
4            8
dtype: int64, shape: (5,)
```

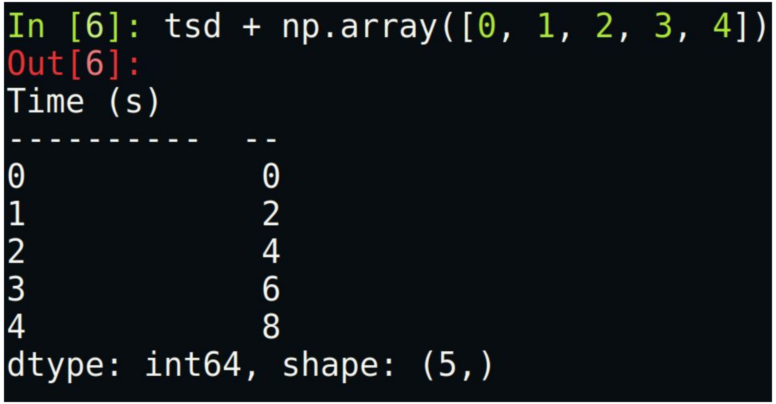
Arithmetical operations



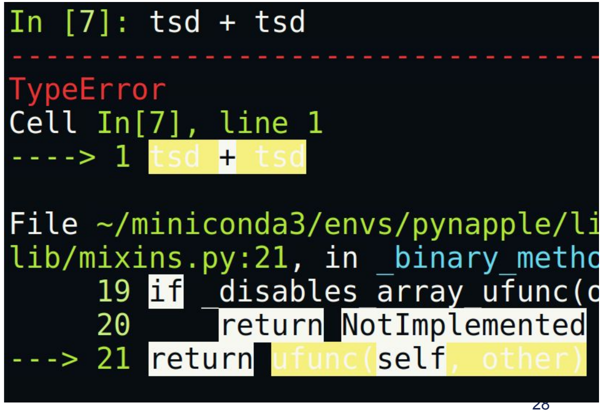
```
In [4]: tsd
Out[4]:
Time (s)
----------  --
0           0
1           1
2           2
3           3
4           4
dtype: int64, shape: (5,)
```



```
In [5]: tsd + 1
Out[5]:
Time (s)
----------  --
0           1
1           2
2           3
3           4
4           5
dtype: int64, shape: (5,)
```

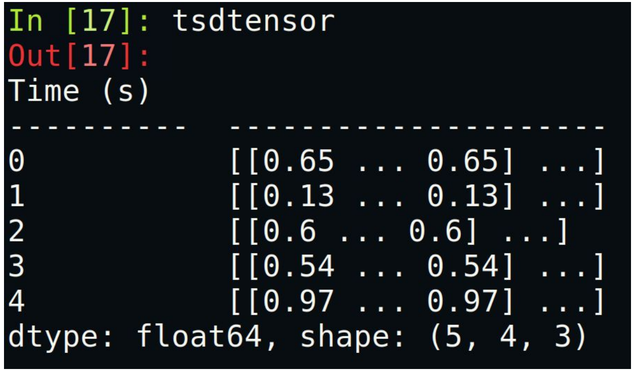


```
In [6]: tsd + np.array([0, 1, 2, 3, 4])
Out[6]:
Time (s)
----------  --
0           0
1           2
2           4
3           6
4           8
dtype: int64, shape: (5,)
```



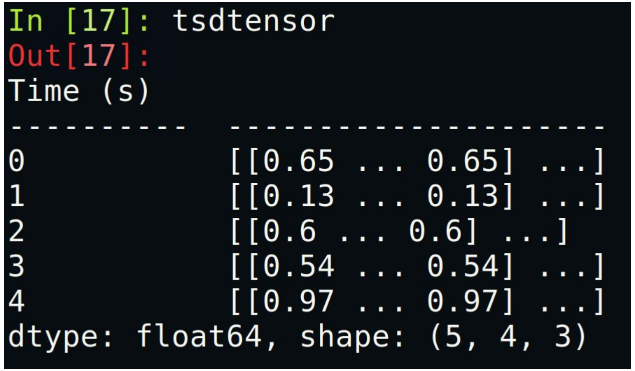
```
In [7]: tsd + tsd
-------------------------------------
TypeError
Cell In[7], line 1
----> 1 tsd + tsd

File ~/miniconda3/envs/pynapple/li
lib/mixins.py:21, in _binary_metho
     19 if _disables_array_ufunc(o
     20     return NotImplemented
---> 21 return ufunc(self, other)
```

Array operations



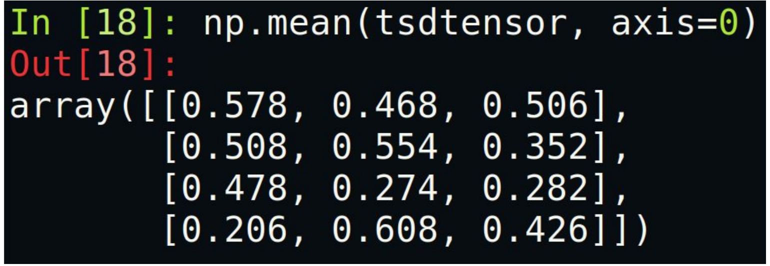
```
In [17]: tsdtensor
Out[17]:
Time (s)
----------  ---------------------
0           [[0.65 ... 0.65] ...]
1           [[0.13 ... 0.13] ...]
2           [[0.6 ... 0.6] ...]
3           [[0.54 ... 0.54] ...]
4           [[0.97 ... 0.97] ...]
dtype: float64, shape: (5, 4, 3)
```

Array operations



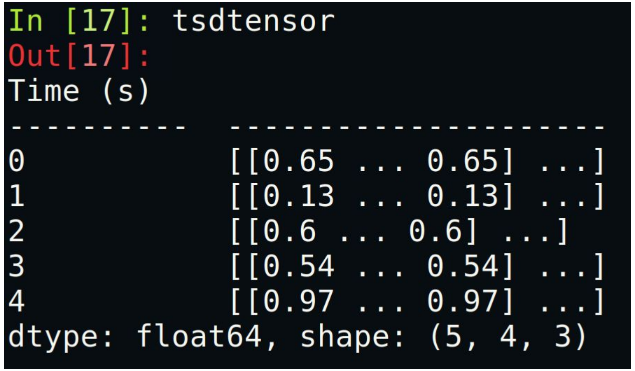
```
In [17]: tsdtensor
Out[17]:
Time (s)
----------  ---------------------
0           [[0.65 ... 0.65] ...]
1           [[0.13 ... 0.13] ...]
2           [[0.6 ... 0.6] ...]
3           [[0.54 ... 0.54] ...]
4           [[0.97 ... 0.97] ...]
dtype: float64, shape: (5, 4, 3)
```

numpy.ndarray



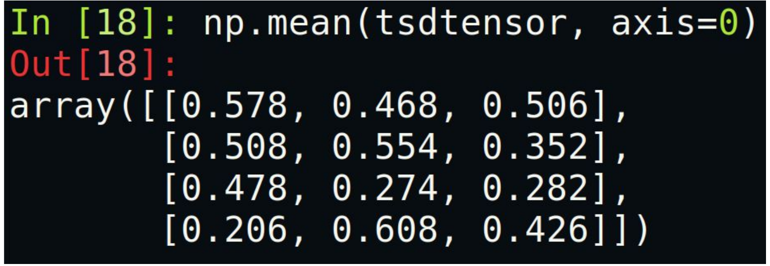
```
In [18]: np.mean(tsdtensor, axis=0)
Out[18]:
array([[0.578, 0.468, 0.506],
       [0.508, 0.554, 0.352],
       [0.478, 0.274, 0.282],
       [0.206, 0.608, 0.426]])
```

Array operations



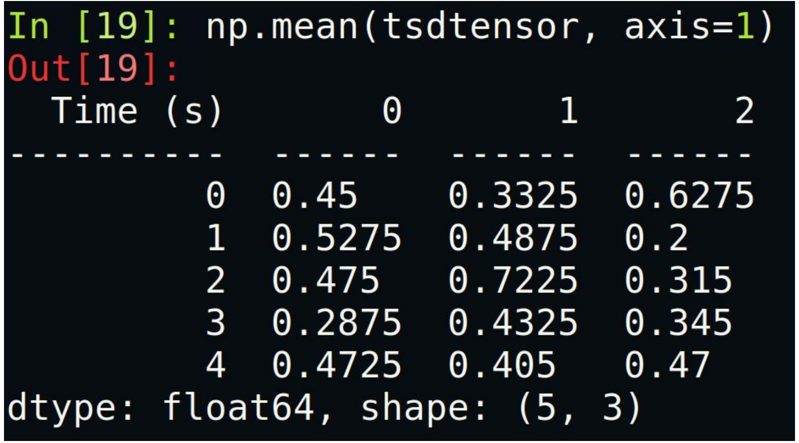
```
In [17]: tsdtensor
Out[17]:
Time (s)
----------  ---------------------
0           [[0.65 ... 0.65] ...]
1           [[0.13 ... 0.13] ...]
2           [[0.6 ... 0.6] ...]
3           [[0.54 ... 0.54] ...]
4           [[0.97 ... 0.97] ...]
dtype: float64, shape: (5, 4, 3)
```

numpy.ndarray



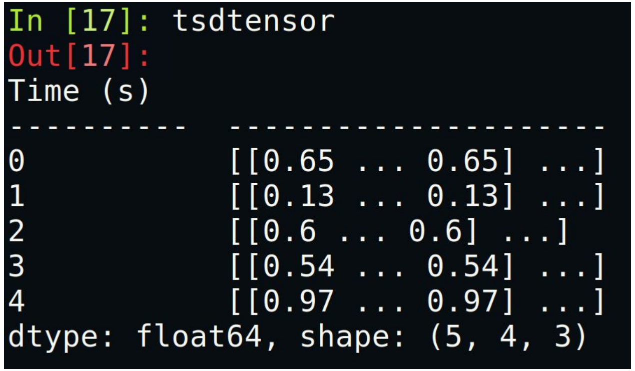
```
In [18]: np.mean(tsdtensor, axis=0)
Out[18]:
array([[0.578, 0.468, 0.506],
       [0.508, 0.554, 0.352],
       [0.478, 0.274, 0.282],
       [0.206, 0.608, 0.426]])
```

nap.TsdFrame



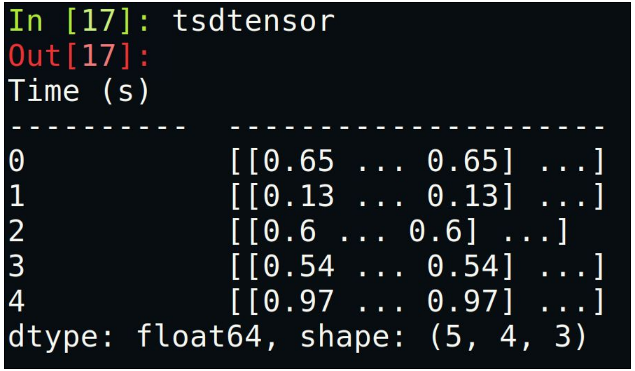
```
In [19]: np.mean(tsdtensor, axis=1)
Out[19]:
  Time (s)       0         1         2
----------  ------    ------    ------
         0  0.45      0.3325    0.6275
         1  0.5275    0.4875    0.2
         2  0.475     0.7225    0.315
         3  0.2875    0.4325    0.345
         4  0.4725    0.405     0.47
dtype: float64, shape: (5, 3)
```

Array slicing



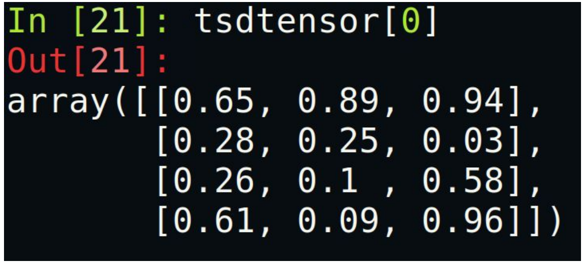
```
In [17]: tsdtensor
Out[17]:
Time (s)
----------  ---------------------
0           [[0.65 ... 0.65] ...]
1           [[0.13 ... 0.13] ...]
2           [[0.6 ... 0.6] ...]
3           [[0.54 ... 0.54] ...]
4           [[0.97 ... 0.97] ...]
dtype: float64, shape: (5, 4, 3)
```

Array slicing



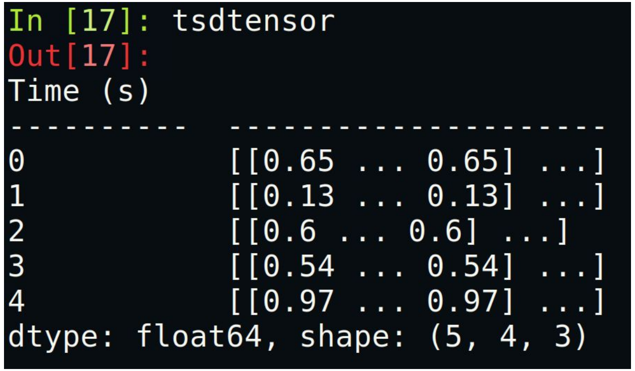
```
In [17]: tsdtensor
Out[17]:
Time (s)
----------  ---------------------
0           [[0.65 ... 0.65] ...]
1           [[0.13 ... 0.13] ...]
2           [[0.6 ... 0.6] ...]
3           [[0.54 ... 0.54] ...]
4           [[0.97 ... 0.97] ...]
dtype: float64, shape: (5, 4, 3)
```

numpy.ndarray



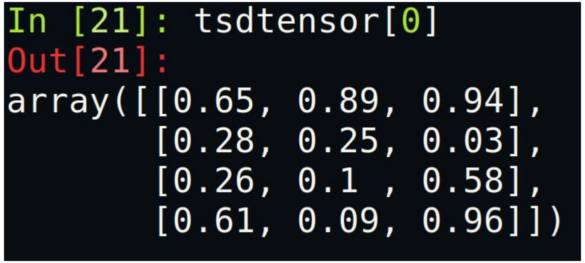
```
In [21]: tsdtensor[0]
Out[21]:
array([[0.65, 0.89, 0.94],
       [0.28, 0.25, 0.03],
       [0.26, 0.1 , 0.58],
       [0.61, 0.09, 0.96]])
```

Array slicing



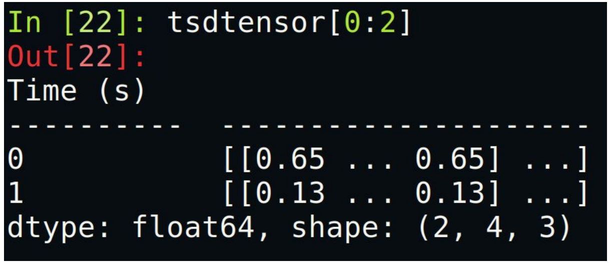
```
In [17]: tsdtensor
Out[17]:
Time (s)
----------  ---------------------
0           [[0.65 ... 0.65] ...]
1           [[0.13 ... 0.13] ...]
2           [[0.6 ... 0.6] ...]
3           [[0.54 ... 0.54] ...]
4           [[0.97 ... 0.97] ...]
dtype: float64, shape: (5, 4, 3)
```

numpy.ndarray



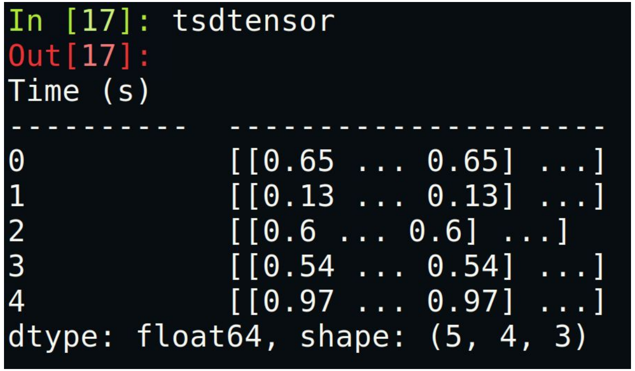
```
In [21]: tsdtensor[0]
Out[21]:
array([[0.65, 0.89, 0.94],
       [0.28, 0.25, 0.03],
       [0.26, 0.1 , 0.58],
       [0.61, 0.09, 0.96]])
```

nap.TsdTensor



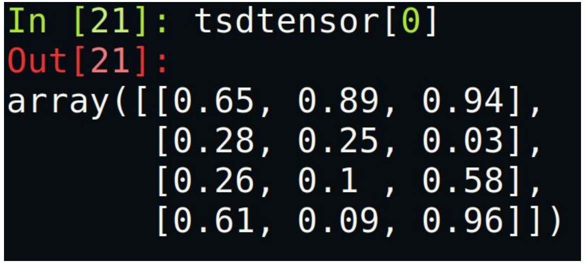
```
In [22]: tsdtensor[0:2]
Out[22]:
Time (s)
----------  ---------------------
0           [[0.65 ... 0.65] ...]
1           [[0.13 ... 0.13] ...]
dtype: float64, shape: (2, 4, 3)
```

Array slicing



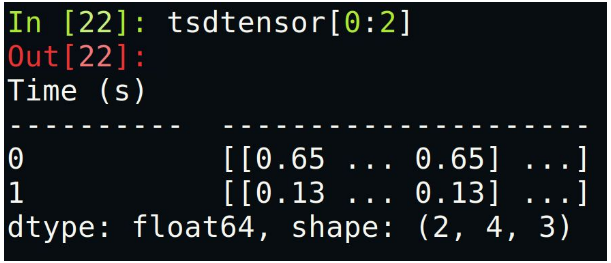
```
In [17]: tsdtensor
Out[17]:
Time (s)
----------  ---------------------
0           [[0.65 ... 0.65] ...]
1           [[0.13 ... 0.13] ...]
2           [[0.6 ... 0.6] ...]
3           [[0.54 ... 0.54] ...]
4           [[0.97 ... 0.97] ...]
dtype: float64, shape: (5, 4, 3)
```

numpy.ndarray



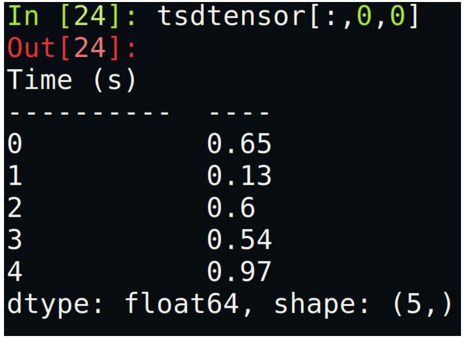
```
In [21]: tsdtensor[0]
Out[21]:
array([[0.65, 0.89, 0.94],
       [0.28, 0.25, 0.03],
       [0.26, 0.1 , 0.58],
       [0.61, 0.09, 0.96]])
```

nap.TsdTensor



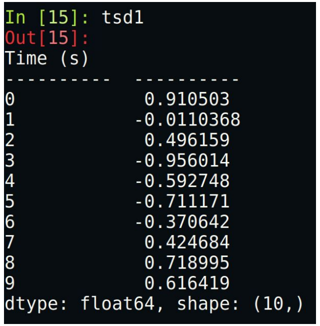
```
In [22]: tsdtensor[0:2]
Out[22]:
Time (s)
----------  ---------------------
0           [[0.65 ... 0.65] ...]
1           [[0.13 ... 0.13] ...]
dtype: float64, shape: (2, 4, 3)
```

nap.Tsd



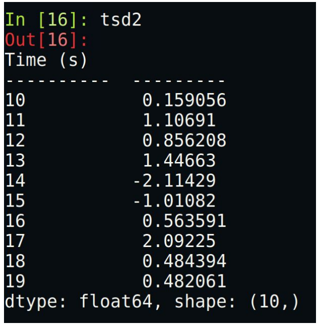
```
In [24]: tsdtensor[:,0,0]
Out[24]:
Time (s)
----------  ----
0           0.65
1           0.13
2           0.6
3           0.54
4           0.97
dtype: float64, shape: (5,)
```

Array concatenation



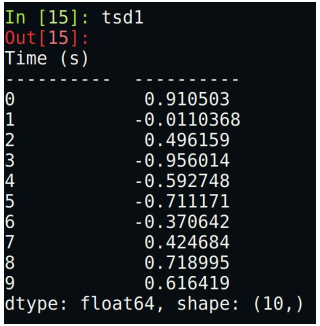
```
In [15]: tsd1
Out[15]:
Time (s)
----------  ----------
0             0.910503
1            -0.0110368
2             0.496159
3            -0.956014
4            -0.592748
5            -0.711171
6            -0.370642
7             0.424684
8             0.718995
9             0.616419
dtype: float64, shape: (10,)
```

+



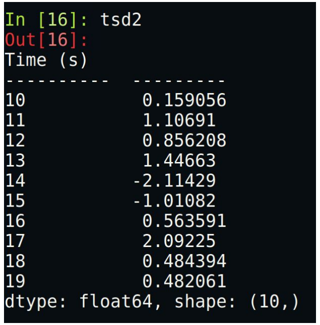
```
In [16]: tsd2
Out[16]:
Time (s)
----------  ---------
10            0.159056
11            1.10691
12            0.856208
13            1.44663
14           -2.11429
15           -1.01082
16            0.563591
17            2.09225
18            0.484394
19            0.482061
dtype: float64, shape: (10,)
```

Array concatenation

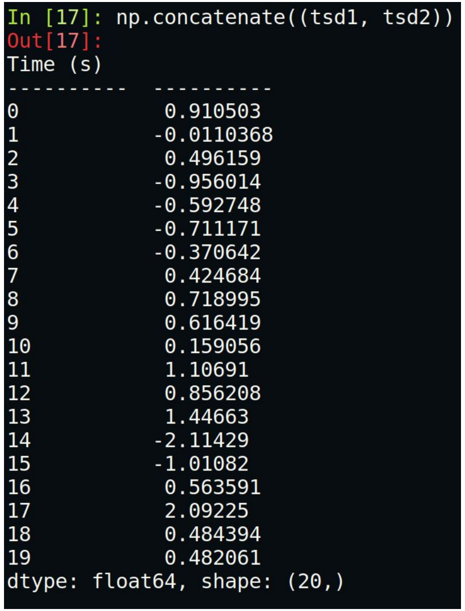


```
In [15]: tsd1
Out[15]:
Time (s)
----------  ----------
0             0.910503
1            -0.0110368
2             0.496159
3            -0.956014
4            -0.592748
5            -0.711171
6            -0.370642
7             0.424684
8             0.718995
9             0.616419
dtype: float64, shape: (10,)
```

+

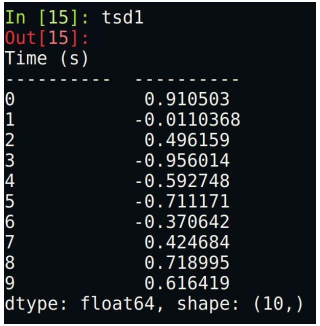


```
In [16]: tsd2
Out[16]:
Time (s)
----------  ---------
10            0.159056
11            1.10691
12            0.856208
13            1.44663
14           -2.11429
15           -1.01082
16            0.563591
17            2.09225
18            0.484394
19            0.482061
dtype: float64, shape: (10,)
```



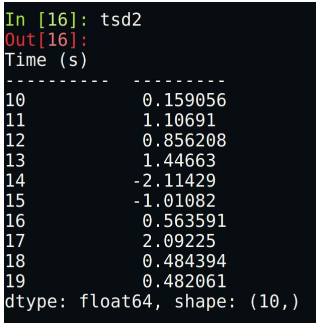
```
In [17]: np.concatenate((tsd1, tsd2))
Out[17]:
Time (s)
----------  ----------
0             0.910503
1            -0.0110368
2             0.496159
3            -0.956014
4            -0.592748
5            -0.711171
6            -0.370642
7             0.424684
8             0.718995
9             0.616419
10            0.159056
11            1.10691
12            0.856208
13            1.44663
14           -2.11429
15           -1.01082
16            0.563591
17            2.09225
18            0.484394
19            0.482061
dtype: float64, shape: (20,)
```

Array concatenation

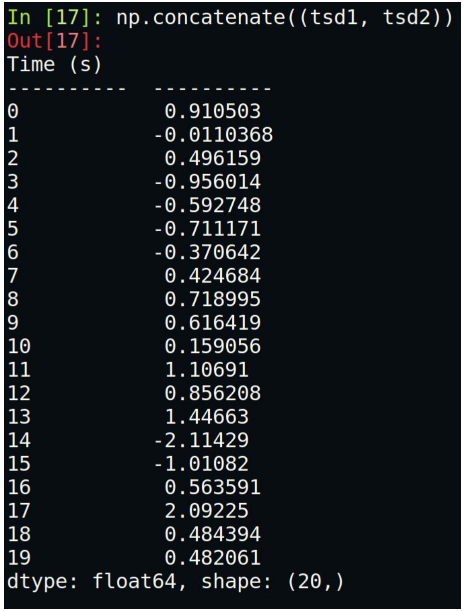


```
In [15]: tsd1
Out[15]:
Time (s)
----------  ----------
0             0.910503
1            -0.0110368
2             0.496159
3            -0.956014
4            -0.592748
5            -0.711171
6            -0.370642
7             0.424684
8             0.718995
9             0.616419
dtype: float64, shape: (10,)
```

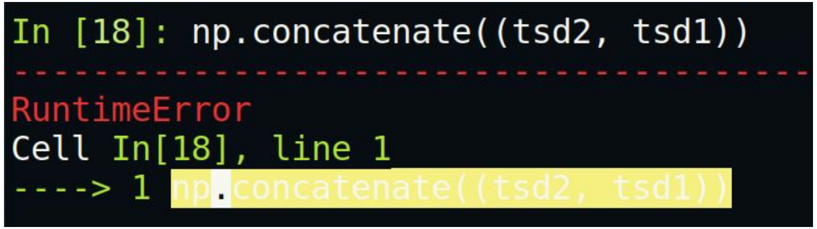
+



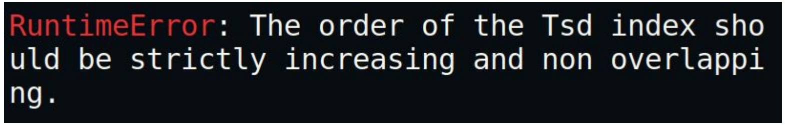
```
In [16]: tsd2
Out[16]:
Time (s)
----------  ---------
10            0.159056
11            1.10691
12            0.856208
13            1.44663
14           -2.11429
15           -1.01082
16            0.563591
17            2.09225
18            0.484394
19            0.482061
dtype: float64, shape: (10,)
```



```
In [17]: np.concatenate((tsd1, tsd2))
Out[17]:
Time (s)
----------  ----------
0             0.910503
1            -0.0110368
2             0.496159
3            -0.956014
4            -0.592748
5            -0.711171
6            -0.370642
7             0.424684
8             0.718995
9             0.616419
10            0.159056
11            1.10691
12            0.856208
13            1.44663
14           -2.11429
15           -1.01082
16            0.563591
17            2.09225
18            0.484394
19            0.482061
dtype: float64, shape: (20,)
```



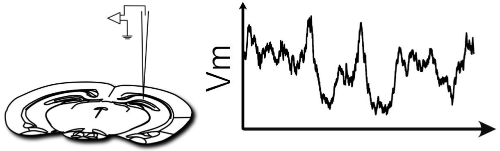
```
In [18]: np.concatenate((tsd2, tsd1))
---------------------------------------------
RuntimeError
Cell In[18], line 1
----> 1 np.concatenate((tsd2, tsd1))
```



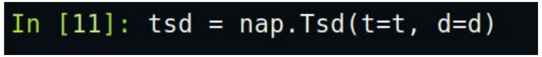
```
RuntimeError: The order of the Tsd index sho
uld be strictly increasing and non overlappi
ng.
```

Time series without data : the timestamps object

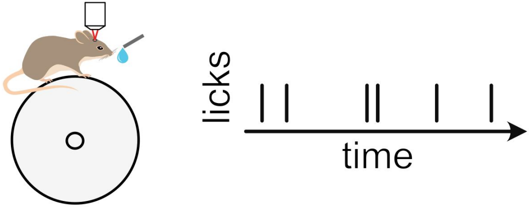
FLATIRON INSTITUTE

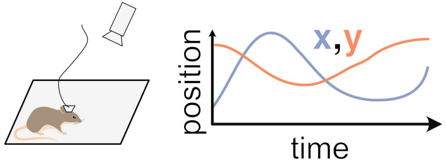


Tsd: 1-dimension

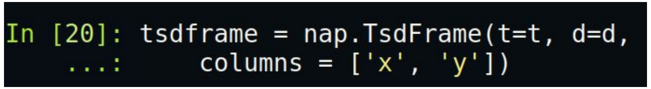


```
In [11]: tsd = nap.Tsd(t=t, d=d)
```

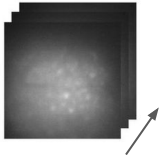




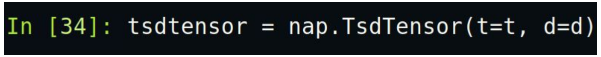
TsdFrame: 2-dimensions



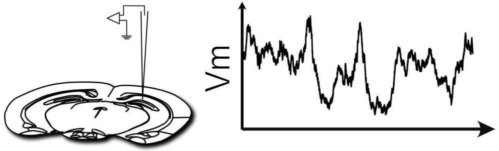
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:         columns = ['x', 'y'])
```



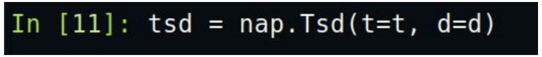
TsdTensor: n-dimensions



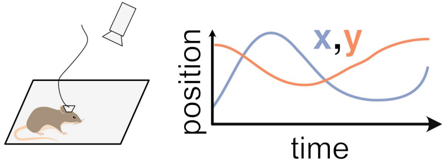
```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)
```



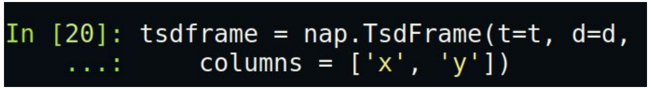
Tsd: 1-dimension



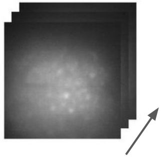
```
In [11]: tsd = nap.Tsd(t=t, d=d)
```



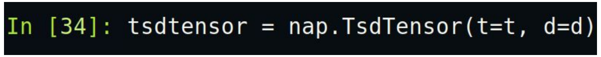
TsdFrame: 2-dimensions



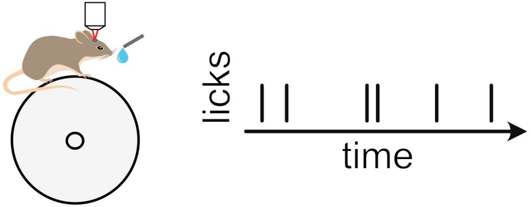
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:        columns = ['x', 'y'])
```



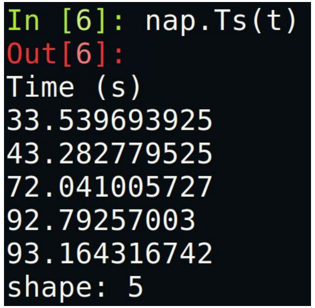
TsdTensor: n-dimensions



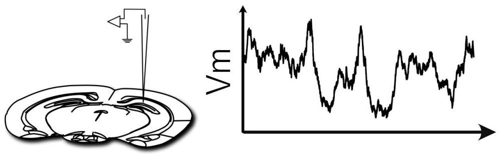
```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)
```

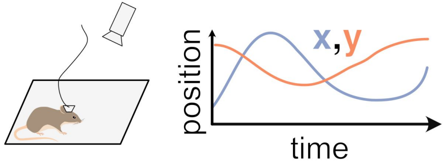


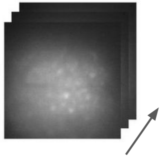
Ts: Timestamps



```
In [6]: nap.Ts(t)
Out[6]:
Time (s)
33.539693925
43.282779525
72.041005727
92.79257003
93.164316742
shape: 5
```

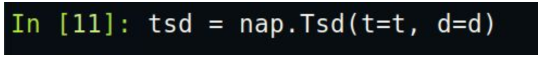






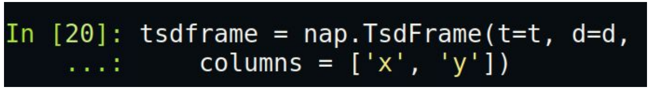
FLATIRON INSTITUTE

Tsd: 1-dimension



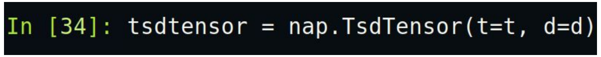
```
In [11]: tsd = nap.Tsd(t=t, d=d)
```

TsdFrame: 2-dimensions

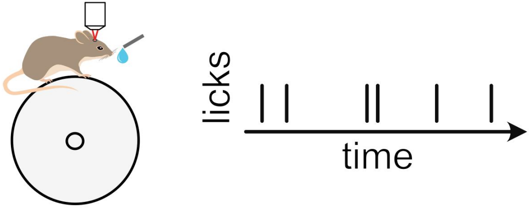


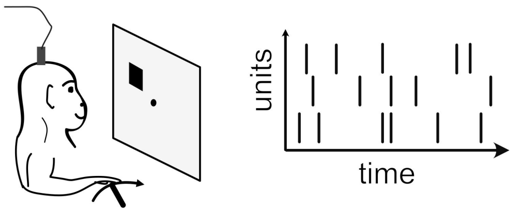
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:        columns = ['x', 'y'])
```

TsdTensor: n-dimensions

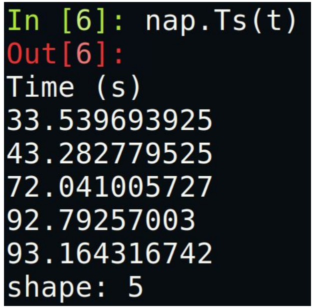


```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)
```

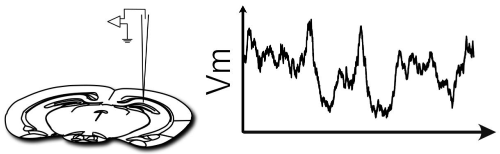




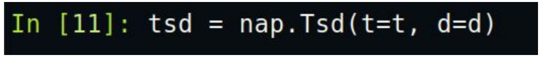
Ts: Timestamps



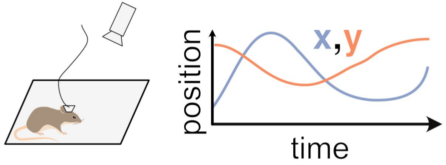
```
In [6]: nap.Ts(t)
Out[6]:
Time (s)
33.539693925
43.282779525
72.041005727
92.79257003
93.164316742
shape: 5
```



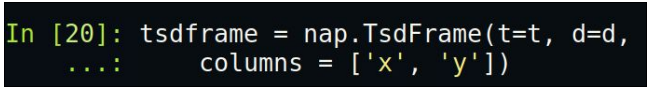
Tsd: 1-dimension



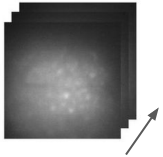
```
In [11]: tsd = nap.Tsd(t=t, d=d)
```



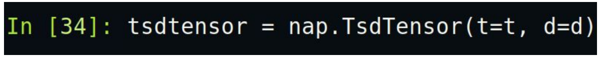
TsdFrame: 2-dimensions



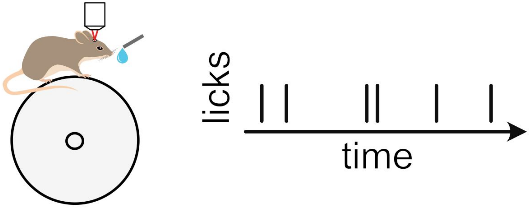
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:        columns = ['x', 'y'])
```



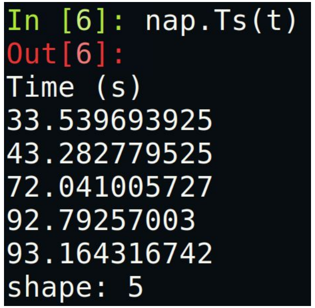
TsdTensor: n-dimensions



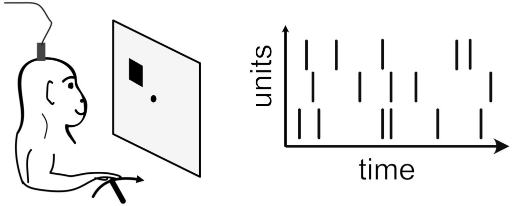
```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)
```



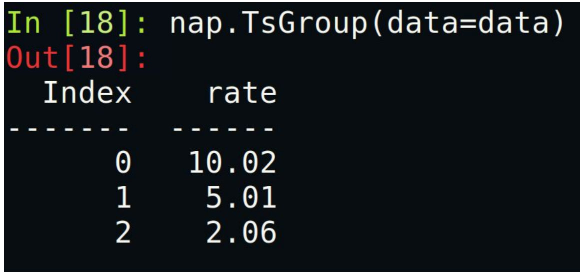
Ts: Timestamps



```
In [6]: nap.Ts(t)
Out[6]:
Time (s)
33.539693925
43.282779525
72.041005727
92.79257003
93.164316742
shape: 5
```



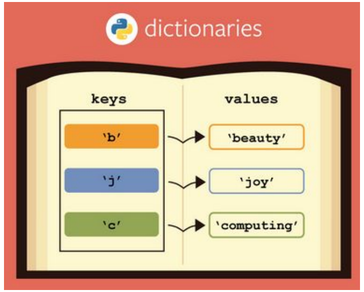
TsGroup: group of timestamps

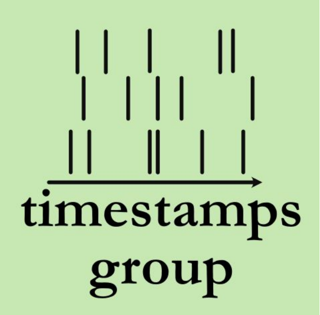


```
In [18]: nap.TsGroup(data=data)
Out[18]:
  Index     rate
-------   ------
      0    10.02
      1     5.01
      2     2.06
```

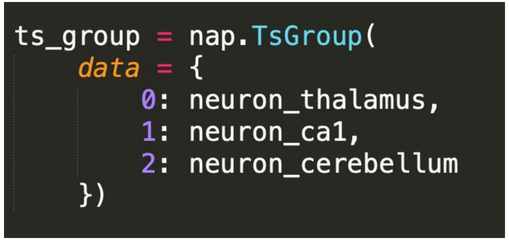
Population analysis made easier: the TsGroup object

TsGroup manipulation

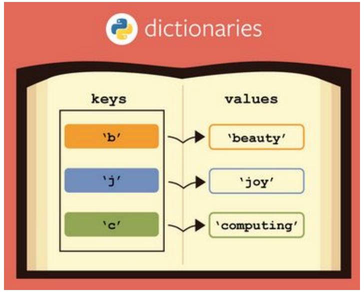


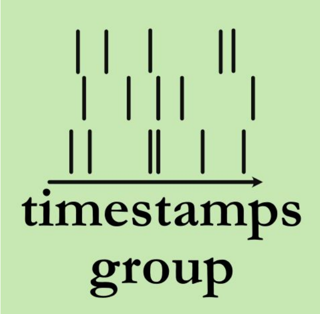


TsGroup manipulation

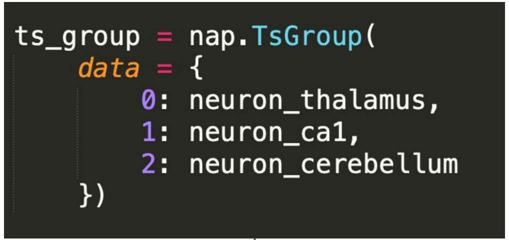


```
ts_group = nap.TsGroup(
    data = {
        0: neuron_thalamus,
        1: neuron_ca1,
        2: neuron_cerebellum
    })
```

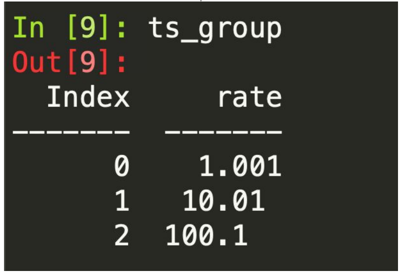




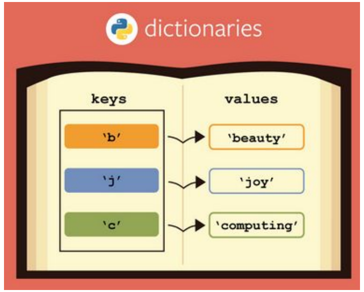
TsGroup manipulation

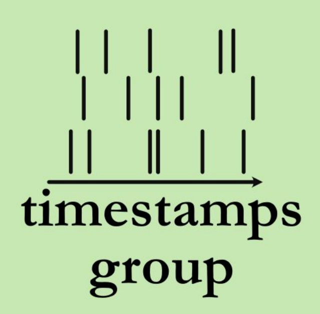


```
ts_group = nap.TsGroup(
    data = {
        0: neuron_thalamus,
        1: neuron_ca1,
        2: neuron_cerebellum
    })
```

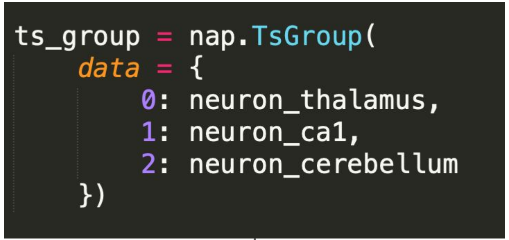


```
In [9]: ts_group
Out[9]:
  Index      rate
-------  -------
      0    1.001
      1   10.01
      2  100.1
```

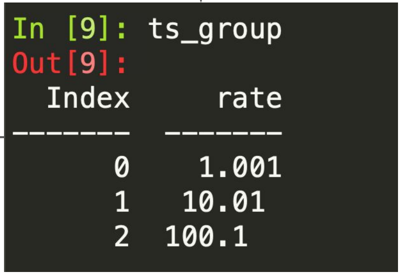




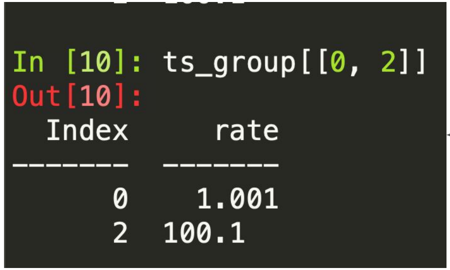
TsGroup manipulation



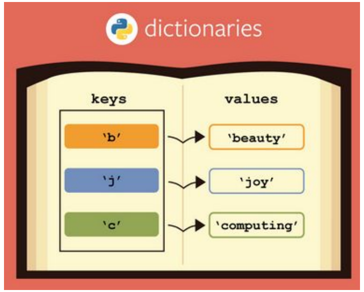
```
ts_group = nap.TsGroup(
    data = {
        0: neuron_thalamus,
        1: neuron_ca1,
        2: neuron_cerebellum
    })
```

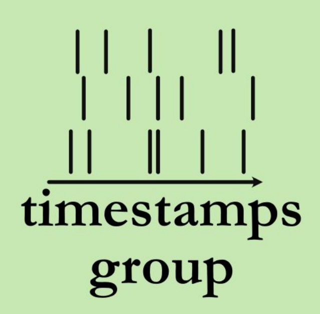


```
In [9]: ts_group
Out[9]:
  Index      rate
-------  -------
      0    1.001
      1   10.01
      2  100.1
```



```
In [10]: ts_group[[0, 2]]
Out[10]:
  Index      rate
-------  -------
      0    1.001
      2  100.1
```

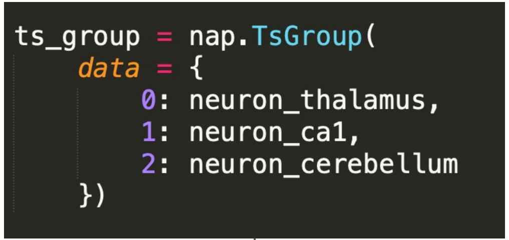




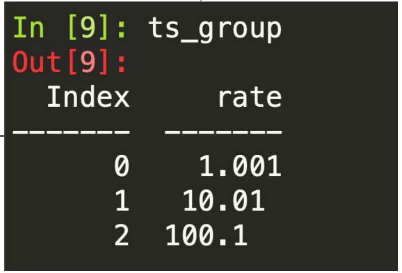
TsGroup manipulation

Operations :

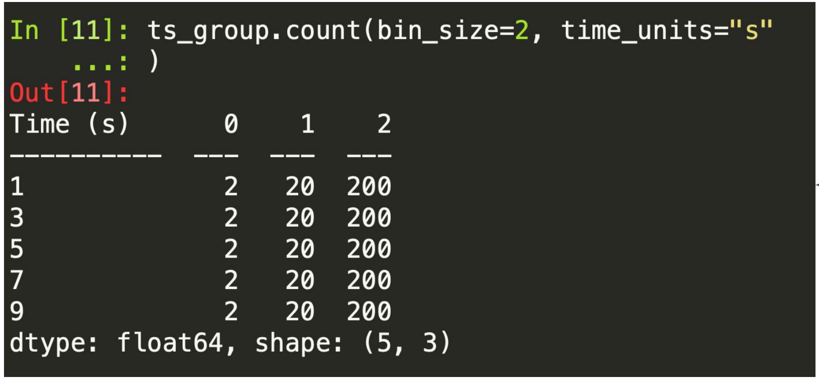
- restrict
- Binning
- …



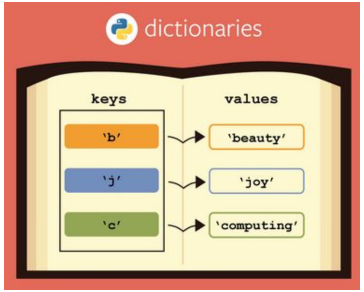
```
ts_group = nap.TsGroup(
    data = {
        0: neuron_thalamus,
        1: neuron_ca1,
        2: neuron_cerebellum
    })
```

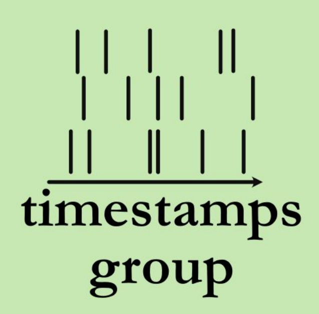


```
In [9]: ts_group
Out[9]:
  Index       rate
--------  --------
       0     1.001
       1    10.01
       2   100.1
```

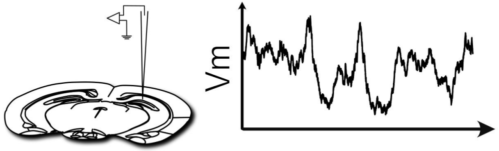


```
In [11]: ts_group.count(bin_size=2, time_units="s"
    ...: )
Out[11]:
Time (s)      0    1    2
----------  ---  ---  ---
1             2   20  200
3             2   20  200
5             2   20  200
7             2   20  200
9             2   20  200
dtype: float64, shape: (5, 3)
```

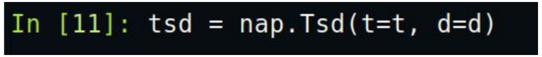




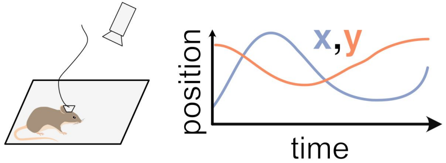
FLATIRON INSTITUTE



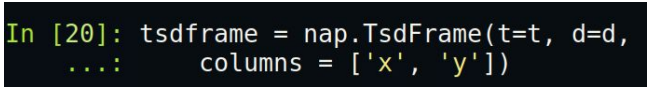
Tsd: 1-dimension



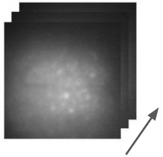
```
In [11]: tsd = nap.Tsd(t=t, d=d)
```



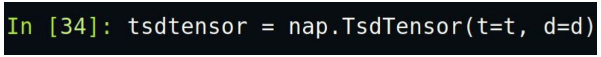
TsdFrame: 2-dimensions



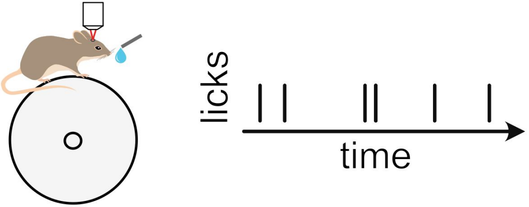
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:      columns = ['x', 'y'])
```



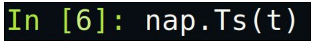
TsdTensor: n-dimensions



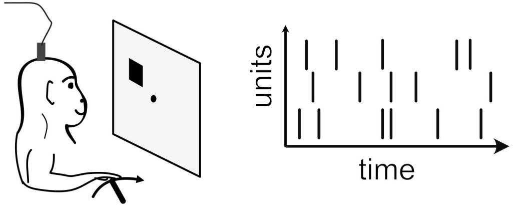
```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)
```



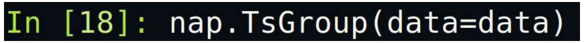
Ts: Timestamps



```
In [6]: nap.Ts(t)
```

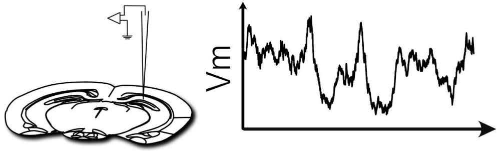


TsGroup: group of timestamps

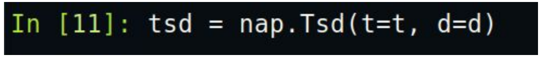


```
In [18]: nap.TsGroup(data=data)
```

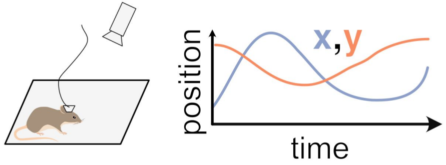
FLATIRON INSTITUTE



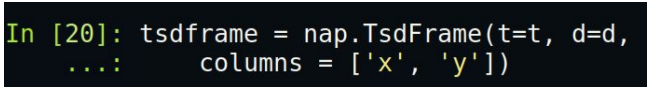
Tsd: 1-dimension



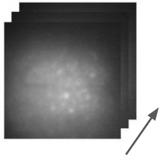
```
In [11]: tsd = nap.Tsd(t=t, d=d)
```



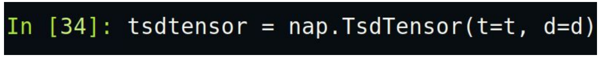
TsdFrame: 2-dimensions



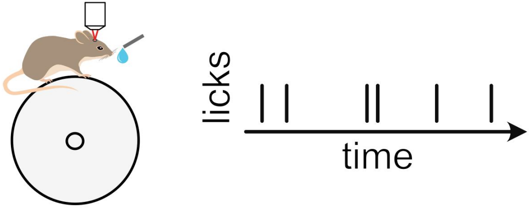
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:        columns = ['x', 'y'])
```



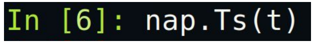
TsdTensor: n-dimensions



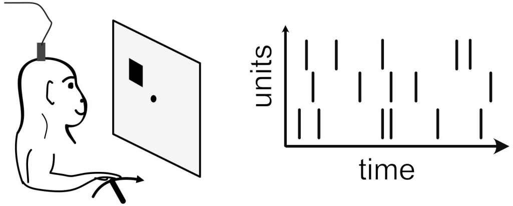
```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)
```



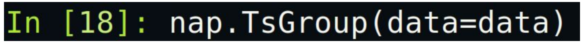
Ts: Timestamps



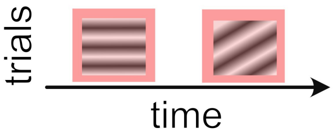
```
In [6]: nap.Ts(t)
```



TsGroup: group of timestamps

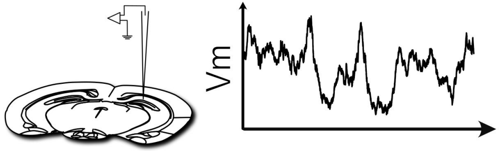


```
In [18]: nap.TsGroup(data=data)
```

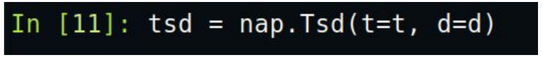


IntervalSet: set of epochs

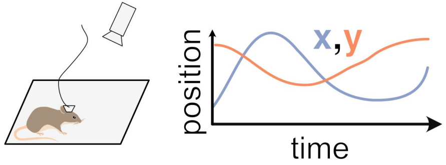
FLATIRON INSTITUTE



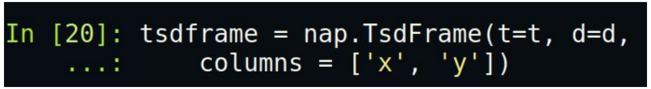
Tsd: 1-dimension



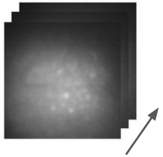
```
In [11]: tsd = nap.Tsd(t=t, d=d)
```



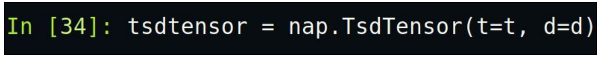
TsdFrame: 2-dimensions



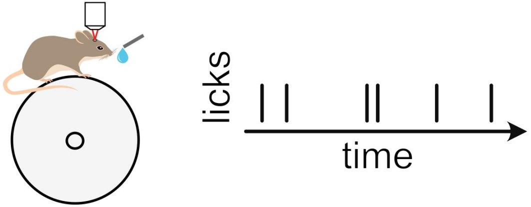
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:       columns = ['x', 'y'])
```



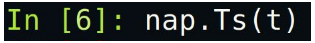
TsdTensor: n-dimensions



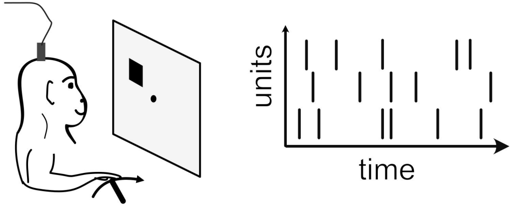
```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)
```



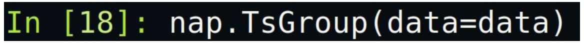
Ts: Timestamps



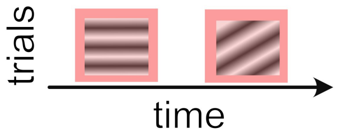
```
In [6]: nap.Ts(t)
```



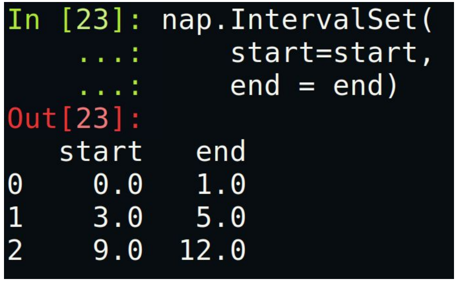
TsGroup: group of timestamps



```
In [18]: nap.TsGroup(data=data)
```



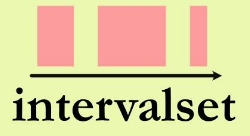
IntervalSet: set of epochs



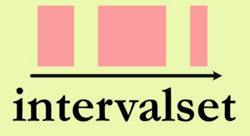
```
In [23]: nap.IntervalSet(
    ...:     start=start,
    ...:     end = end)
Out[23]:
   start   end
0    0.0   1.0
1    3.0   5.0
2    9.0  12.0
```

Manipulating time : the IntervalSet object

- Sleep/wake
- Stimulus on/off
- Lick start/end

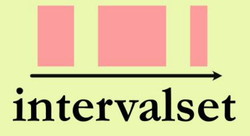


- Sleep/wake
- Stimulus on/off
- Lick start/end

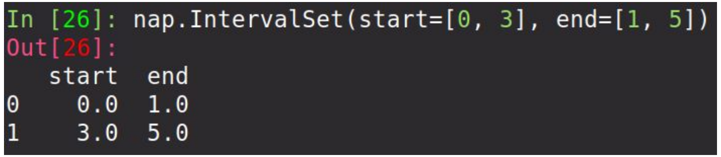


	Start (second)	End (second)
Stim 0	0	1
Stim 1	3	5

- Sleep/wake
- Stimulus on/off
- Lick start/end

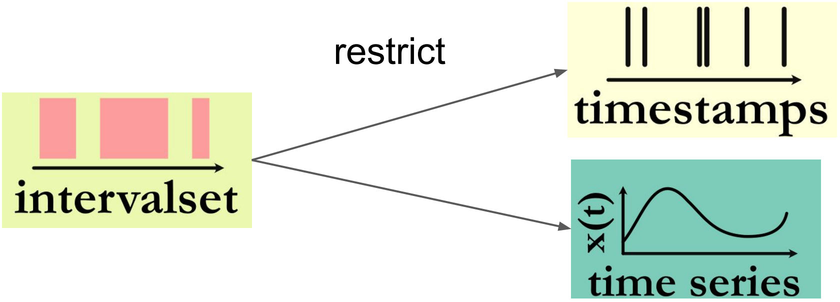


	Start (second)	End (second)
Stim 0	0	1
Stim 1	3	5

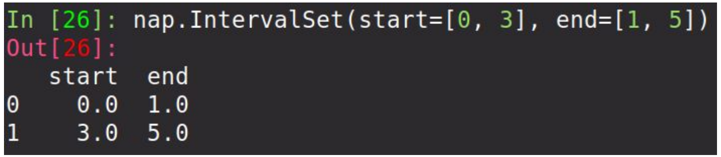


```
In [26]: nap.IntervalSet(start=[0, 3], end=[1, 5])
Out[26]:
   start  end
0    0.0  1.0
1    3.0  5.0
```

- Sleep/wake
- Stimulus on/off
- Lick start/end

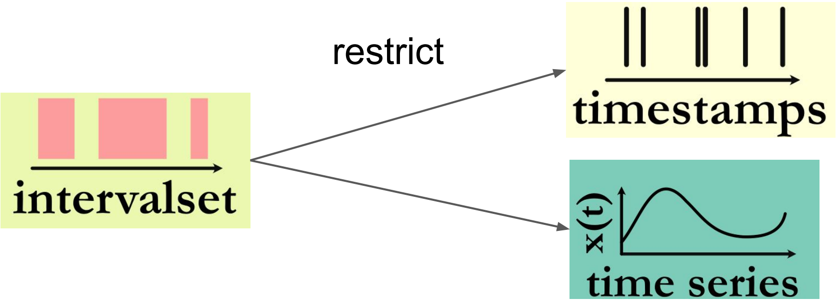


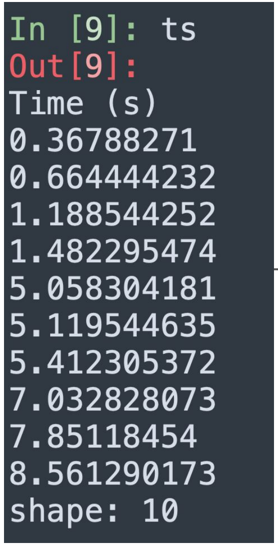
	Start (second)	End (second)
Stim 0	0	1
Stim 1	3	5



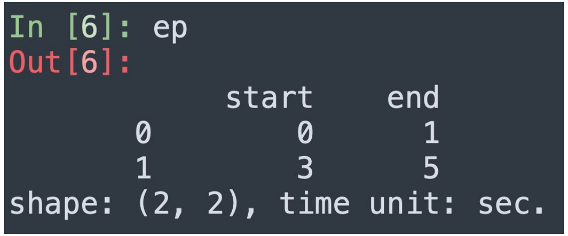
```
In [26]: nap.IntervalSet(start=[0, 3], end=[1, 5])
Out[26]:
   start  end
0    0.0  1.0
1    3.0  5.0
```

- Sleep/wake
- Stimulus on/off
- Lick start/end

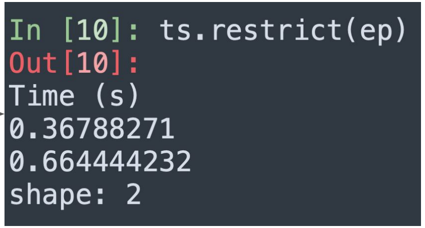




```
In [9]: ts
Out[9]:
Time (s)
0.36788271
0.664444232
1.188544252
1.482295474
5.058304181
5.119544635
5.412305372
7.032828073
7.85118454
8.561290173
shape: 10
```

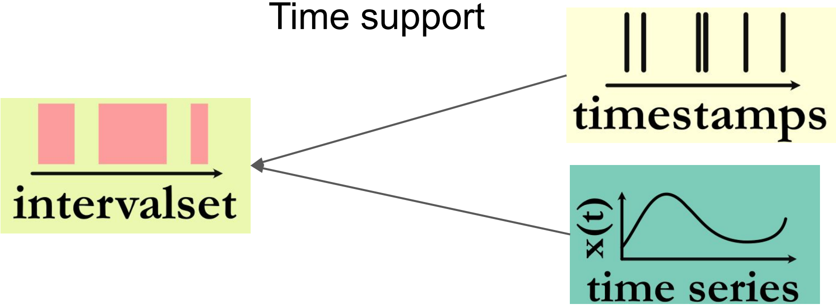


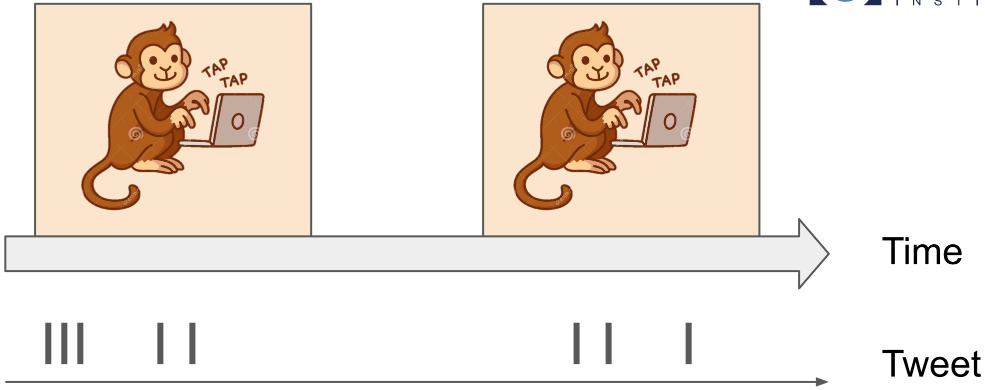
```
In [6]: ep
Out[6]:
            start     end
       0        0       1
       1        3       5
shape: (2, 2), time unit: sec.
```



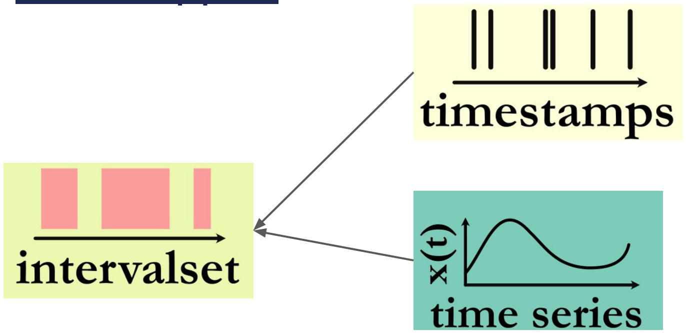
```
In [10]: ts.restrict(ep)
Out[10]:
Time (s)
0.36788271
0.664444232
shape: 2
```

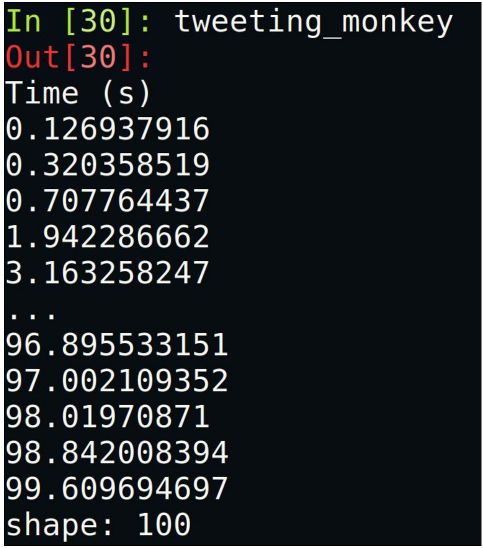
Time support



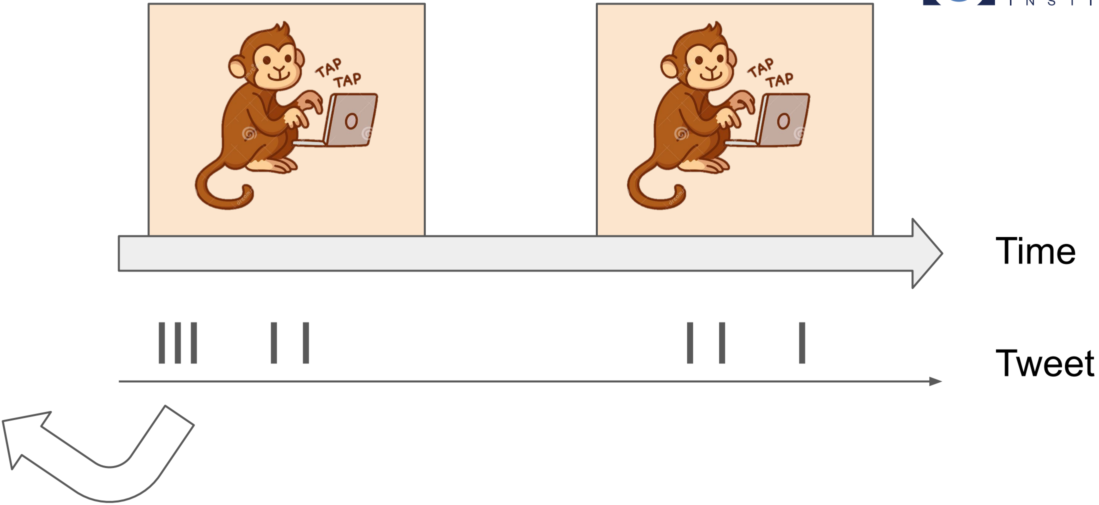


Time support

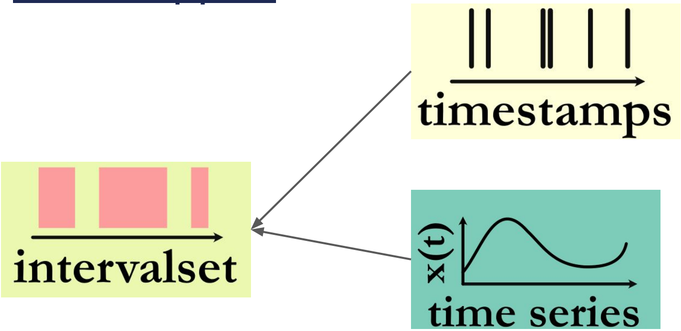




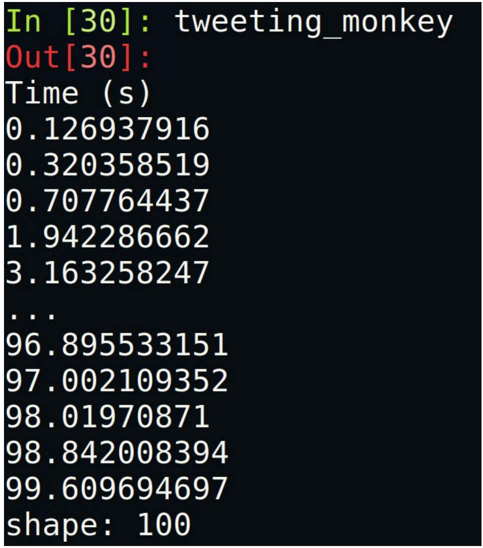
```
In [30]: tweeting_monkey
Out[30]:
Time (s)
0.126937916
0.320358519
0.707764437
1.942286662
3.163258247
...
96.895533151
97.002109352
98.01970871
98.842008394
99.609694697
shape: 100
```



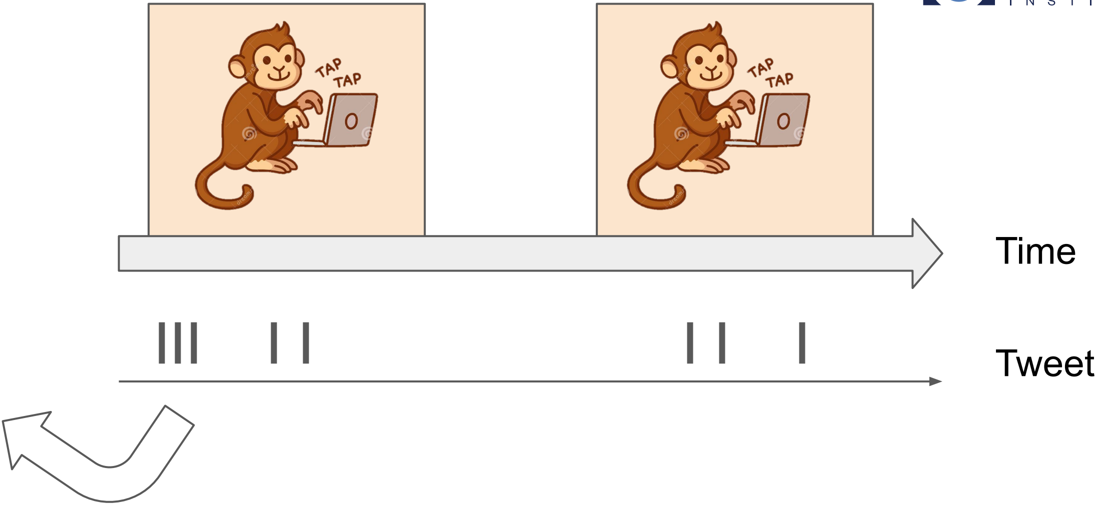
Time support



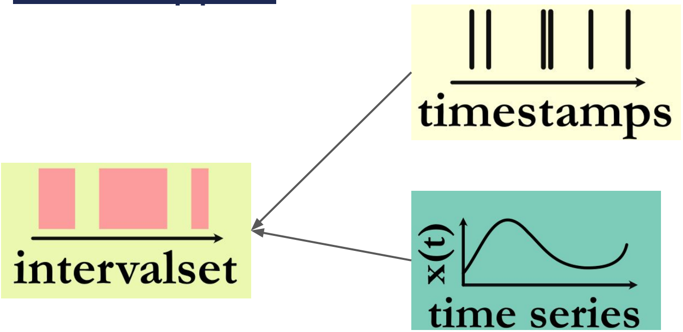
Tweet frequency?



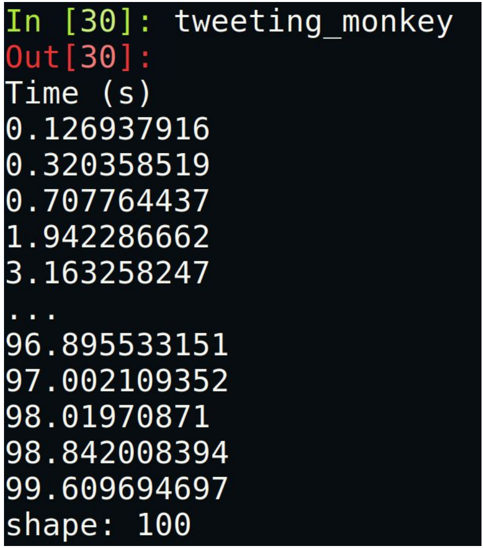
```
In [30]: tweeting_monkey
Out[30]:
Time (s)
0.126937916
0.320358519
0.707764437
1.942286662
3.163258247
...
96.895533151
97.002109352
98.01970871
98.842008394
99.609694697
shape: 100
```



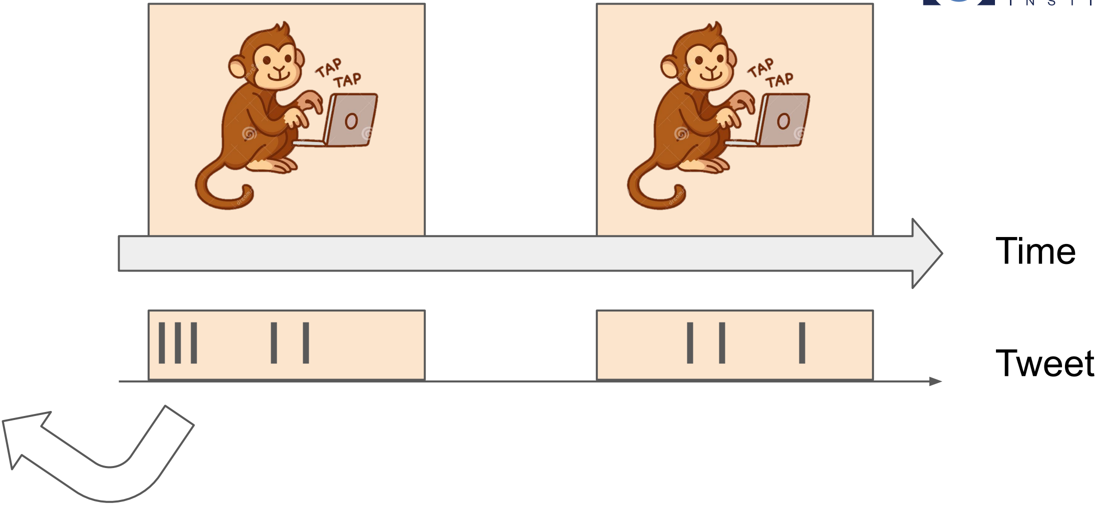
Time support



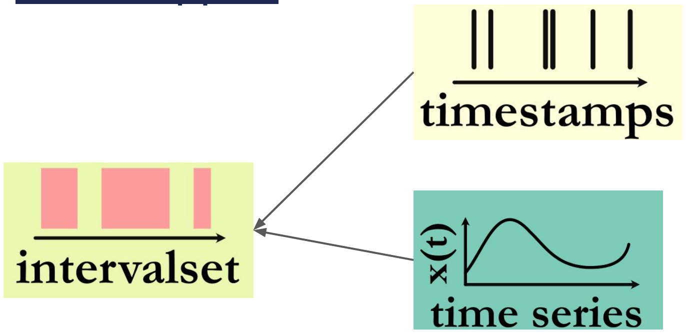
Tweet frequency?



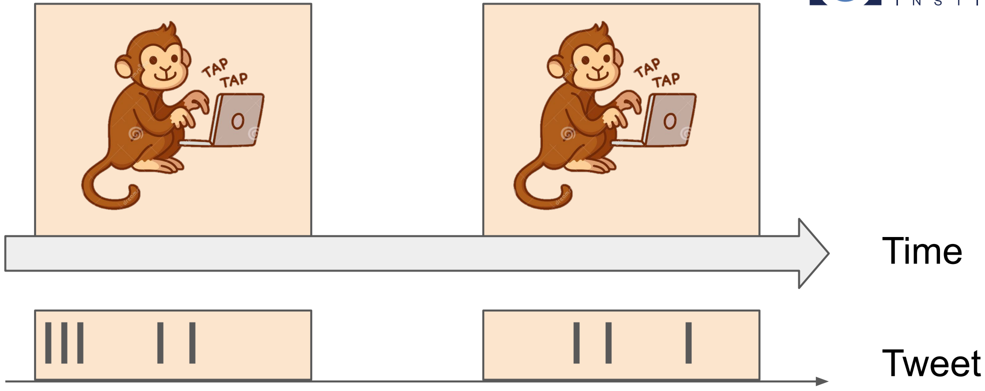
```
In [30]: tweeting_monkey
Out[30]:
Time (s)
0.126937916
0.320358519
0.707764437
1.942286662
3.163258247
...
96.895533151
97.002109352
98.01970871
98.842008394
99.609694697
shape: 100
```

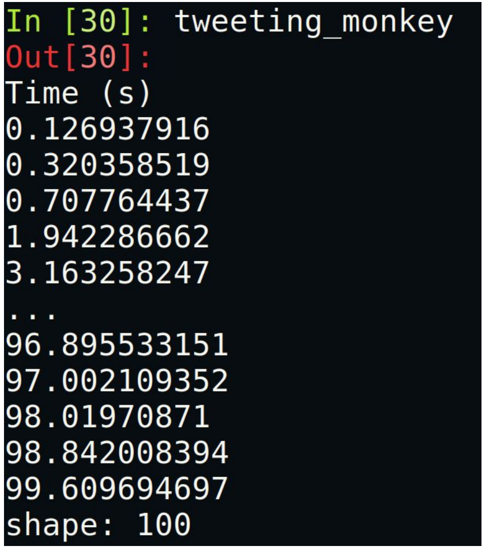


Time support

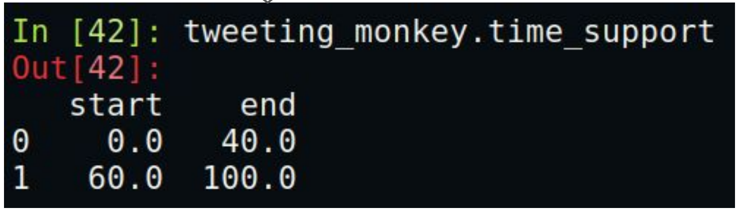


Tweet frequency?



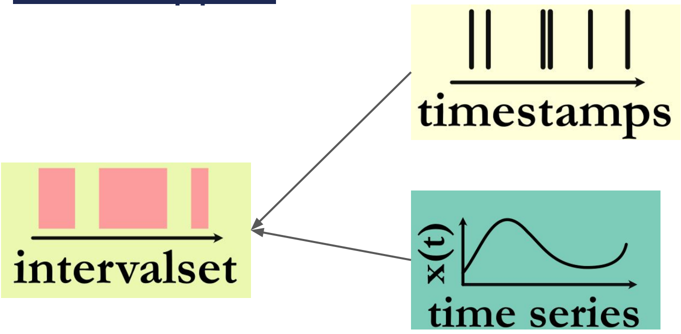


```
In [30]: tweeting_monkey
Out[30]:
Time (s)
0.126937916
0.320358519
0.707764437
1.942286662
3.163258247
...
96.895533151
97.002109352
98.01970871
98.842008394
99.609694697
shape: 100
```

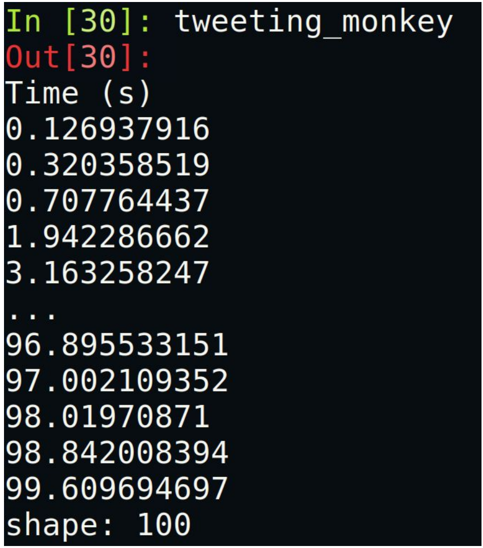


```
In [42]: tweeting_monkey.time_support
Out[42]:
    start    end
0     0.0   40.0
1    60.0  100.0
```

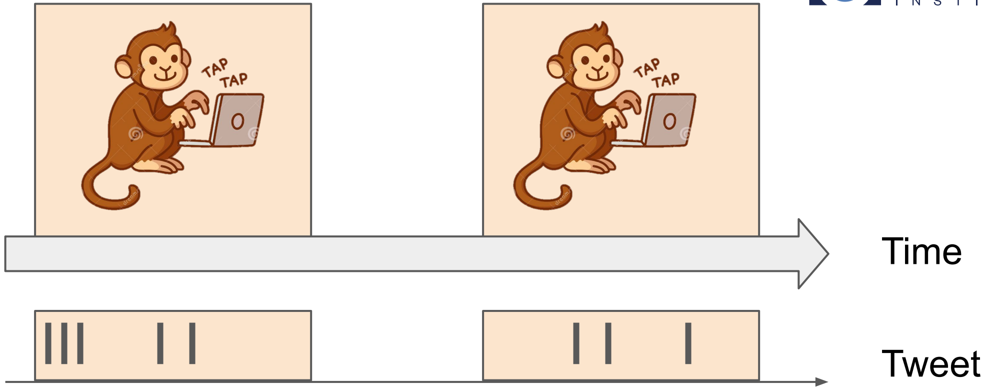
Time support



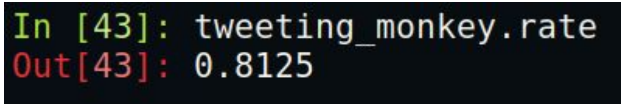
Tweet frequency?



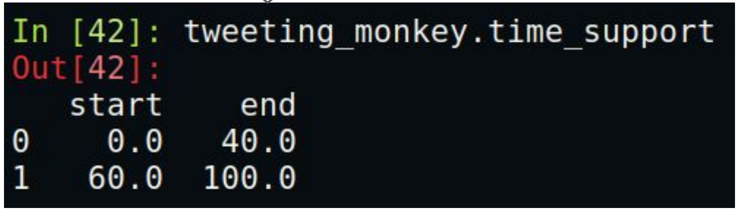
```
In [30]: tweeting_monkey
Out[30]:
Time (s)
0.126937916
0.320358519
0.707764437
1.942286662
3.163258247
...
96.895533151
97.002109352
98.01970871
98.842008394
99.609694697
shape: 100
```



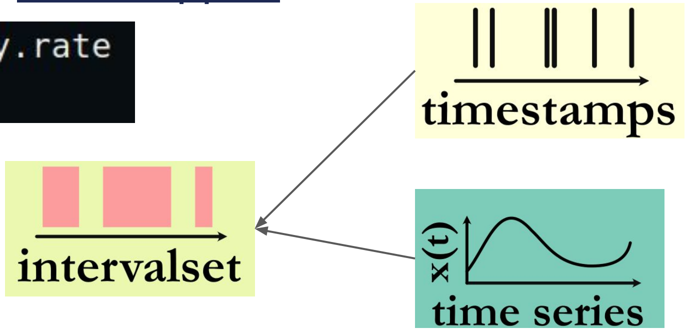
Time support



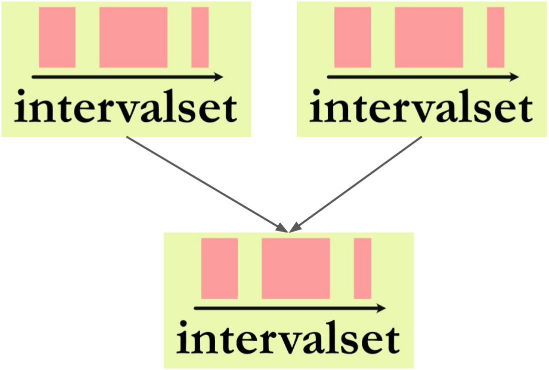
```
In [43]: tweeting_monkey.rate
Out[43]: 0.8125
```



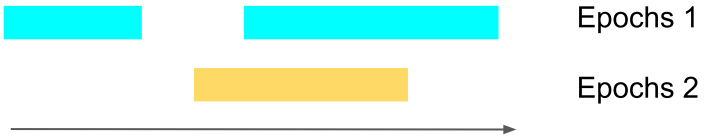
```
In [42]: tweeting_monkey.time_support
Out[42]:
   start    end
0    0.0   40.0
1   60.0  100.0
```

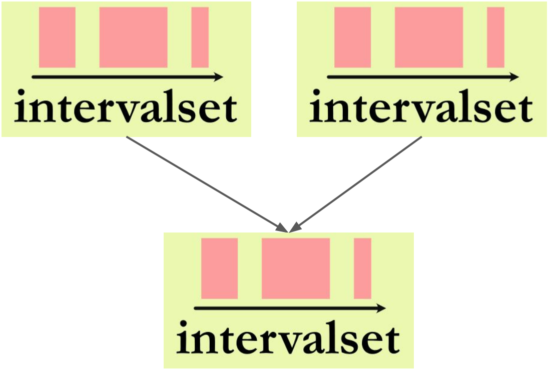


IntervalSet operations

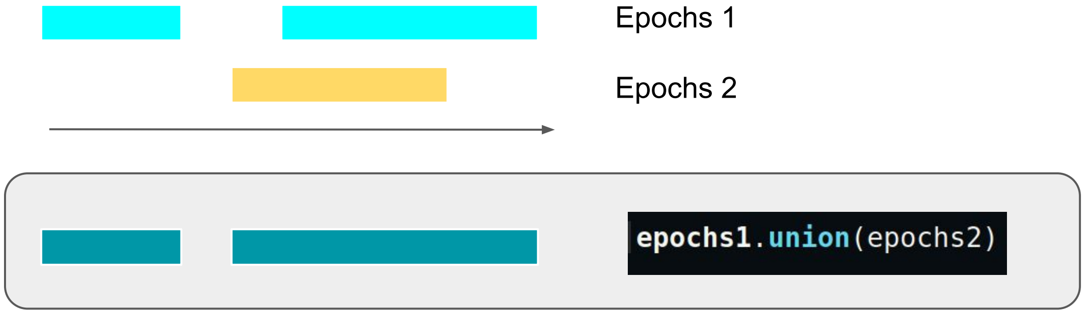


IntervalSet operations

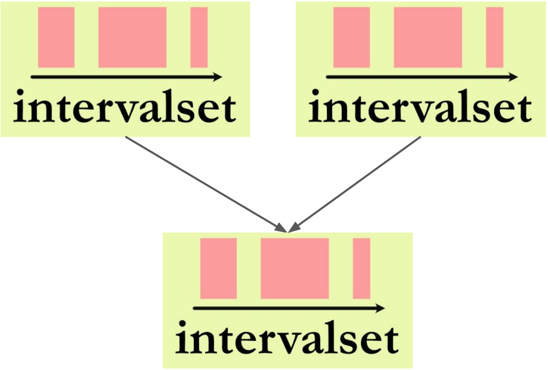




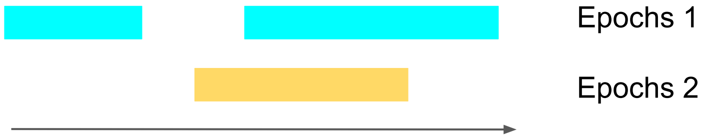
IntervalSet operations

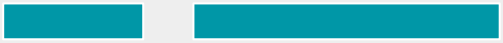


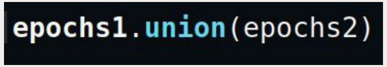
```
epochs1.union(epochs2)
```



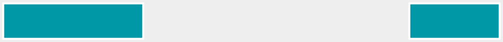
IntervalSet operations

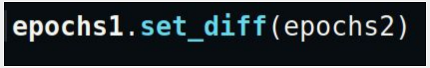




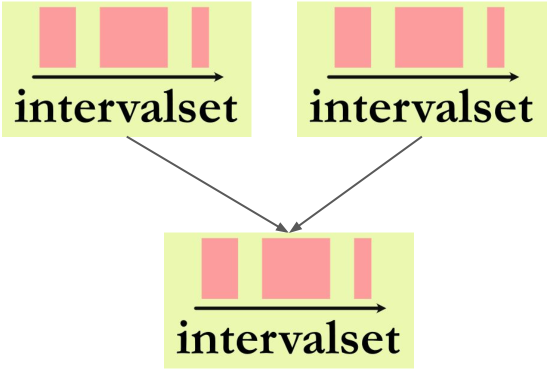


```
epochs1.union(epochs2)
```

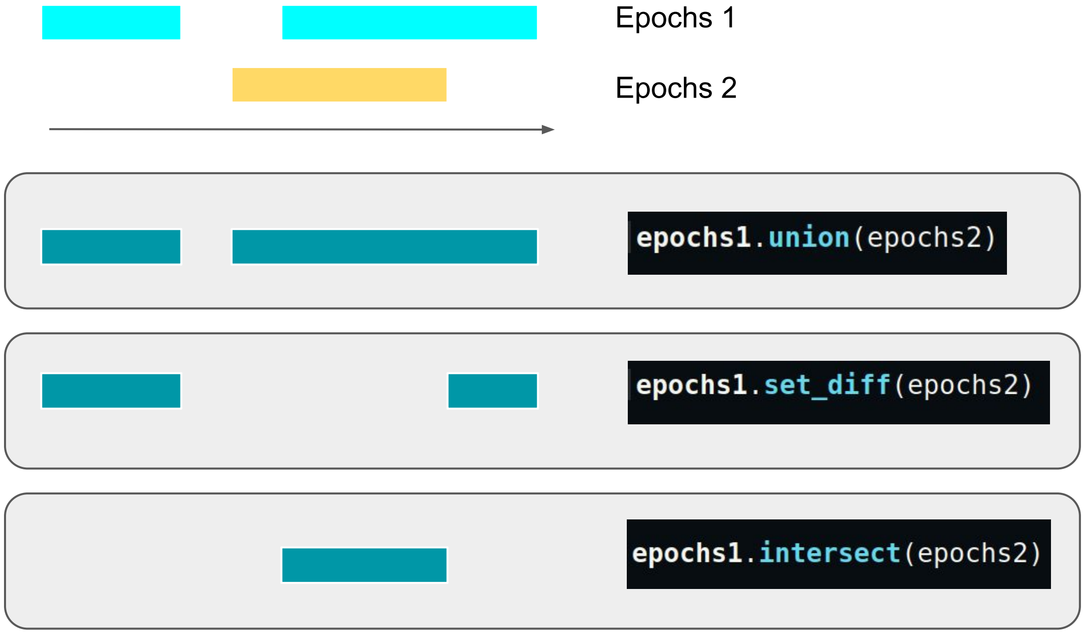




```
epochs1.set_diff(epochs2)
```



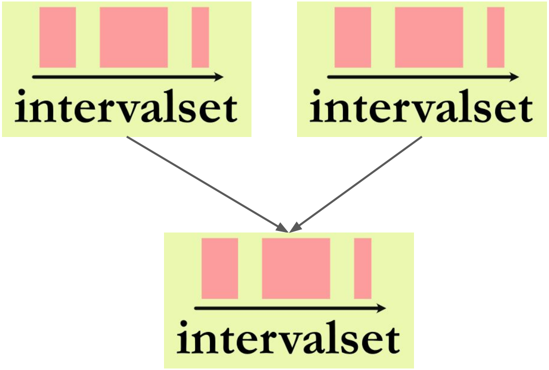
IntervalSet operations



```
epochs1.union(epochs2)
```

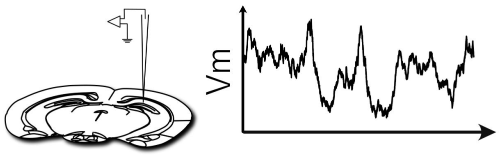
```
epochs1.set_diff(epochs2)
```

```
epochs1.intersect(epochs2)
```

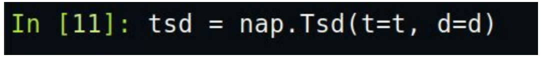


Summary : 6 objects to represent data

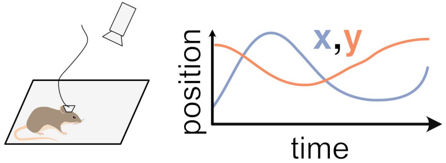
FLATIRON INSTITUTE



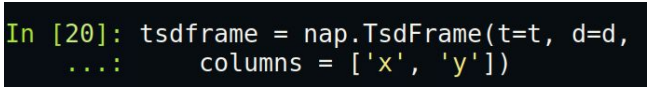
Tsd: 1-dimension



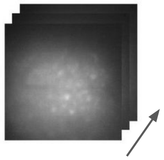
```
In [11]: tsd = nap.Tsd(t=t, d=d)
```



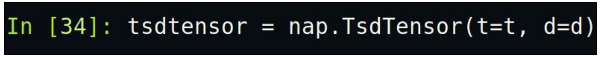
TsdFrame: 2-dimensions



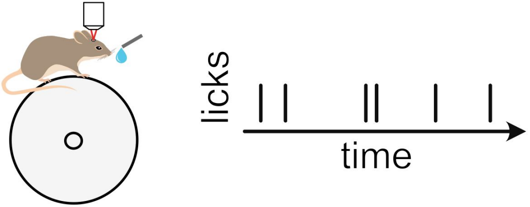
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:     columns = ['x', 'y'])
```



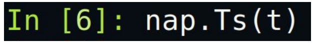
TsdTensor: n-dimensions



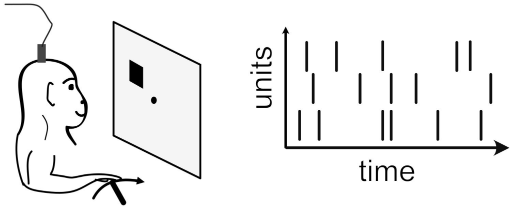
```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)
```



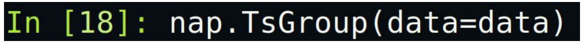
Ts: Timestamps



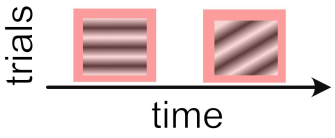
```
In [6]: nap.Ts(t)
```



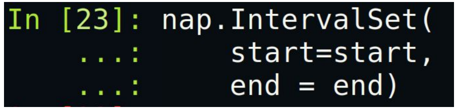
TsGroup: group of timestamps



```
In [18]: nap.TsGroup(data=data)
```

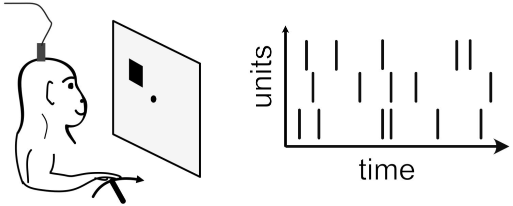


IntervalSet: set of epochs

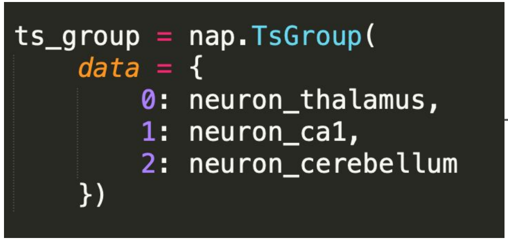


```
In [23]: nap.IntervalSet(
    ...:     start=start,
    ...:     end = end)
```

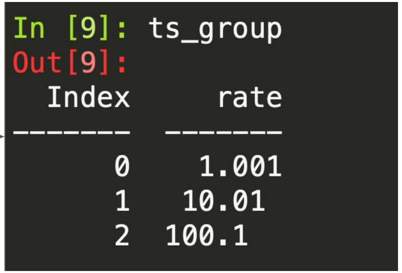
Metadata : extra informations



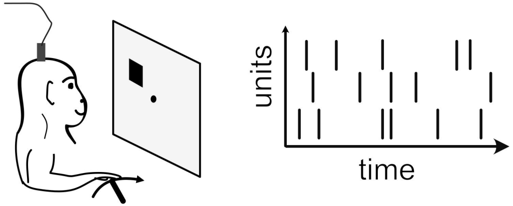
TsGroup: group of timestamps



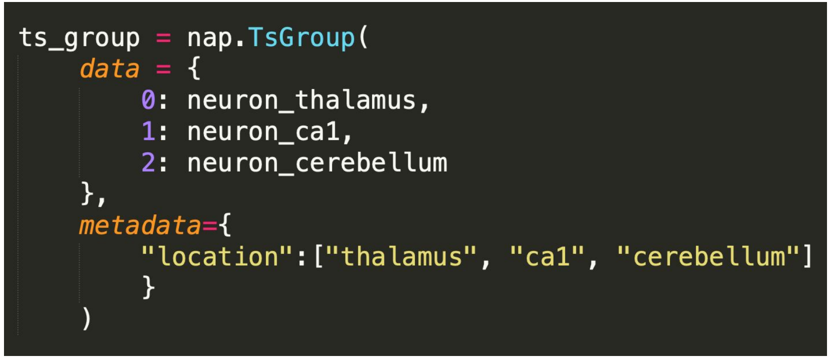
```
ts_group = nap.TsGroup(
    data = {
        0: neuron_thalamus,
        1: neuron_ca1,
        2: neuron_cerebellum
    })
```



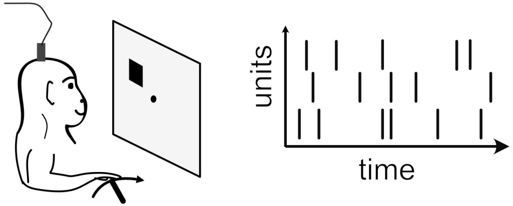
```
In [9]: ts_group
Out[9]:
  Index      rate
-------   -------
      0     1.001
      1    10.01
      2   100.1
```



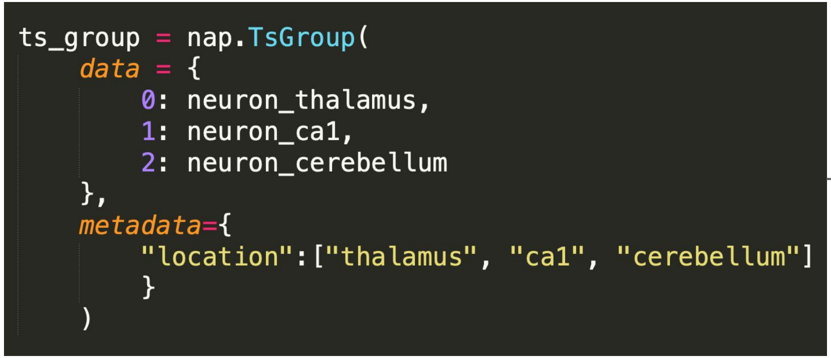
TsGroup: group of timestamps



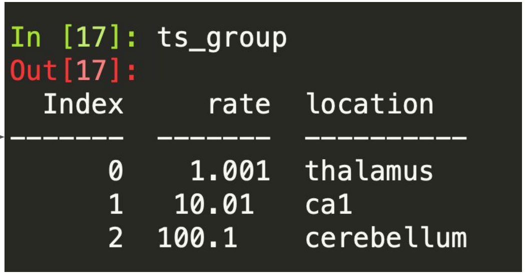
```
ts_group = nap.TsGroup(
    data = {
        0: neuron_thalamus,
        1: neuron_ca1,
        2: neuron_cerebellum
    },
    metadata={
        "location":["thalamus", "ca1", "cerebellum"]
        }
    )
```



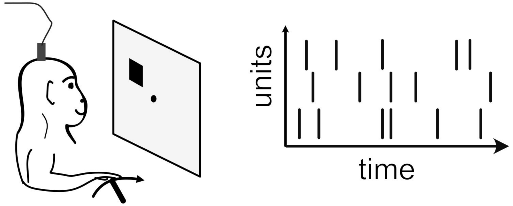
TsGroup: group of timestamps



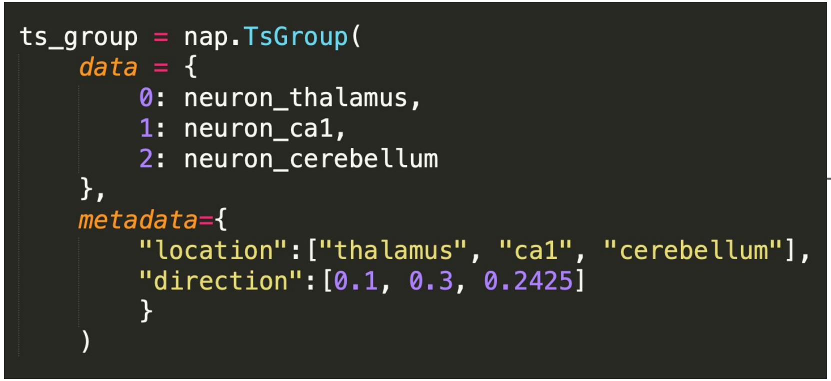
```
ts_group = nap.TsGroup(
    data = {
        0: neuron_thalamus,
        1: neuron_ca1,
        2: neuron_cerebellum
    },
    metadata={
        "location":["thalamus", "ca1", "cerebellum"]
        }
    )
```



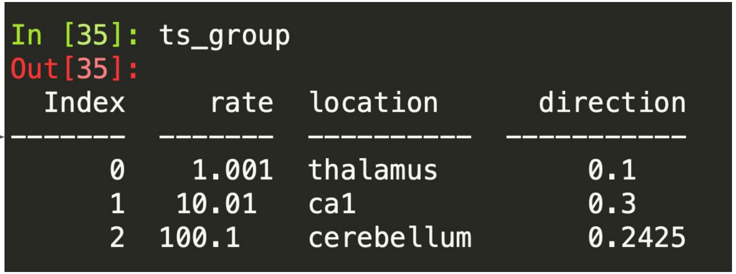
```
In [17]: ts_group
Out[17]:
  Index      rate  location
-------  -------  ----------
      0    1.001  thalamus
      1   10.01   ca1
      2  100.1    cerebellum
```



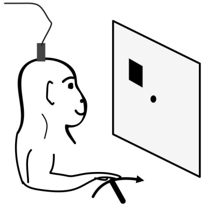
TsGroup: group of timestamps

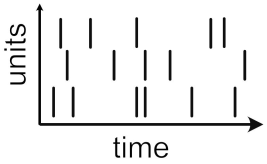


```
ts_group = nap.TsGroup(
    data = {
        0: neuron_thalamus,
        1: neuron_ca1,
        2: neuron_cerebellum
    },
    metadata={
        "location":["thalamus", "ca1", "cerebellum"],
        "direction":[0.1, 0.3, 0.2425]
        }
    )
```

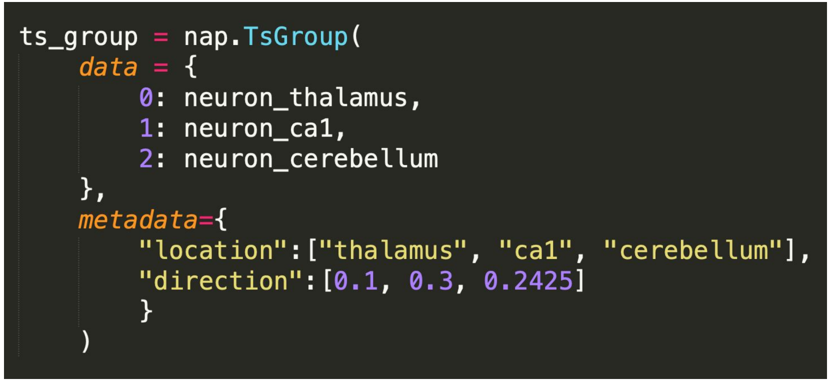


```
In [35]: ts_group
Out[35]:
  Index     rate  location       direction
-------  -------  ----------  -----------
      0    1.001  thalamus            0.1
      1   10.01   ca1                 0.3
      2  100.1    cerebellum          0.2425
```

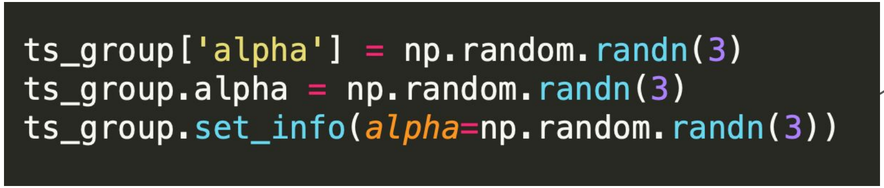




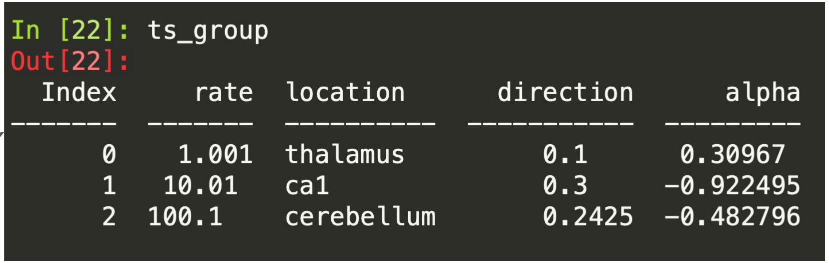
TsGroup: group of timestamps



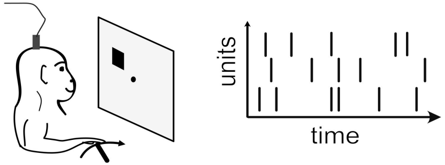
```
ts_group = nap.TsGroup(
    data = {
        0: neuron_thalamus,
        1: neuron_ca1,
        2: neuron_cerebellum
    },
    metadata={
        "location":["thalamus", "ca1", "cerebellum"],
        "direction":[0.1, 0.3, 0.2425]
        }
    )
```



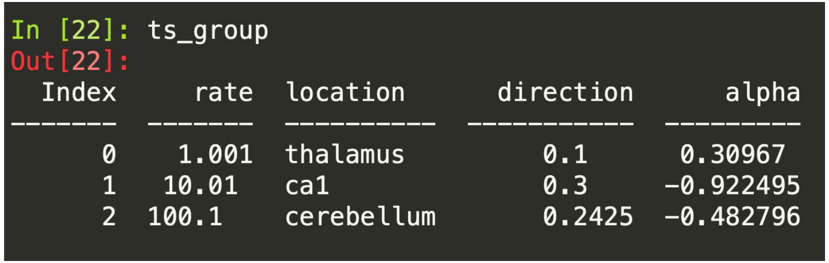
```
ts_group['alpha'] = np.random.randn(3)
ts_group.alpha = np.random.randn(3)
ts_group.set_info(alpha=np.random.randn(3))
```



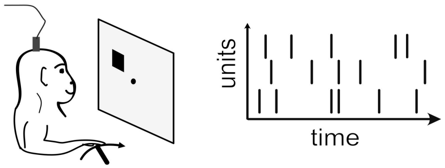
```
In [22]: ts_group
Out[22]:
  Index      rate  location      direction      alpha
-------  -------  ----------  -----------  ---------
      0    1.001  thalamus          0.1     0.30967
      1   10.01   ca1               0.3    -0.922495
      2  100.1    cerebellum        0.2425 -0.482796
```



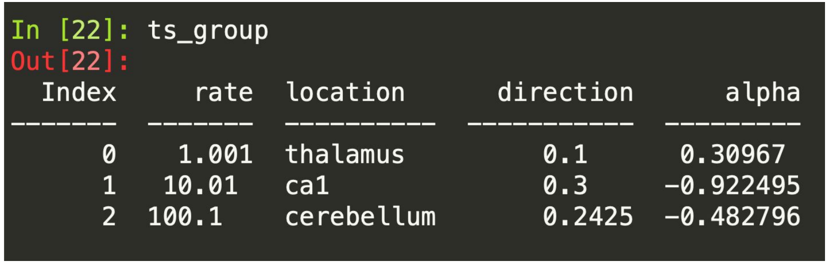
TsGroup: group of timestamps



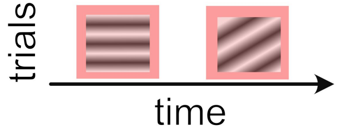
```
In [22]: ts_group
Out[22]:
  Index     rate  location        direction      alpha
-------  -------  ----------  -----------  ---------
      0    1.001  thalamus         0.1       0.30967
      1   10.01   ca1              0.3      -0.922495
      2  100.1    cerebellum       0.2425   -0.482796
```



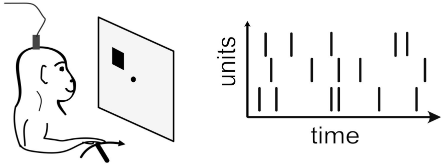
TsGroup: group of timestamps



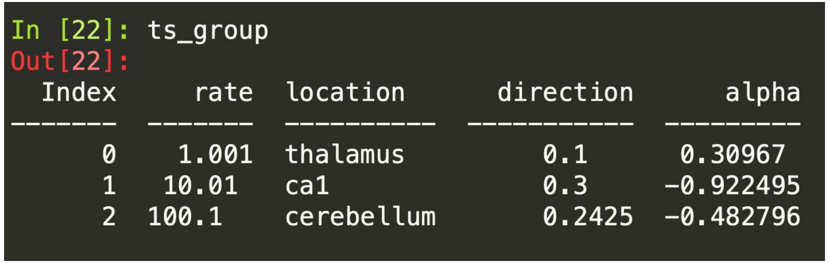
```
In [22]: ts_group
Out[22]:
  Index     rate  location      direction       alpha
-------  -------  ----------  -----------  ---------
      0    1.001  thalamus         0.1      0.30967
      1   10.01   ca1              0.3     -0.922495
      2  100.1    cerebellum       0.2425  -0.482796
```



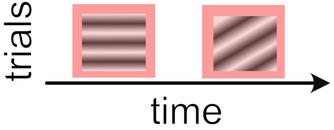
IntervalSet: set of epochs



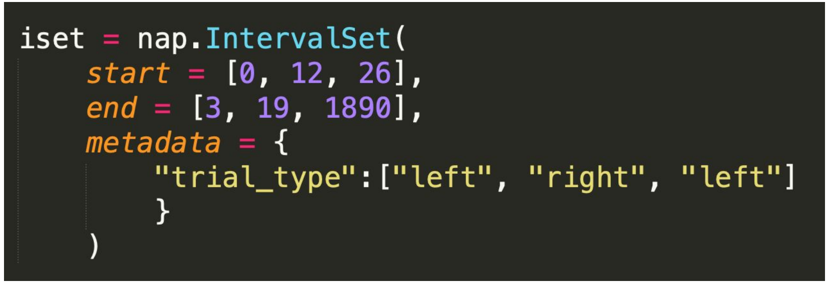
TsGroup: group of timestamps



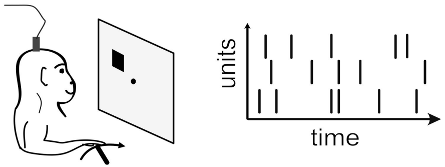
```
In [22]: ts_group
Out[22]:
  Index      rate   location          direction      alpha
-------   -------   ----------      -----------  ---------
      0     1.001   thalamus              0.1      0.30967
      1    10.01    ca1                   0.3    -0.922495
      2   100.1     cerebellum         0.2425    -0.482796
```



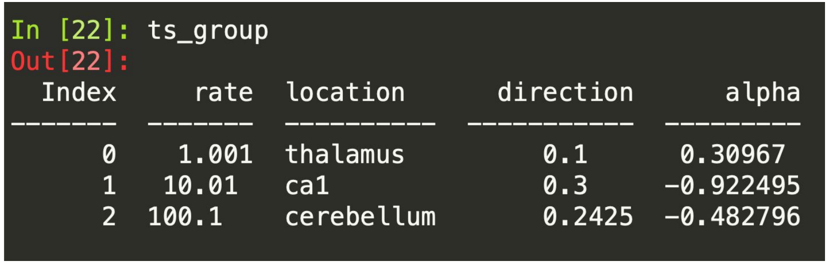
IntervalSet: set of epochs



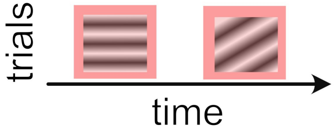
```
iset = nap.IntervalSet(
    start = [0, 12, 26],
    end = [3, 19, 1890],
    metadata = {
        "trial_type":["left", "right", "left"]
        }
    )
```



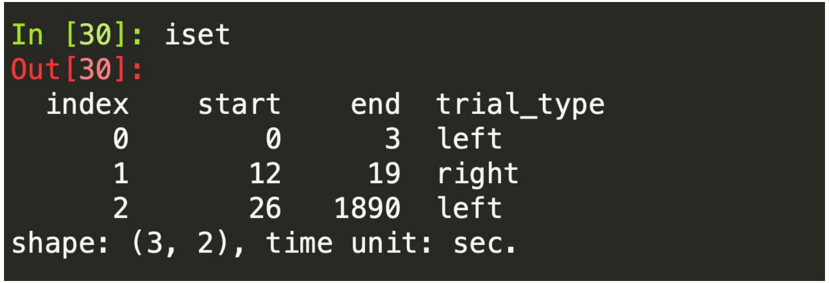
TsGroup: group of timestamps



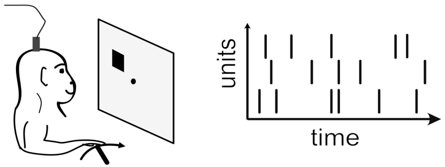
```
In [22]: ts_group
Out[22]:
  Index      rate  location        direction      alpha
-------  -------  ----------  -----------  ---------
      0    1.001  thalamus            0.1    0.30967
      1   10.01   ca1                 0.3   -0.922495
      2  100.1    cerebellum       0.2425   -0.482796
```



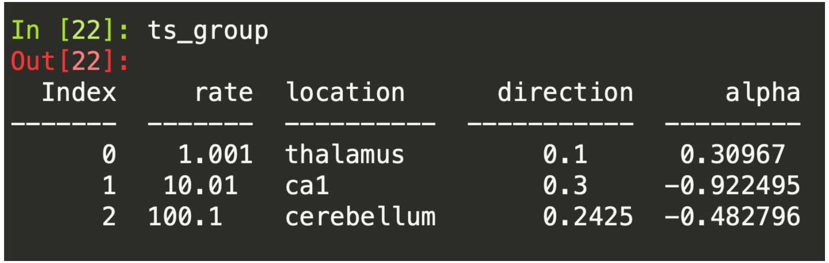
IntervalSet: set of epochs



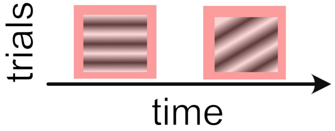
```
In [30]: iset
Out[30]:
  index     start     end  trial_type
      0         0       3  left
      1        12      19  right
      2        26    1890  left
shape: (3, 2), time unit: sec.
```



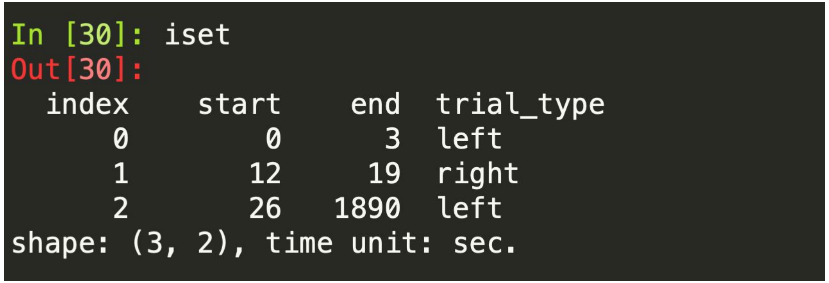
TsGroup: group of timestamps



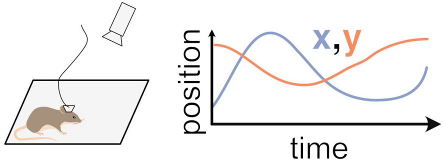
```
In [22]: ts_group
Out[22]:
  Index      rate  location      direction      alpha
-------  --------  ----------  -----------  ---------
      0     1.001  thalamus            0.1    0.30967
      1    10.01   ca1                 0.3  -0.922495
      2   100.1    cerebellum       0.2425  -0.482796
```



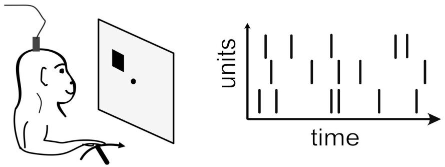
IntervalSet: set of epochs



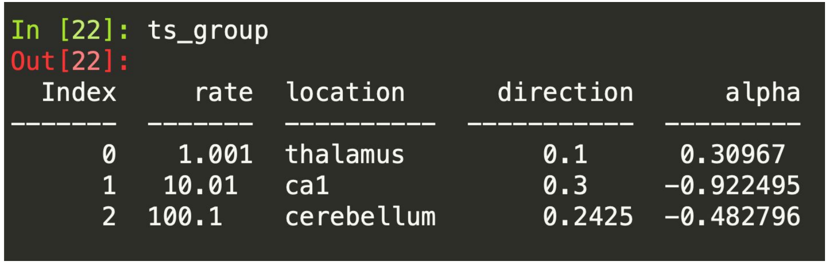
```
In [30]: iset
Out[30]:
  index     start     end  trial_type
      0         0       3  left
      1        12      19  right
      2        26    1890  left
shape: (3, 2), time unit: sec.
```



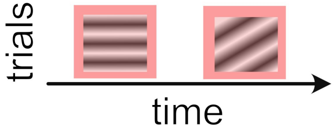
TsdFrame: 2-dimensions



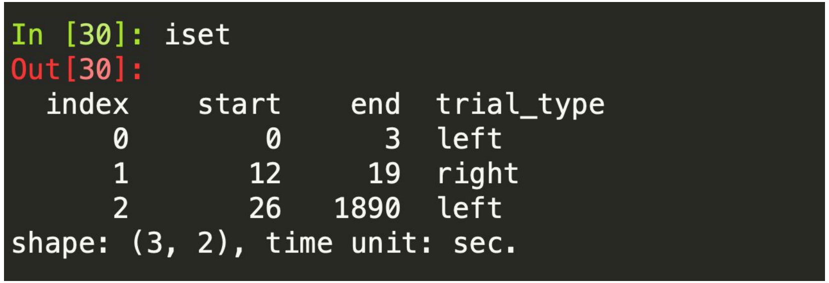
TsGroup: group of timestamps



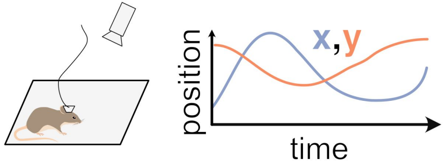
```
In [22]: ts_group
Out[22]:
  Index      rate  location      direction      alpha
-------  --------  ----------  -----------  ---------
      0     1.001  thalamus            0.1    0.30967
      1    10.01   ca1                 0.3  -0.922495
      2   100.1    cerebellum       0.2425  -0.482796
```



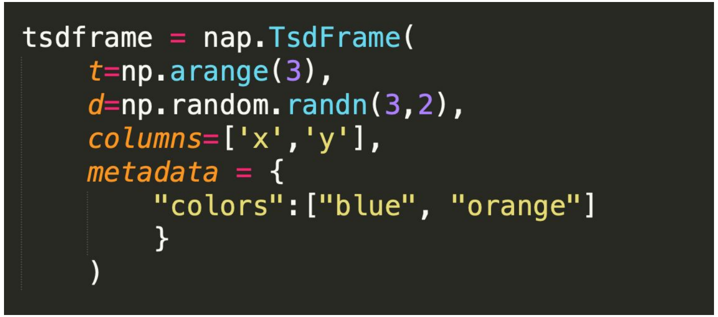
IntervalSet: set of epochs



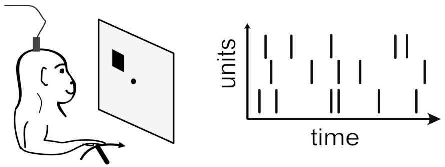
```
In [30]: iset
Out[30]:
  index    start    end  trial_type
      0        0      3  left
      1       12     19  right
      2       26   1890  left
shape: (3, 2), time unit: sec.
```



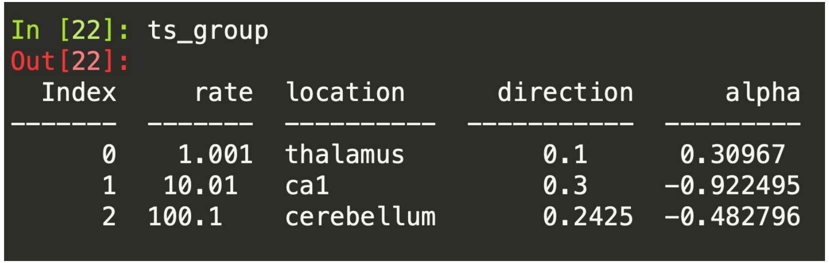
TsdFrame: 2-dimensions



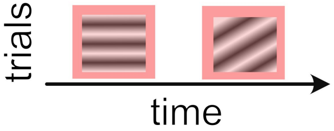
```
tsdframe = nap.TsdFrame(
    t=np.arange(3),
    d=np.random.randn(3,2),
    columns=['x','y'],
    metadata = {
        "colors":["blue", "orange"]
        }
    )
```



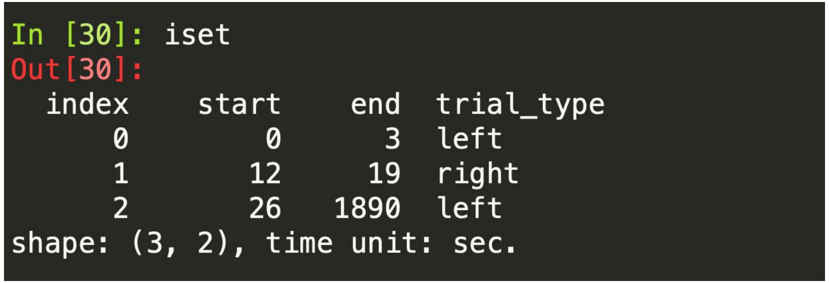
TsGroup: group of timestamps



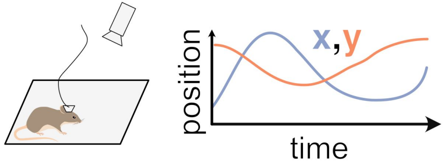
```
In [22]: ts_group
Out[22]:
  Index      rate   location        direction      alpha
-------  --------   ----------    -----------  ---------
      0     1.001   thalamus              0.1    0.30967
      1    10.01    ca1                   0.3  -0.922495
      2   100.1     cerebellum         0.2425  -0.482796
```



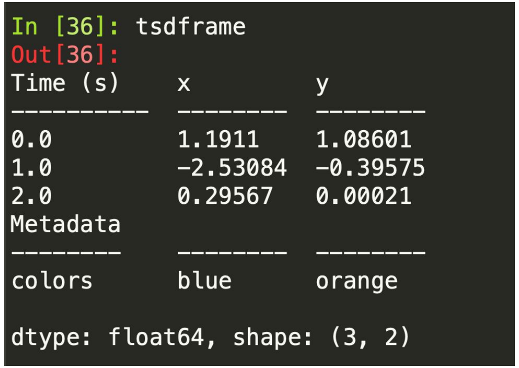
IntervalSet: set of epochs



```
In [30]: iset
Out[30]:
  index     start     end  trial_type
      0         0       3  left
      1        12      19  right
      2        26    1890  left
shape: (3, 2), time unit: sec.
```

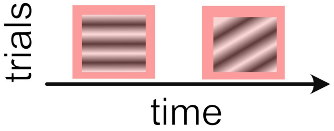


TsdFrame: 2-dimensions

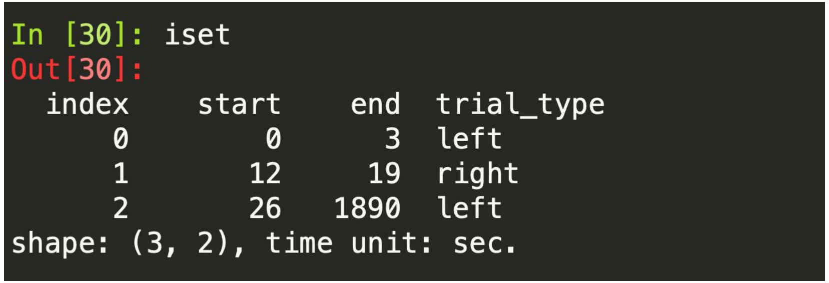


```
In [36]: tsdframe
Out[36]:
Time (s)    x         y
----------  --------  --------
0.0         1.1911    1.08601
1.0         -2.53084  -0.39575
2.0         0.29567   0.00021
Metadata
--------    --------  --------
colors      blue      orange

dtype: float64, shape: (3, 2)
```

Accessing metadata

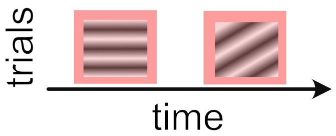


IntervalSet: set of epochs

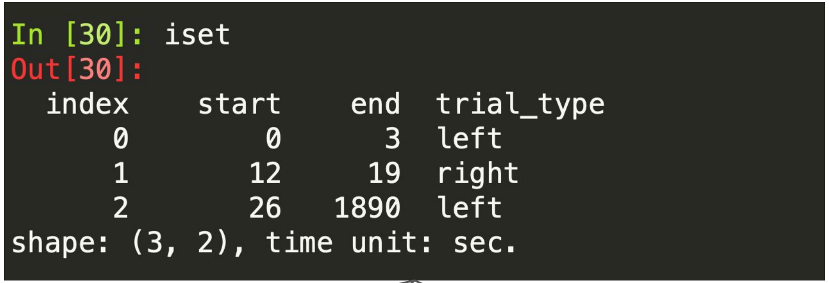


```
In [30]: iset
Out[30]:
  index     start     end  trial_type
      0         0       3  left
      1        12      19  right
      2        26    1890  left
shape: (3, 2), time unit: sec.
```

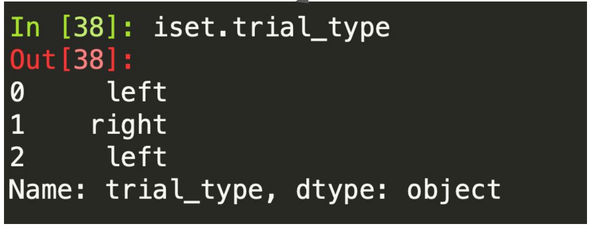
Accessing metadata



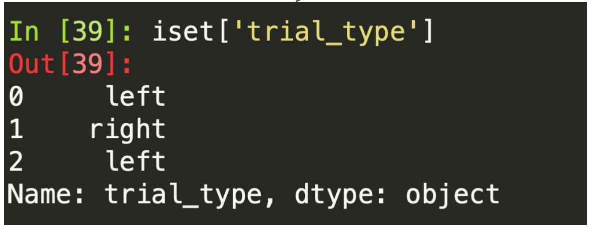
IntervalSet: set of epochs



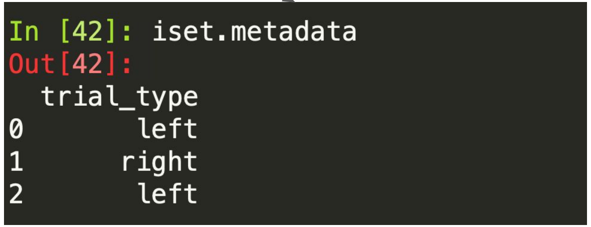
```
In [30]: iset
Out[30]:
  index     start     end  trial_type
      0         0       3  left
      1        12      19  right
      2        26    1890  left
shape: (3, 2), time unit: sec.
```



```
In [38]: iset.trial_type
Out[38]:
0     left
1    right
2     left
Name: trial_type, dtype: object
```

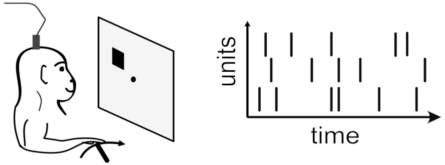


```
In [39]: iset['trial_type']
Out[39]:
0     left
1    right
2     left
Name: trial_type, dtype: object
```

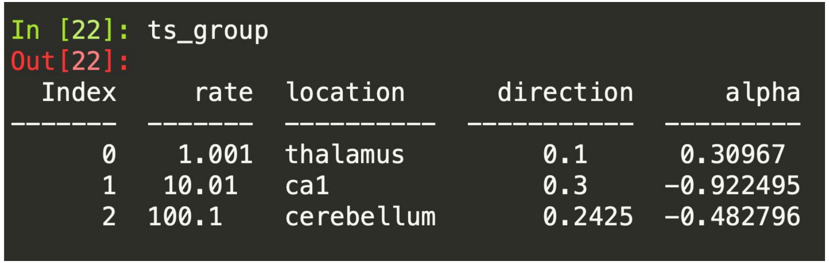


```
In [42]: iset.metadata
Out[42]:
  trial_type
0       left
1      right
2       left
```

Slicing using metadata

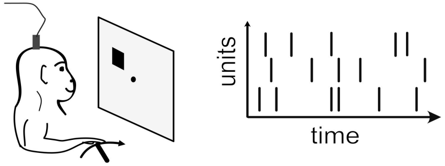


TsGroup: group of timestamps

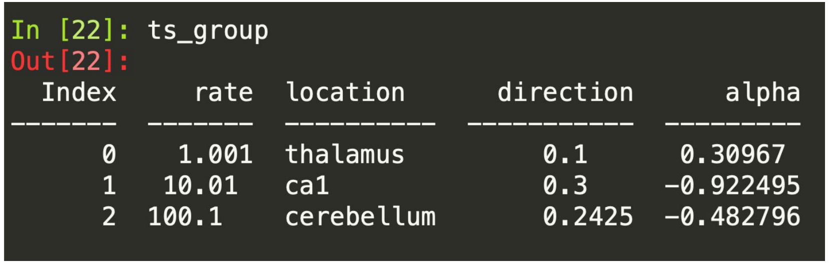


```
In [22]: ts_group
Out[22]:
  Index      rate  location      direction      alpha
-------  --------  -----------  -----------  ---------
      0     1.001  thalamus         0.1       0.30967
      1    10.01   ca1              0.3      -0.922495
      2   100.1    cerebellum       0.2425   -0.482796
```

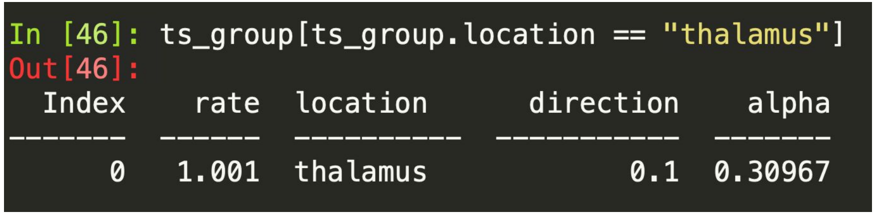
Slicing using metadata



TsGroup: group of timestamps

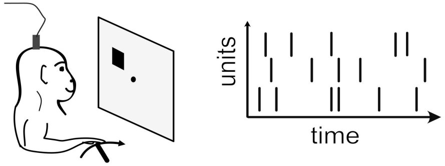


```
In [22]: ts_group
Out[22]:
  Index      rate  location       direction      alpha
-------   -------  ----------   -----------  ---------
      0     1.001  thalamus             0.1    0.30967
      1    10.01   ca1                  0.3  -0.922495
      2   100.1    cerebellum        0.2425  -0.482796
```

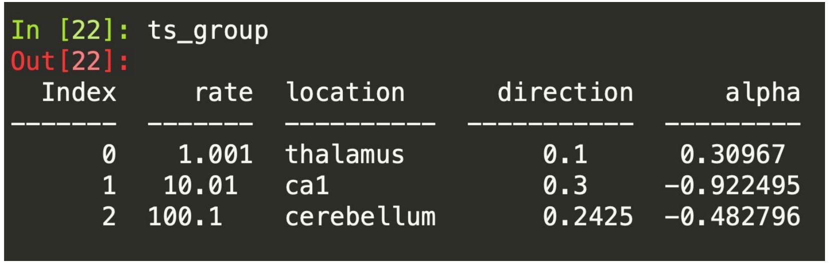


```
In [46]: ts_group[ts_group.location == "thalamus"]
Out[46]:
  Index    rate  location       direction    alpha
-------  ------  ----------   -----------  -------
      0   1.001  thalamus             0.1  0.30967
```

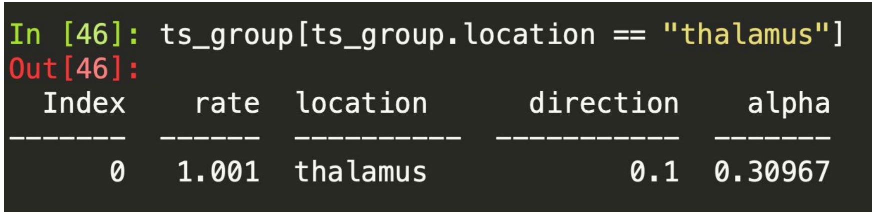
Slicing using metadata



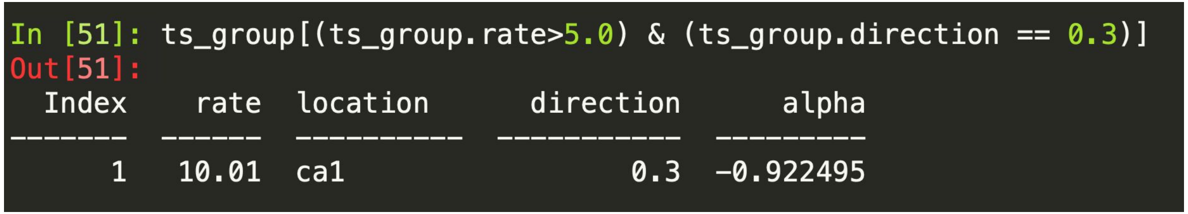
TsGroup: group of timestamps



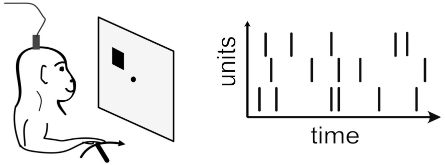
```
In [22]: ts_group
Out[22]:
  Index     rate  location      direction      alpha
-------  -------  ----------  -----------  ---------
      0    1.001  thalamus         0.1       0.30967
      1   10.01   ca1              0.3      -0.922495
      2  100.1    cerebellum       0.2425   -0.482796
```



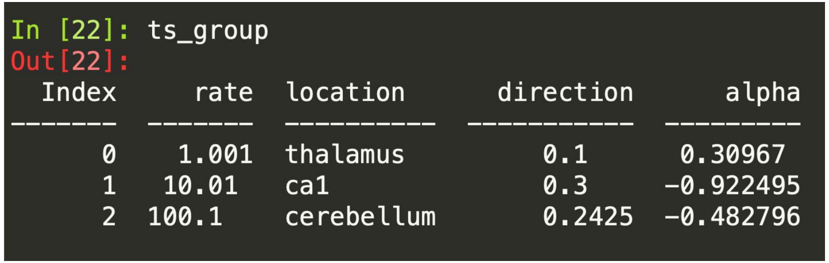
```
In [46]: ts_group[ts_group.location == "thalamus"]
Out[46]:
  Index    rate  location      direction    alpha
-------  ------  ----------  -----------  -------
      0   1.001  thalamus            0.1  0.30967
```



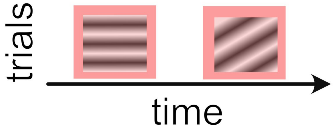
```
In [51]: ts_group[(ts_group.rate>5.0) & (ts_group.direction == 0.3)]
Out[51]:
  Index    rate  location      direction      alpha
-------  ------  ----------  -----------  ---------
      1   10.01  ca1                 0.3  -0.922495
```



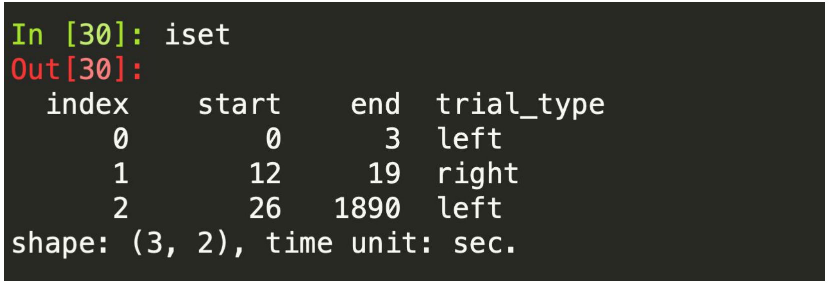
TsGroup: group of timestamps



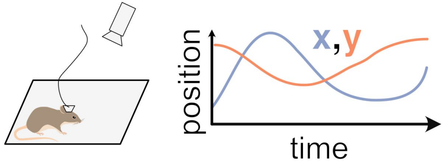
```
In [22]: ts_group
Out[22]:
  Index     rate  location      direction      alpha
-------  -------  ----------  -----------  ---------
      0    1.001  thalamus         0.1       0.30967
      1   10.01   ca1              0.3      -0.922495
      2  100.1    cerebellum       0.2425   -0.482796
```



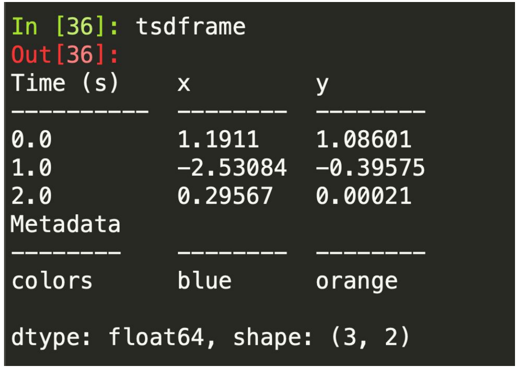
IntervalSet: set of epochs



```
In [30]: iset
Out[30]:
  index    start    end  trial_type
      0        0      3  left
      1       12     19  right
      2       26   1890  left
shape: (3, 2), time unit: sec.
```



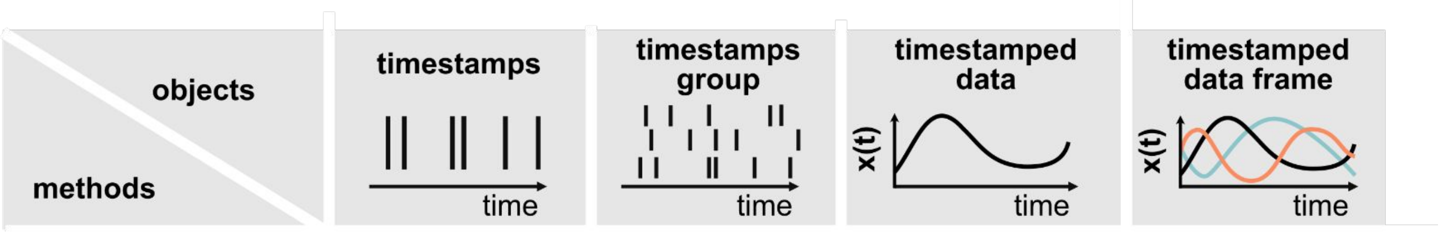
TsdFrame: 2-dimensions



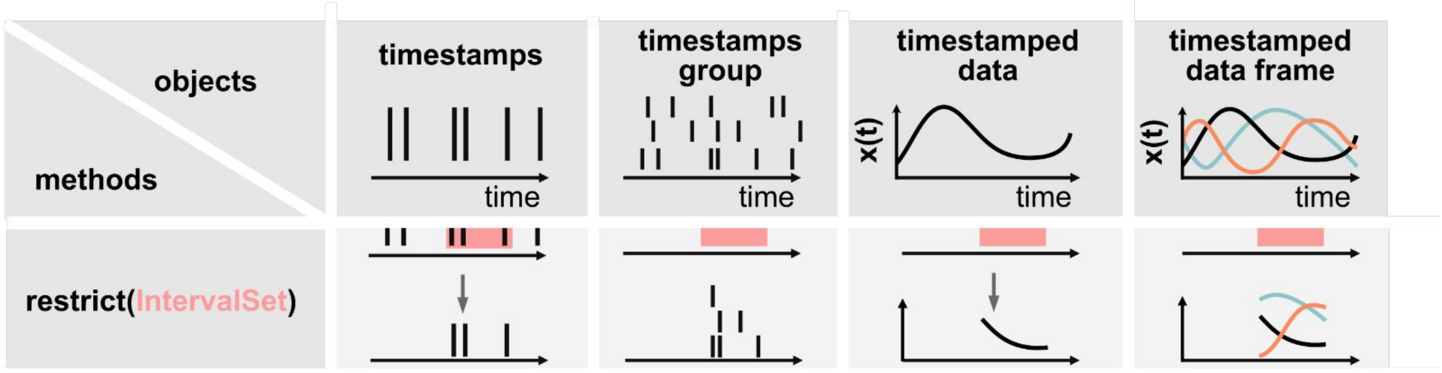
```
In [36]: tsdframe
Out[36]:
Time (s)    x         y
----------  --------  --------
0.0         1.1911    1.08601
1.0         -2.53084  -0.39575
2.0         0.29567   0.00021
Metadata
--------    --------  --------
colors      blue      orange

dtype: float64, shape: (3, 2)
```

FLATIRON INSTITUTE

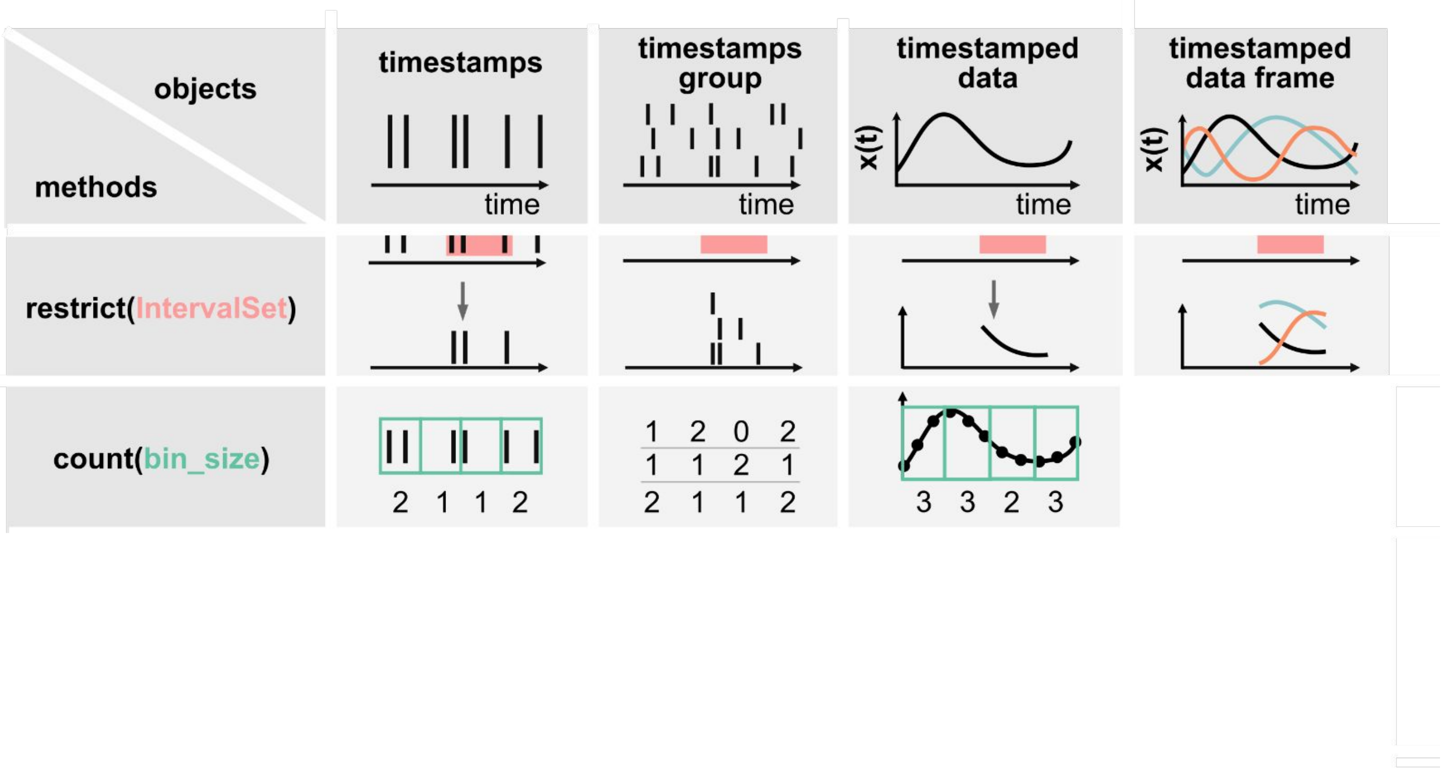
Core functions of pynapple



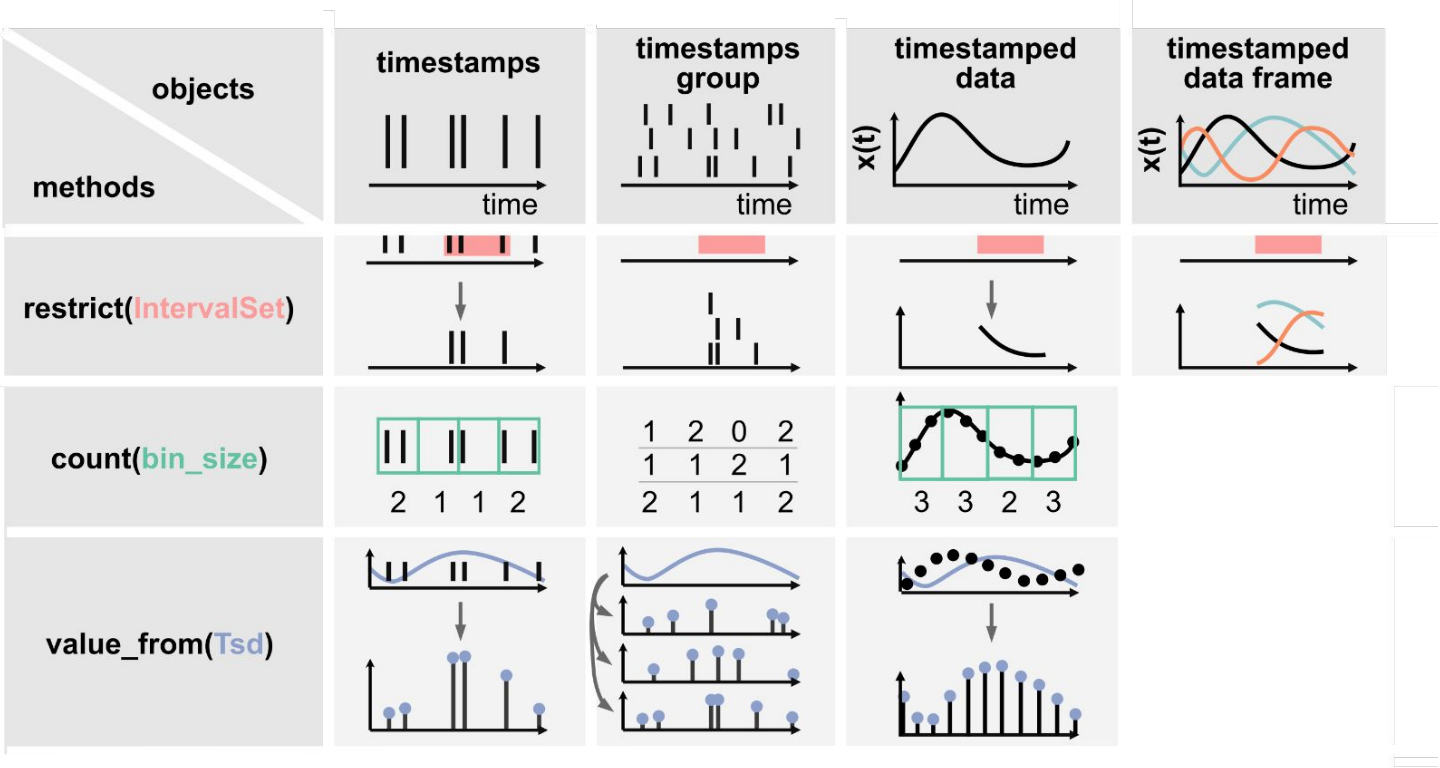
Viejo, G., Levenstein, D., Carrasco, S. S., Mehrotra, D., Mahallati, S., Vite, G. R., ... & Peyrache, A. (2023). Pynapple, a toolbox for data analysis in neuroscience. *eLife*, *12*, RP85786.



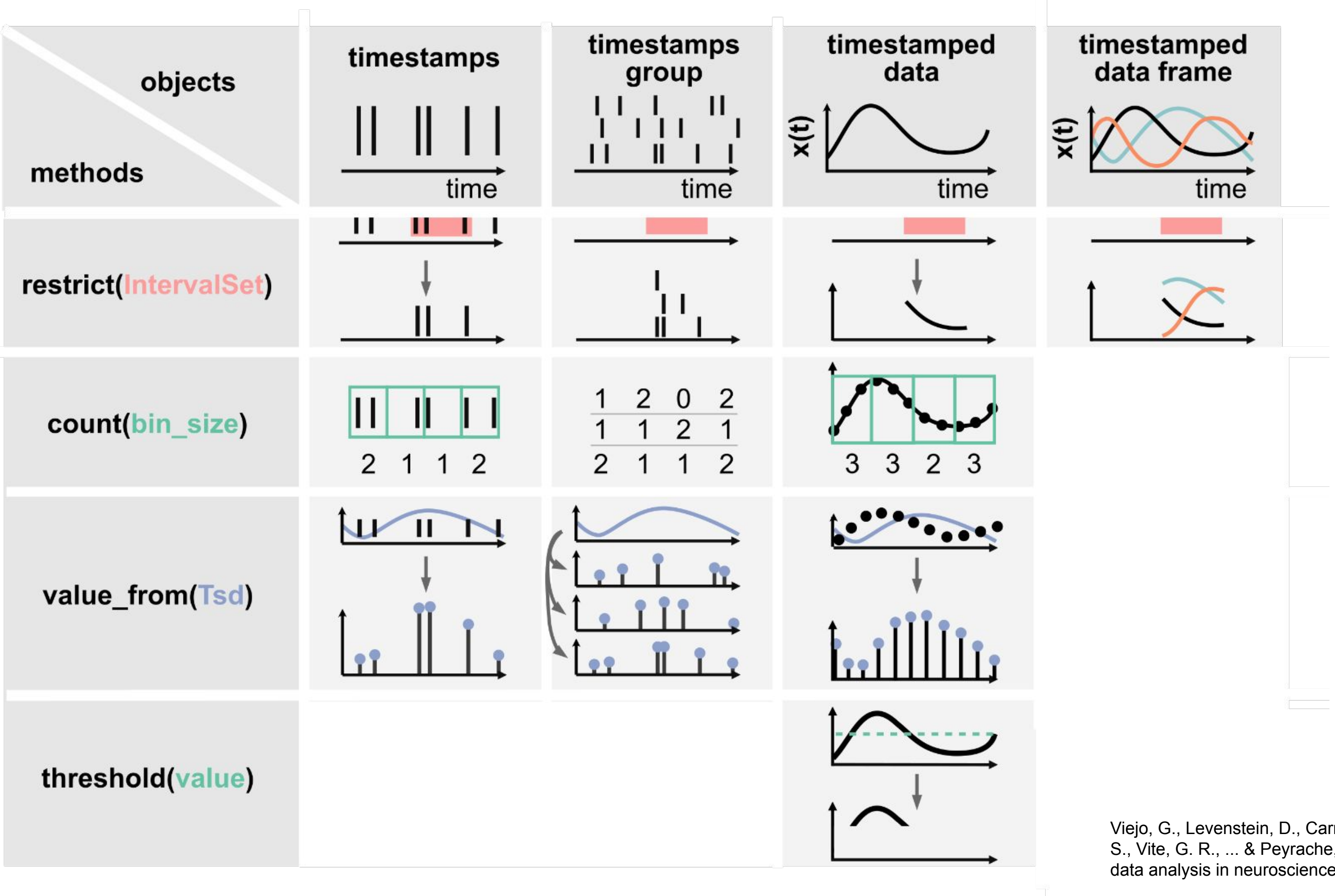
Viejo, G., Levenstein, D., Carrasco, S. S., Mehrotra, D., Mahallati, S., Vite, G. R., ... & Peyrache, A. (2023). Pynapple, a toolbox for data analysis in neuroscience. *eLife*, *12*, RP85786.



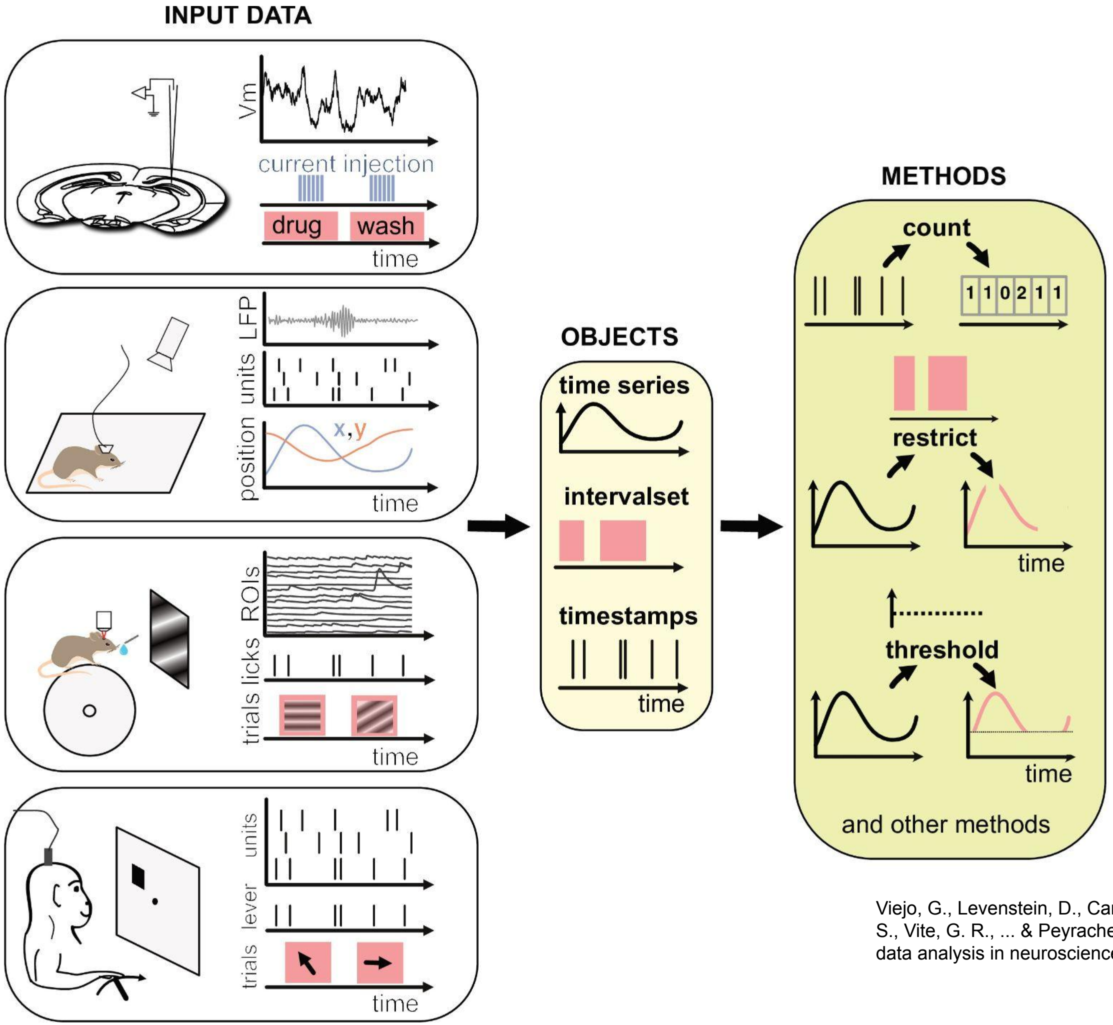
Viejo, G., Levenstein, D., Carrasco, S. S., Mehrotra, D., Mahallati, S., Vite, G. R., ... & Peyrache, A. (2023). Pynapple, a toolbox for data analysis in neuroscience. *eLife*, *12*, RP85786.



Viejo, G., Levenstein, D., Carrasco, S. S., Mehrotra, D., Mahallati, S., Vite, G. R., ... & Peyrache, A. (2023). Pynapple, a toolbox for data analysis in neuroscience. *eLife*, *12*, RP85786.



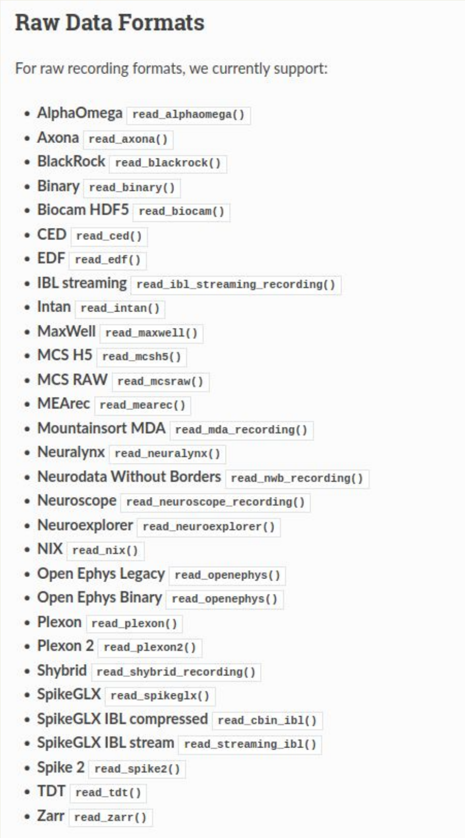
Viejo, G., Levenstein, D., Carrasco, S. S., Mehrotra, D., Mahallati, S., Vite, G. R., ... & Peyrache, A. (2023). Pynapple, a toolbox for data analysis in neuroscience. *eLife*, *12*, RP85786.



Viejo, G., Levenstein, D., Carrasco, S. S., Mehrotra, D., Mahallati, S., Vite, G. R., ... & Peyrache, A. (2023). Pynapple, a toolbox for data analysis in neuroscience. *eLife*, *12*, RP85786.

Pynapple IO

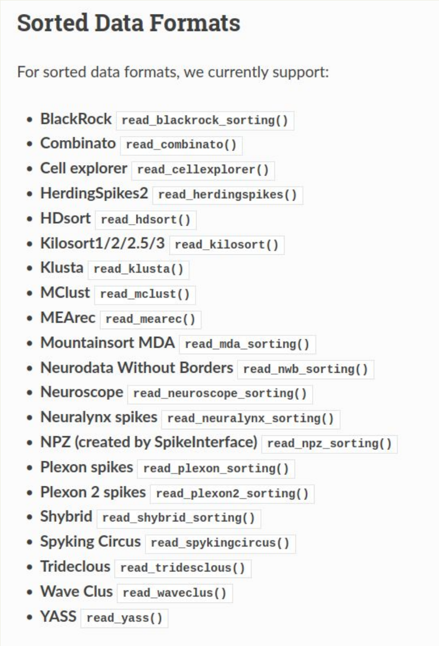
From **spikeinterface** : a unified framework for spike sorting



Raw Data Formats

For raw recording formats, we currently support:

- AlphaOmega `read_alphaomega()`
- Axona `read_axona()`
- BlackRock `read_blackrock()`
- Binary `read_binary()`
- Biocam HDF5 `read_biocam()`
- CED `read_ced()`
- EDF `read_edf()`
- IBL streaming `read_ibl_streaming_recording()`
- Intan `read_intan()`
- MaxWell `read_maxwell()`
- MCS H5 `read_mcsh5()`
- MCS RAW `read_mcsraw()`
- MEArec `read_mearec()`
- Mountainsort MDA `read_mda_recording()`
- Neuralynx `read_neuralynx()`
- Neurodata Without Borders `read_nwb_recording()`
- Neuroscope `read_neuroscope_recording()`
- Neuroexplorer `read_neuroexplorer()`
- NIX `read_nix()`
- Open Ephys Legacy `read_openephys()`
- Open Ephys Binary `read_openephys()`
- Plexon `read_plexon()`
- Plexon 2 `read_plexon2()`
- Shybrid `read_shybrid_recording()`
- SpikeGLX `read_spikeglx()`
- SpikeGLX IBL compressed `read_cbin_ibl()`
- SpikeGLX IBL stream `read_streaming_ibl()`
- Spike 2 `read_spike2()`
- TDT `read_tdt()`
- Zarr `read_zarr()`



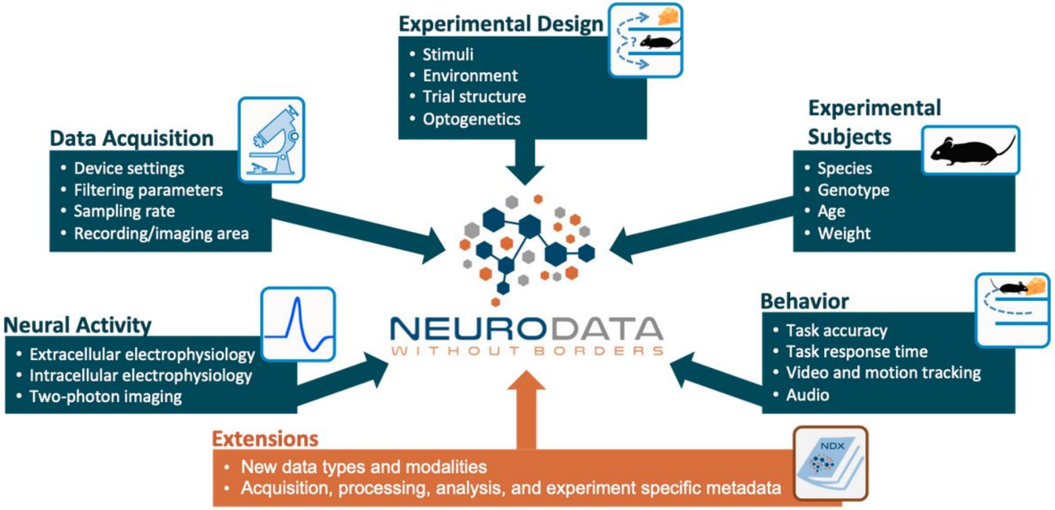
Sorted Data Formats

For sorted data formats, we currently support:

- BlackRock `read_blackrock_sorting()`
- Combinato `read_combinato()`
- Cell explorer `read_cellexplorer()`
- HerdingSpikes2 `read_herdingspikes()`
- HDsort `read_hdsort()`
- Kilosort1/2/2.5/3 `read_kilosort()`
- Klusta `read_klusta()`
- MClust `read_mclust()`
- MEArec `read_mearec()`
- Mountainsort MDA `read_mda_sorting()`
- Neurodata Without Borders `read_nwb_sorting()`
- Neuroscope `read_neuroscope_sorting()`
- Neuralynx spikes `read_neuralynx_sorting()`
- NPZ (created by SpikeInterface) `read_npz_sorting()`
- Plexon spikes `read_plexon_sorting()`
- Plexon 2 spikes `read_plexon2_sorting()`
- Shybrid `read_shybrid_sorting()`
- Spyking Circus `read_spykingcircus()`
- Trideclous `read_tridesclous()`
- Wave Clus `read_waveclus()`
- YASS `read_yass()`

On universal standard : the neurodata without borders format (NWB)

Neurodata Without Borders (NWB) is a data standard for neurophysiology, providing neuroscientists with a common standard to share, archive, use, and build common analysis tools for neurophysiology data.



Experimental Design
- Stimuli
- Environment
- Trial structure
- Optogenetics

Data Acquisition
- Device settings
- Filtering parameters
- Sampling rate
- Recording/imaging area

Experimental Subjects
- Species
- Genotype
- Age
- Weight

Neural Activity
- Extracellular electrophysiology
- Intracellular electrophysiology
- Two-photon imaging

Behavior
- Task accuracy
- Task response time
- Video and motion tracking
- Audio

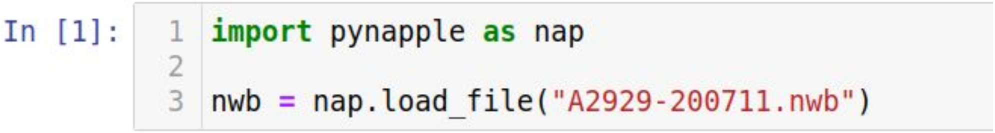
Extensions
- New data types and modalities
- Acquisition, processing, analysis, and experiment specific metadata

Teeters, J. L., Godfrey,K., Young, R., Dang, C., Friedsam, C., Wark, B., ... & Sommer, F. T. (2015). Neurodata without borders: creating a common data format for neurophysiology. *Neuron*, *88*(4), 629-634.

Rübel, O., Tritt, A., Ly, R., Dichter, B. K., Ghosh, S., Niu, L., ... & Bouchard, K. E. (2022). The neurodata without borders ecosystem for neurophysiological data science. *Elife*, *11*, e78362.

Loading NWB with pynapple

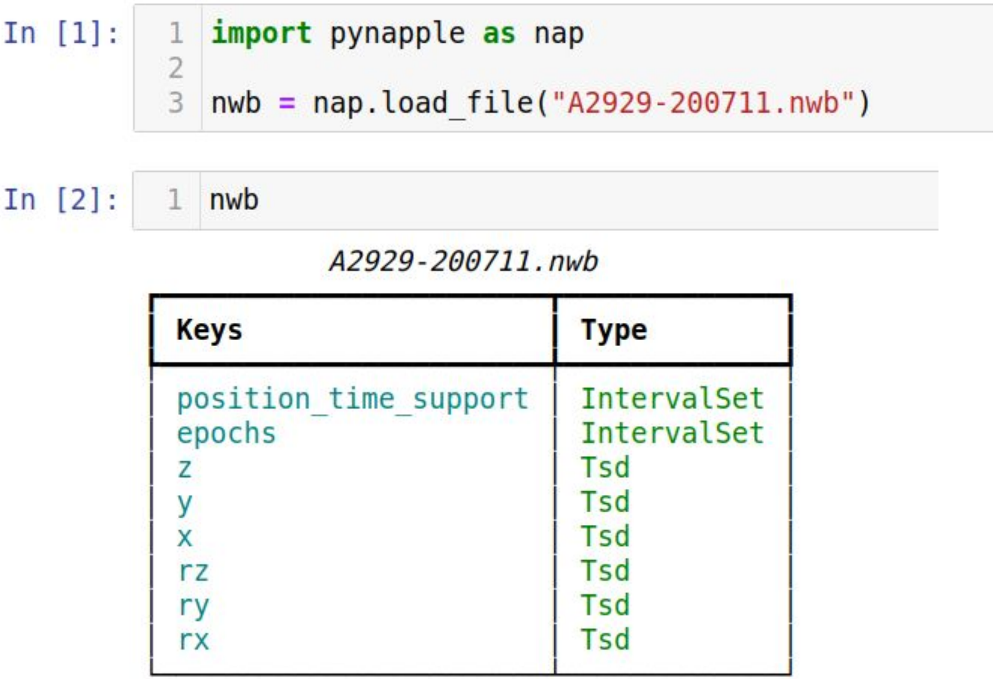
NWB interface



In [1]:

```
import pynapple as nap

nwb = nap.load_file("A2929-200711.nwb")
```

NWB interface



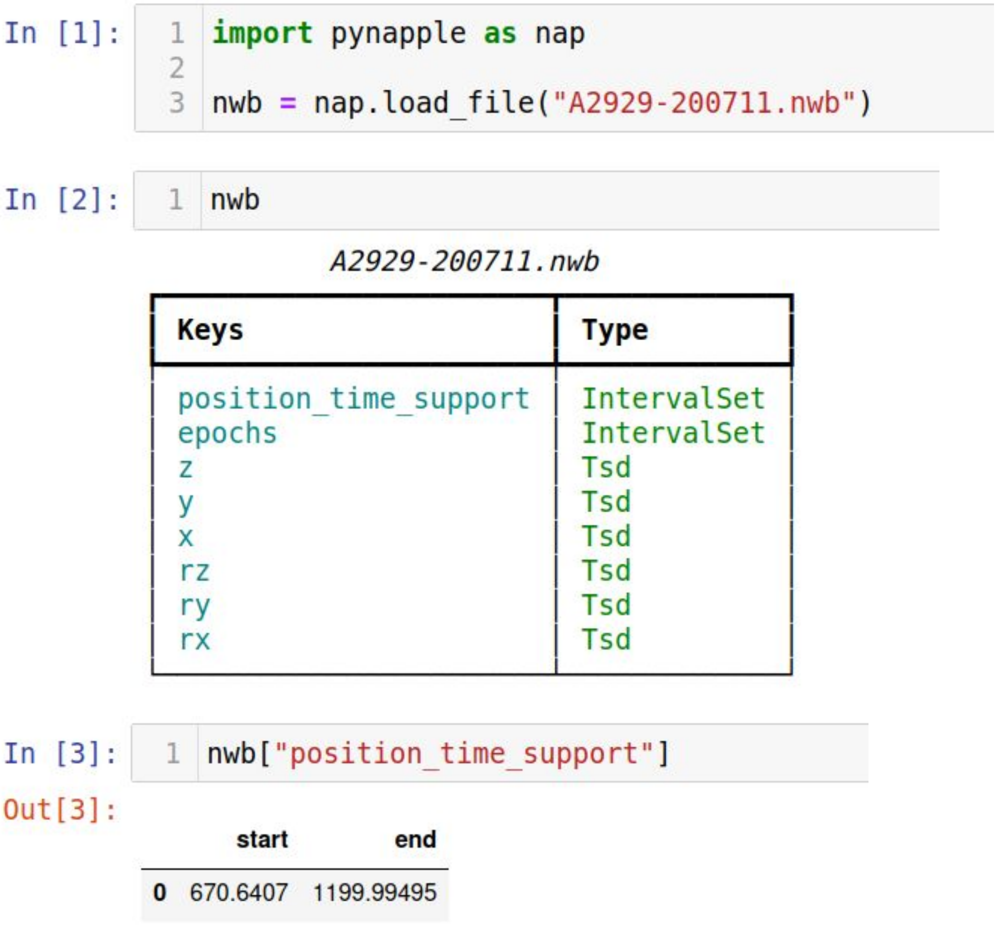
```
In [1]: import pynapple as nap

        nwb = nap.load_file("A2929-200711.nwb")
```

```
In [2]: nwb
```

A2929-200711.nwb

Keys	Type
position_time_support	IntervalSet
epochs	IntervalSet
z	Tsd
y	Tsd
x	Tsd
rz	Tsd
ry	Tsd
rx	Tsd

NWB interface



```
In [1]:
import pynapple as nap

nwb = nap.load_file("A2929-200711.nwb")
```

```
In [2]:
nwb
```

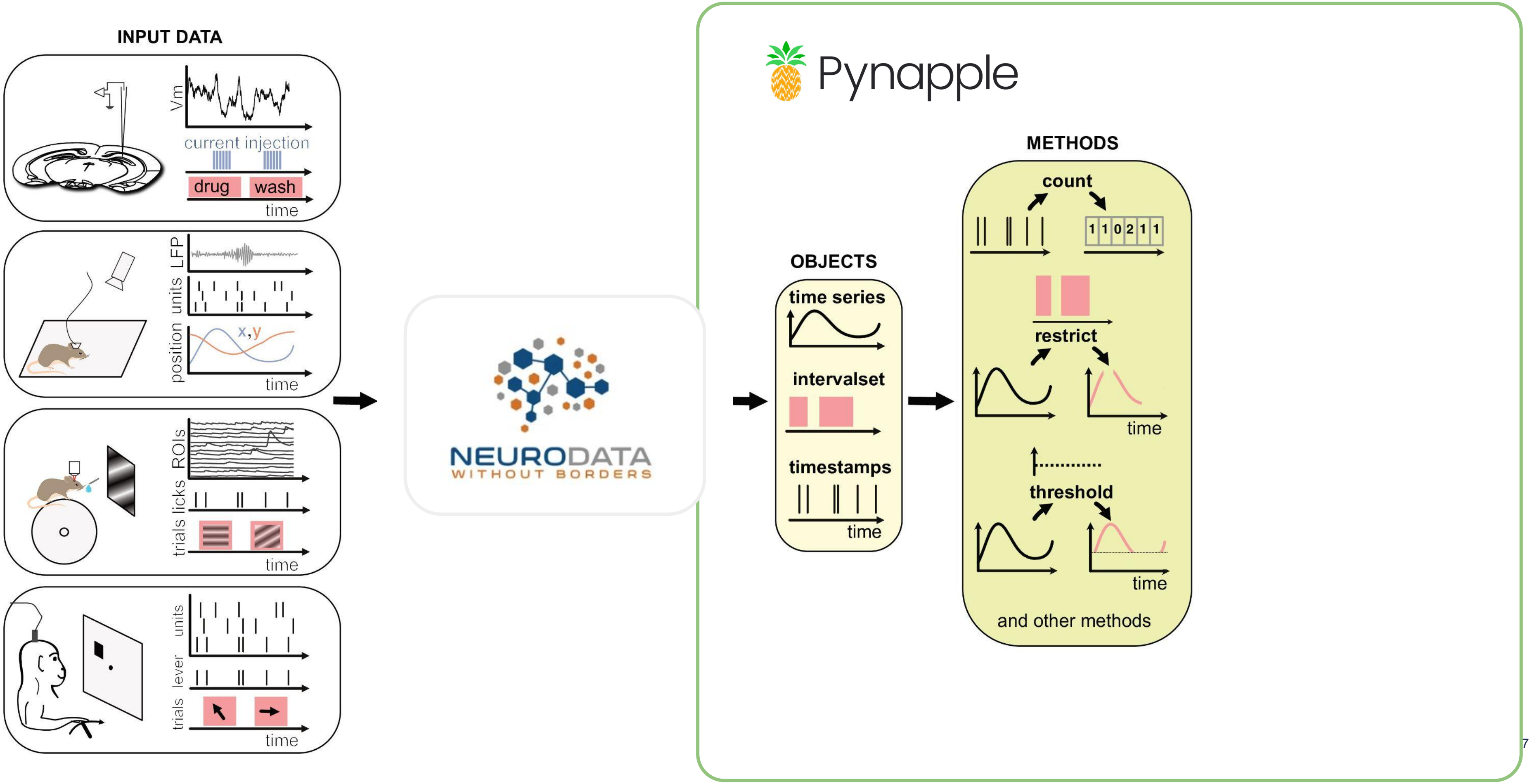
A2929-200711.nwb

Keys	Type
position_time_support	IntervalSet
epochs	IntervalSet
z	Tsd
y	Tsd
x	Tsd
rz	Tsd
ry	Tsd
rx	Tsd

```
In [3]:
nwb["position_time_support"]
```

Out[3]:

	start	end
0	670.6407	1199.99495



INPUT DATA

Pynapple

OBJECTS

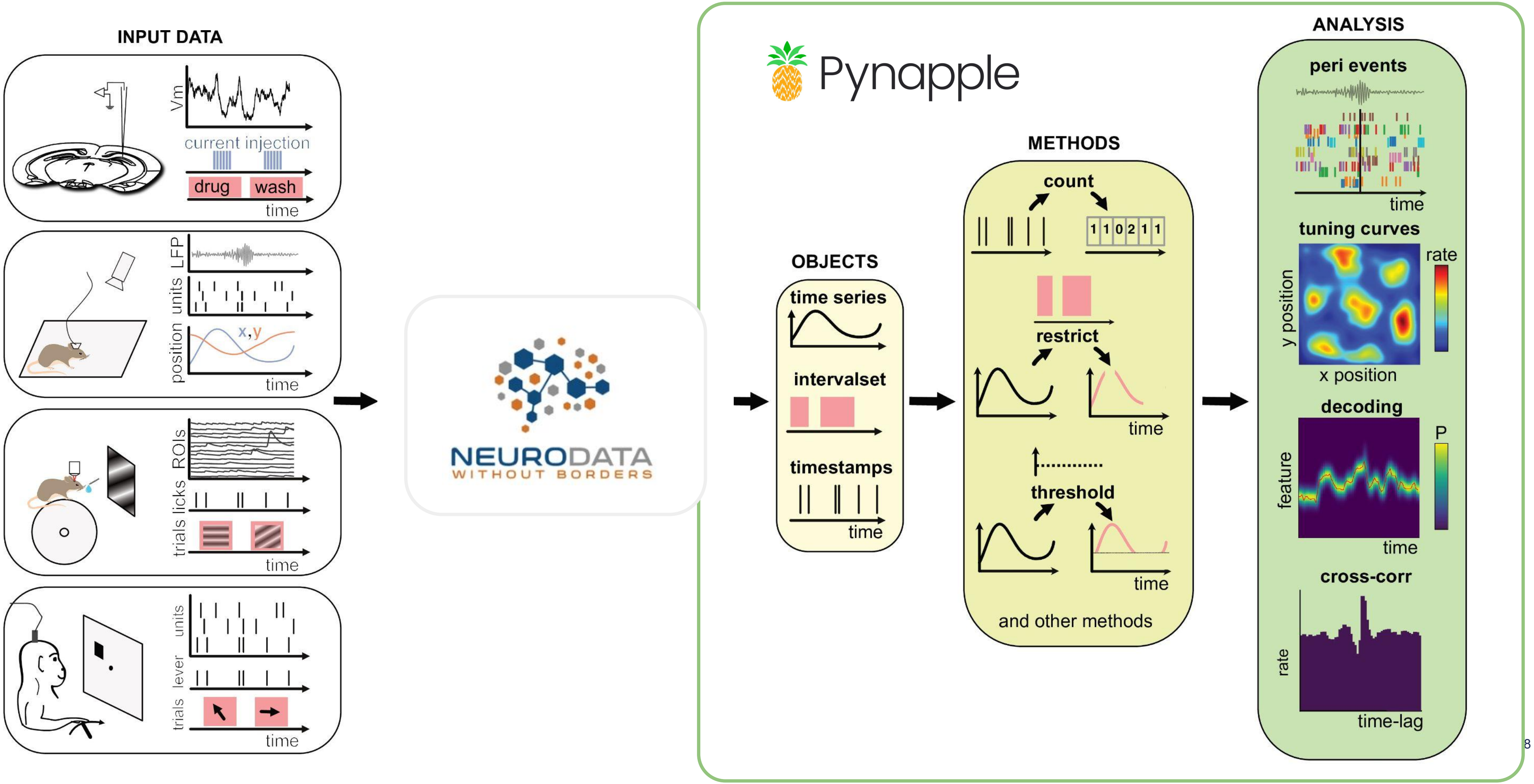
- time series
- intervalset
- timestamps

METHODS

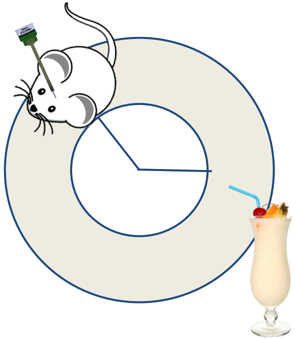
- count
- restrict
- threshold

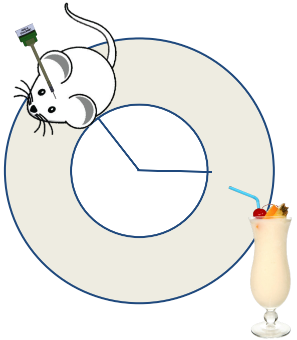
and other methods

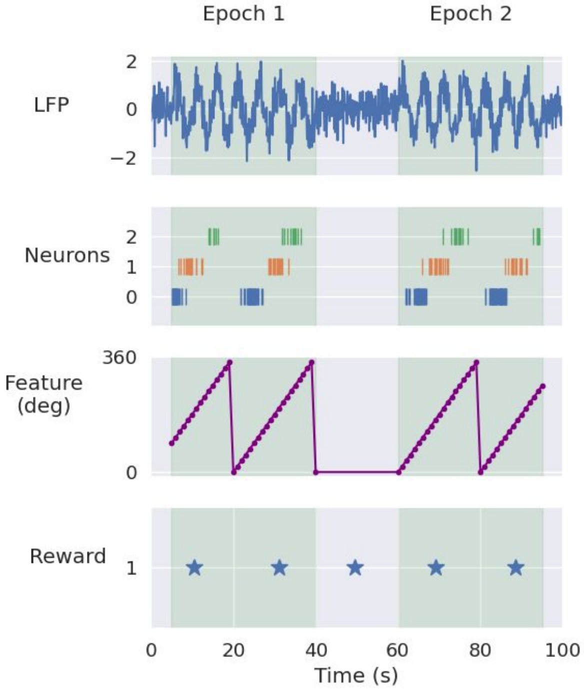
FLATIRON INSTITUTE



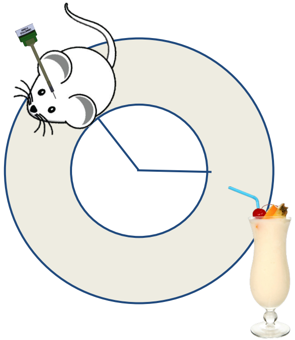
Standard analysis in systems neuroscience

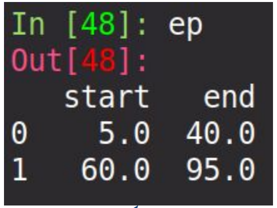




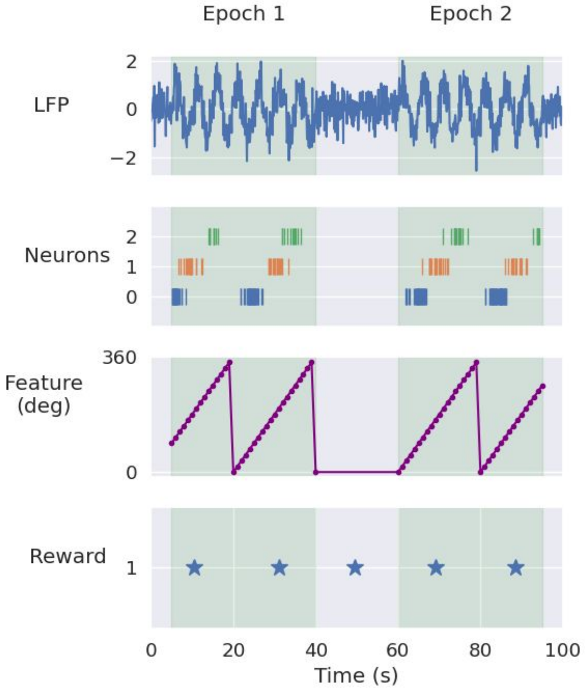

Epoch 1
Epoch 2
LFP
2
0
−2
Neurons
2
1
0
Feature
(deg)
360
0
Reward
1
0
20
40
60
80
100
Time (s)

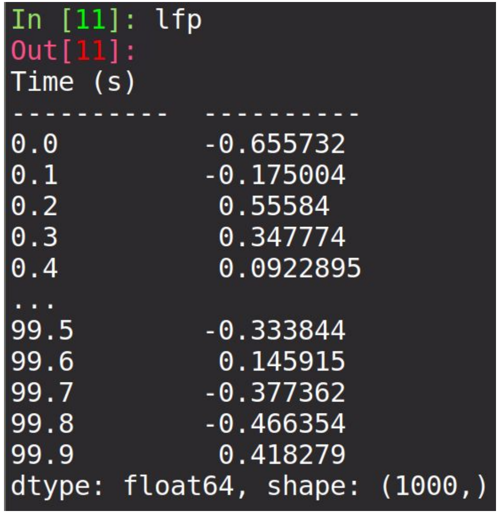
FLATIRON INSTITUTE



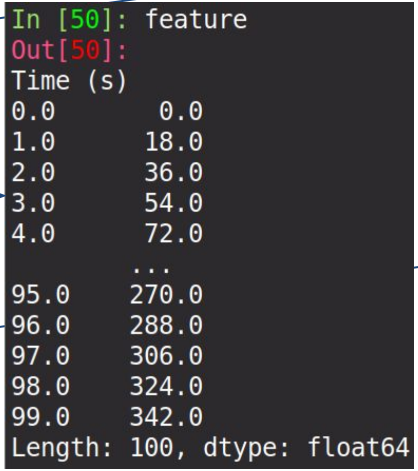


```
In [48]: ep
Out[48]:
   start   end
0    5.0  40.0
1   60.0  95.0
```

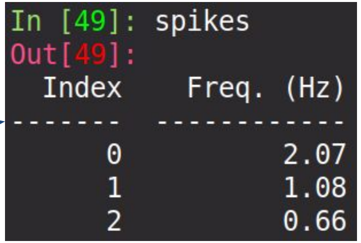




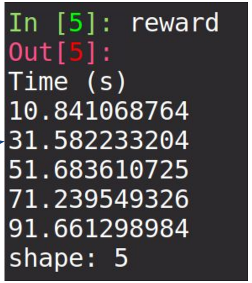
```
In [11]: lfp
Out[11]:
Time (s)
----------  ----------
0.0          -0.655732
0.1          -0.175004
0.2           0.55584
0.3           0.347774
0.4           0.0922895
...
99.5         -0.333844
99.6          0.145915
99.7         -0.377362
99.8         -0.466354
99.9          0.418279
dtype: float64, shape: (1000,)
```



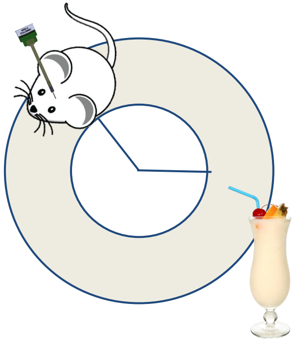
```
In [50]: feature
Out[50]:
Time (s)
0.0       0.0
1.0      18.0
2.0      36.0
3.0      54.0
4.0      72.0
         ...
95.0    270.0
96.0    288.0
97.0    306.0
98.0    324.0
99.0    342.0
Length: 100, dtype: float64
```

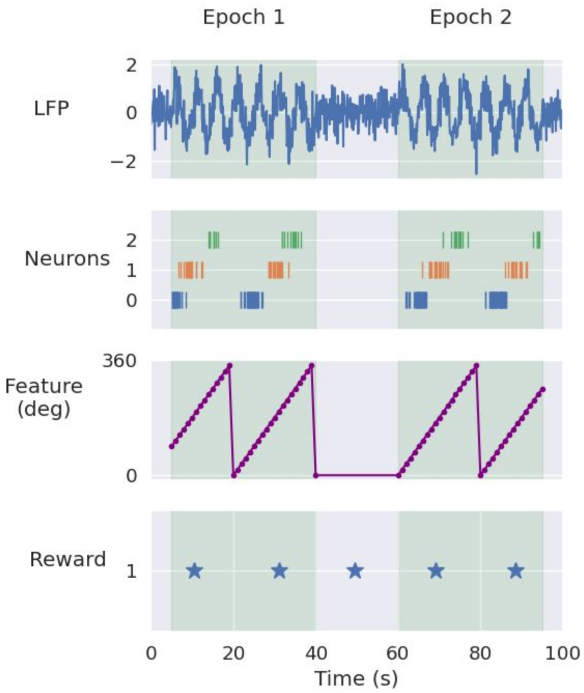


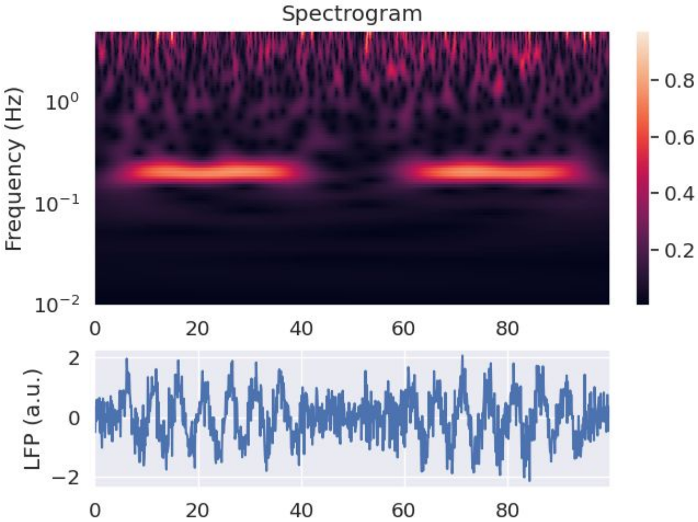
```
In [49]: spikes
Out[49]:
  Index      Freq. (Hz)
-------    ------------
      0            2.07
      1            1.08
      2            0.66
```

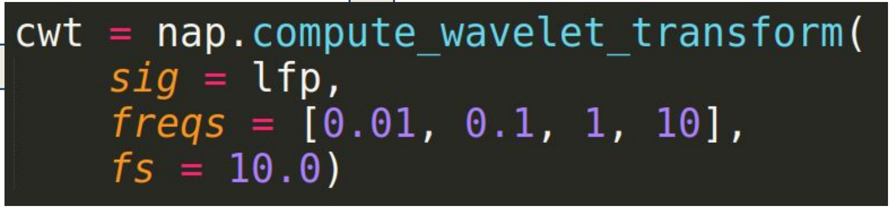


```
In [5]: reward
Out[5]:
Time (s)
10.841068764
31.582233204
51.683610725
71.239549326
91.661298984
shape: 5
```

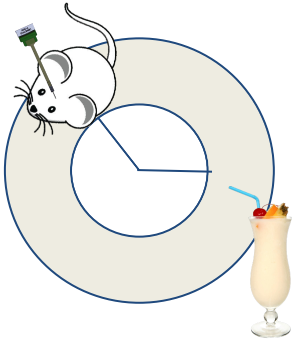


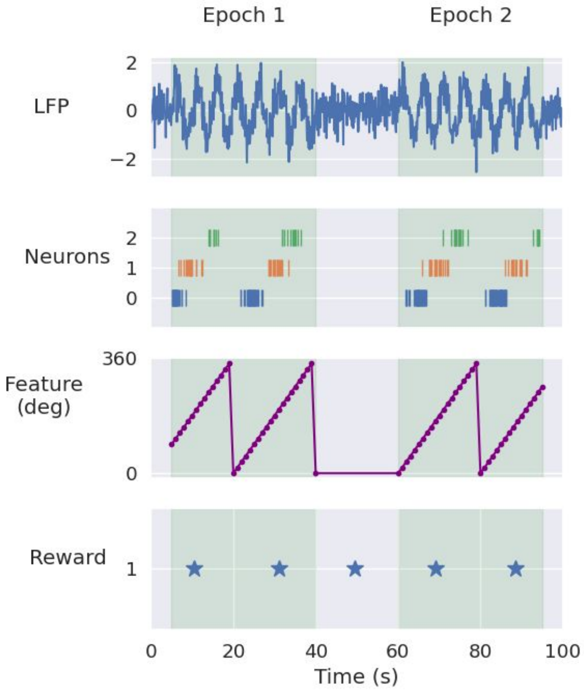


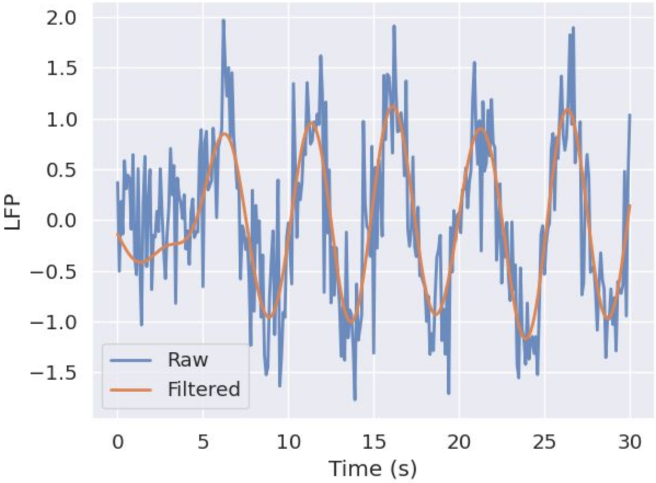


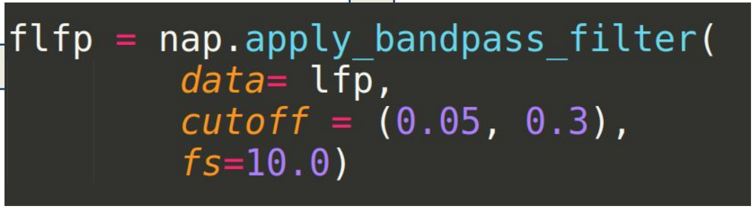


```
cwt = nap.compute_wavelet_transform(
    sig = lfp,
    freqs = [0.01, 0.1, 1, 10],
    fs = 10.0)
```

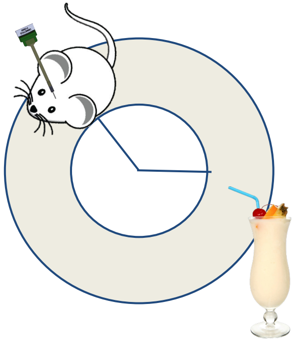


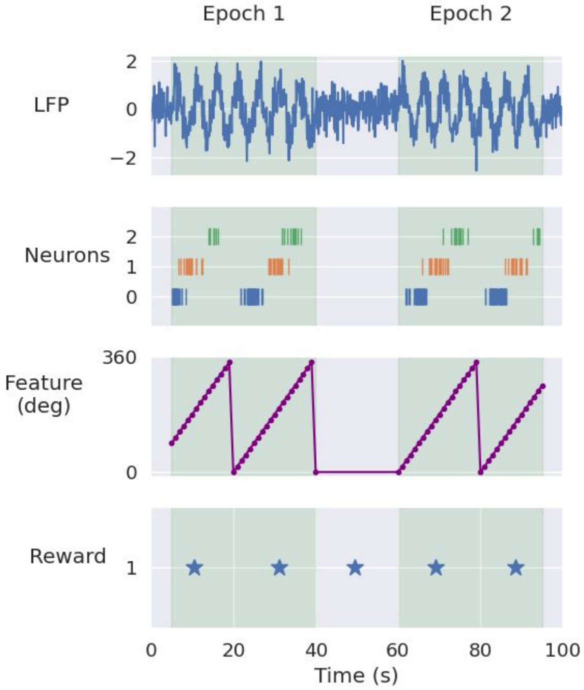


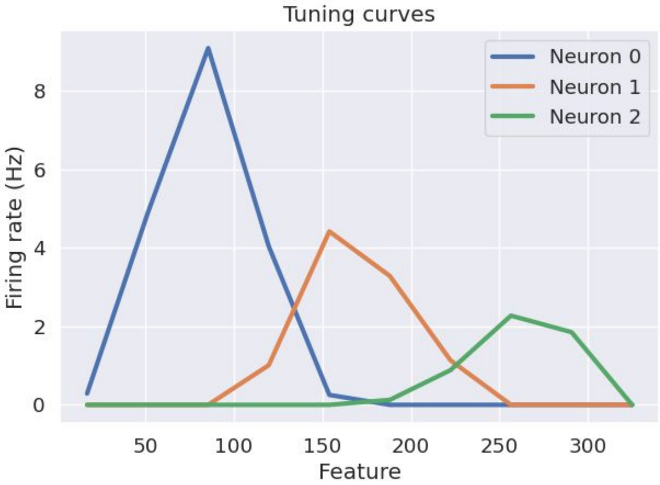


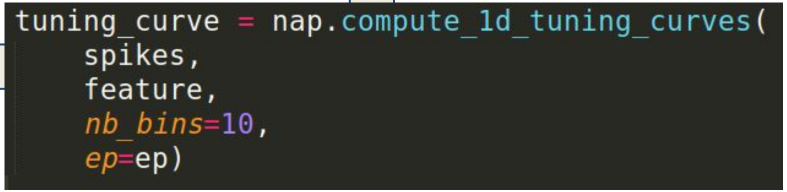


```
flfp = nap.apply_bandpass_filter(
          data= lfp,
          cutoff = (0.05, 0.3),
          fs=10.0)
```

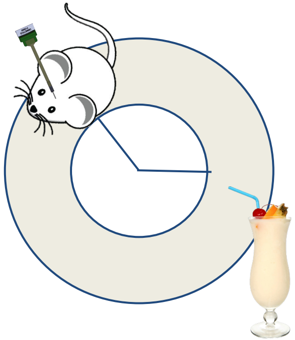


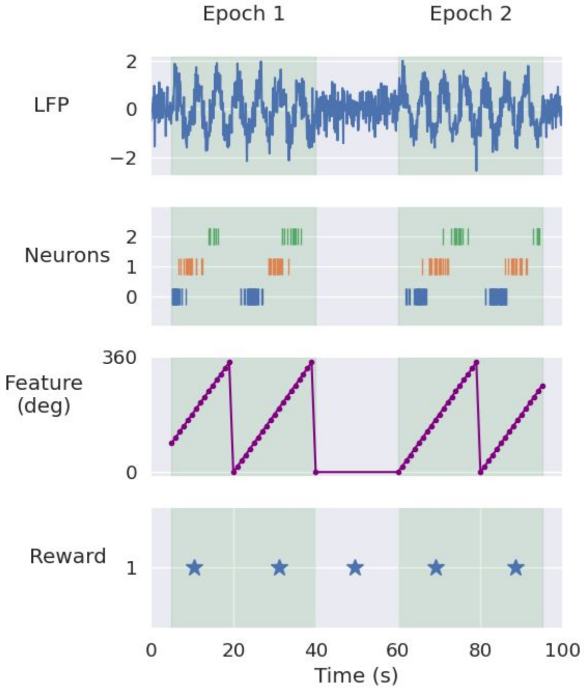


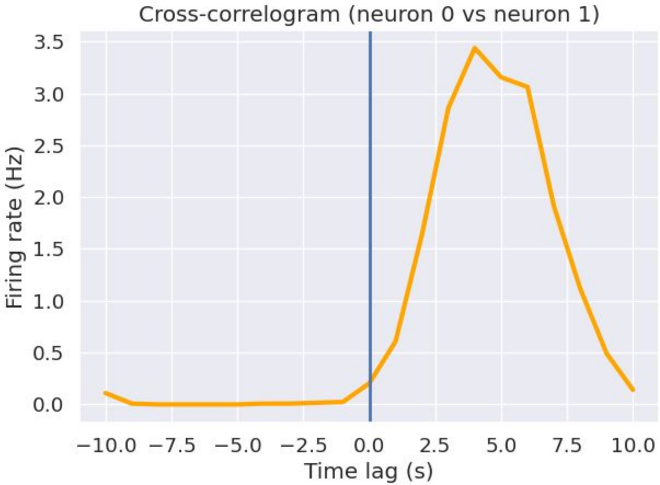


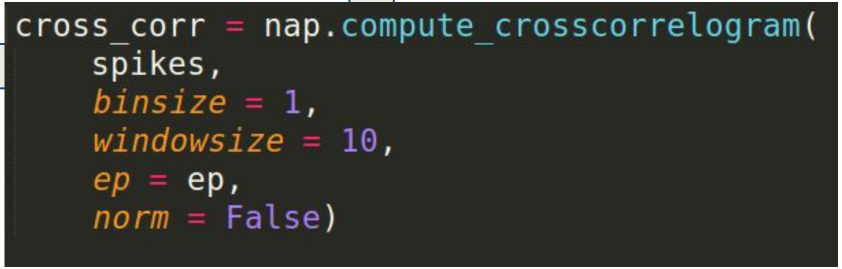


```
tuning_curve = nap.compute_1d_tuning_curves(
    spikes,
    feature,
    nb_bins=10,
    ep=ep)
```

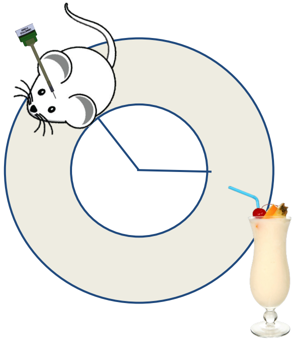


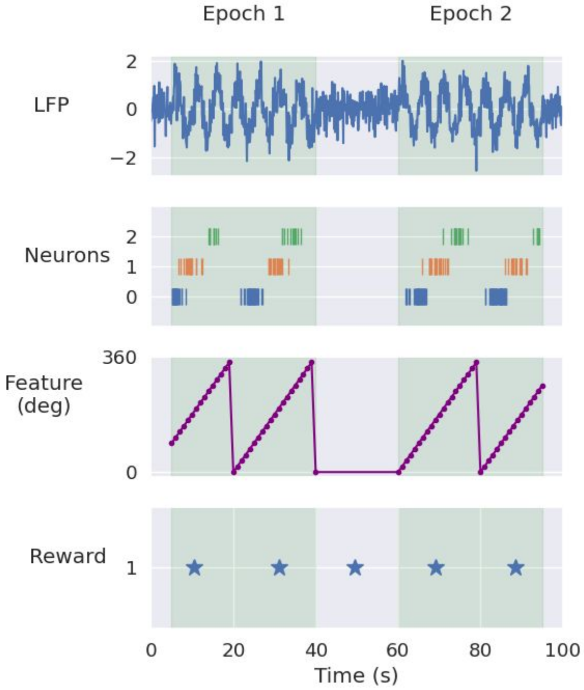


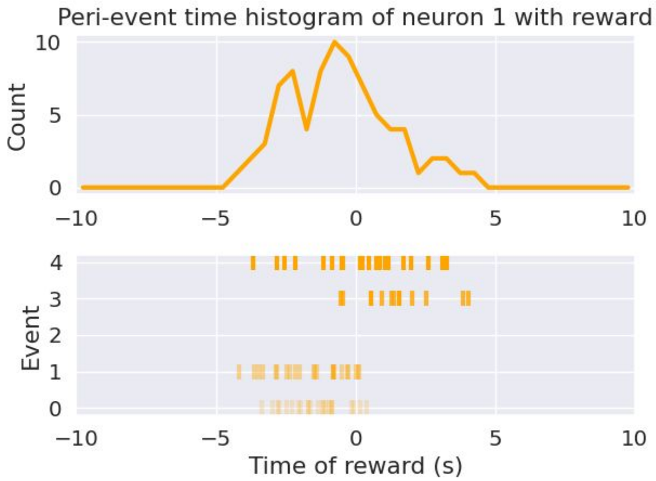


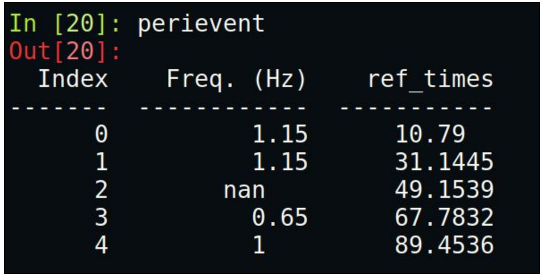


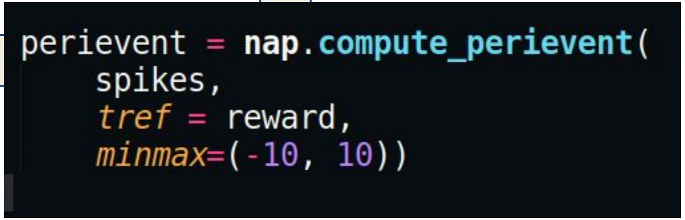
```
cross_corr = nap.compute_crosscorrelogram(
    spikes,
    binsize = 1,
    windowsize = 10,
    ep = ep,
    norm = False)
```

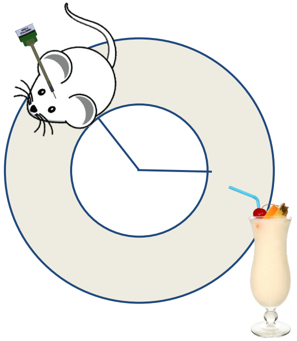


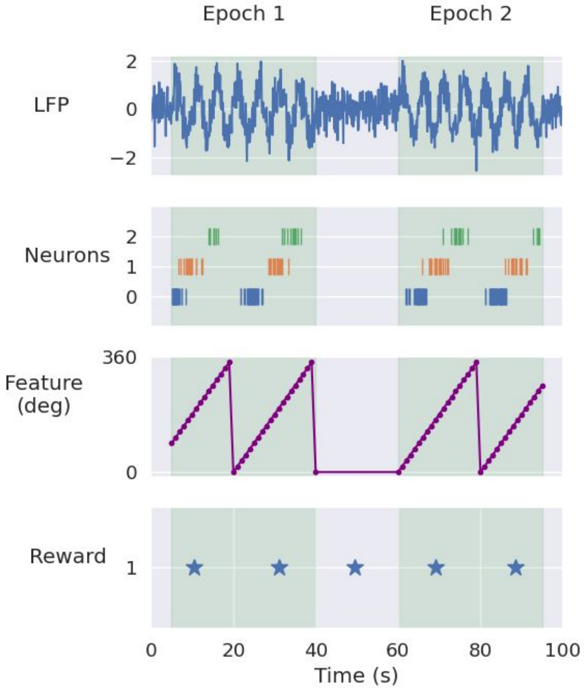


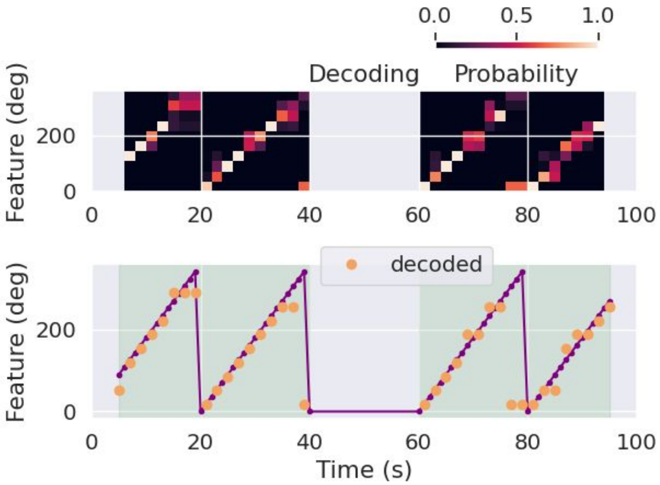


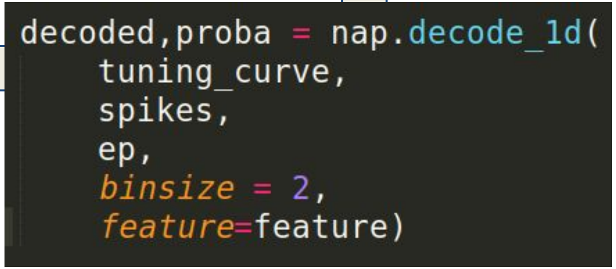

```
In [20]: perievent
Out[20]:
  Index     Freq. (Hz)    ref_times
-------  ------------  -----------
      0          1.15      10.79
      1          1.15      31.1445
      2        nan         49.1539
      3          0.65      67.7832
      4          1        89.4536
```


```
perievent = nap.compute_perievent(
    spikes,
    tref = reward,
    minmax=(-10, 10))
```



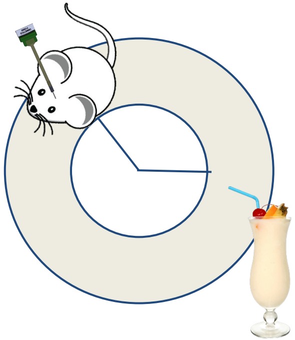


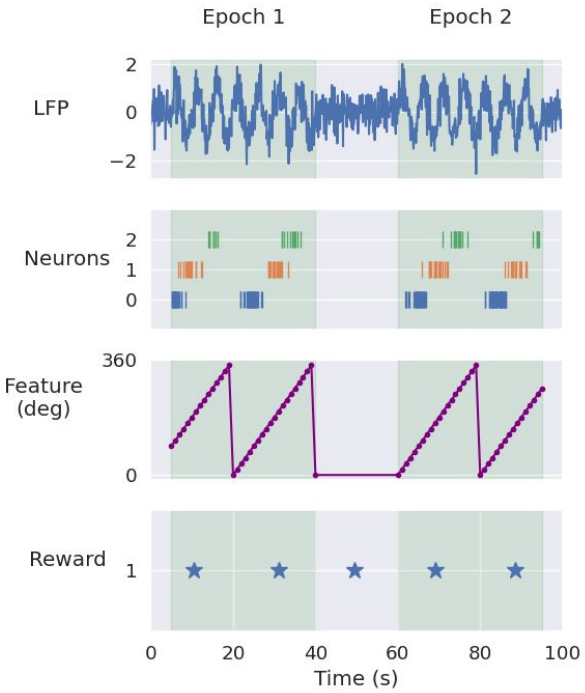


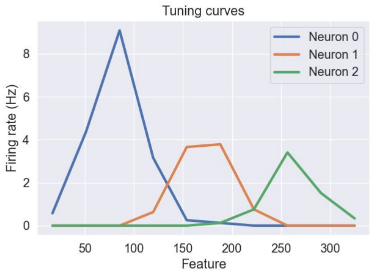


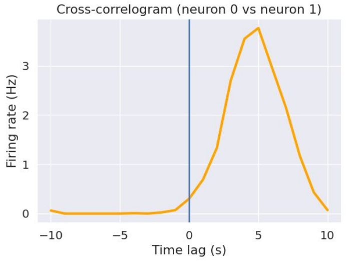
```
decoded,proba = nap.decode_1d(
    tuning_curve,
    spikes,
    ep,
    binsize = 2,
    feature=feature)
```

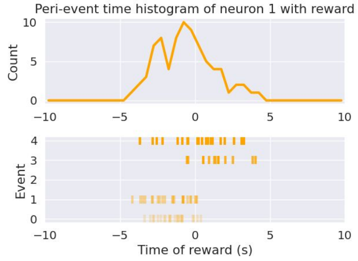
FLATIRON INSTITUTE

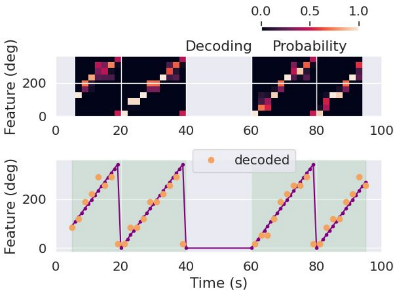


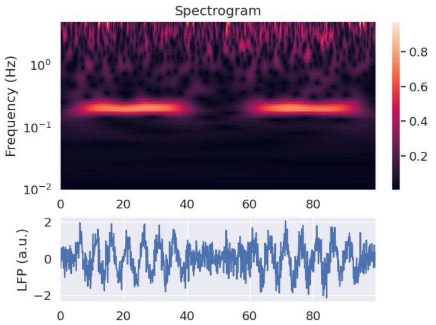


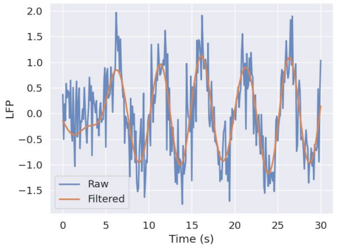


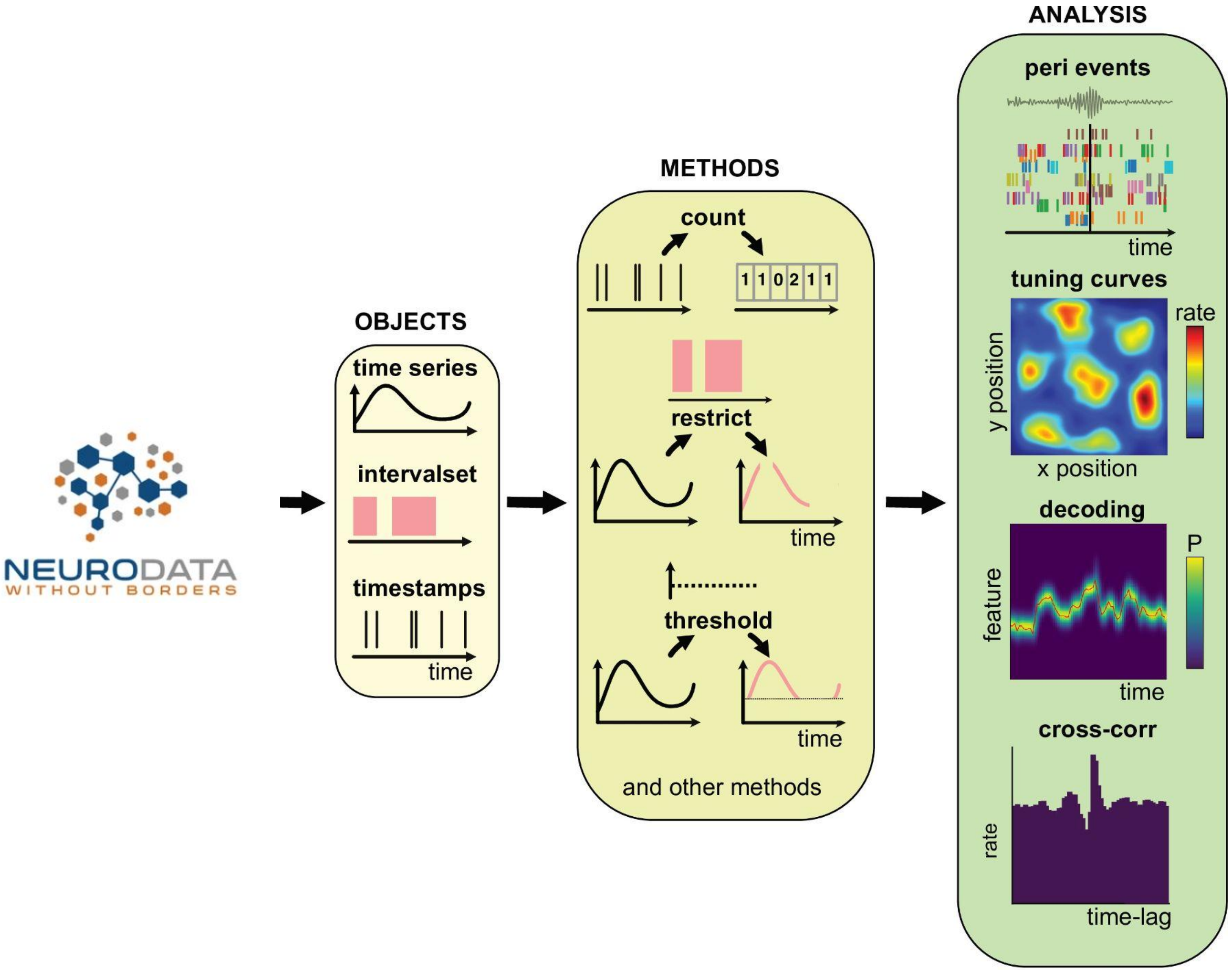










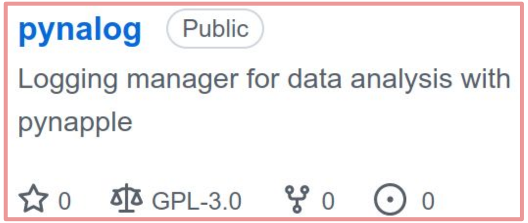

NEURODATA
WITHOUT BORDERS
OBJECTS
time series
intervalset
timestamps
time
METHODS
count
1 1 0 2 1 1
restrict
time
threshold
time
and other methods
ANALYSIS
peri events
time
tuning curves
rate
y position
x position
decoding
P
feature
time
cross-corr
rate
time-lag

Future developments

FLATIRON INSTITUTE

Pynapple

Pynasuite

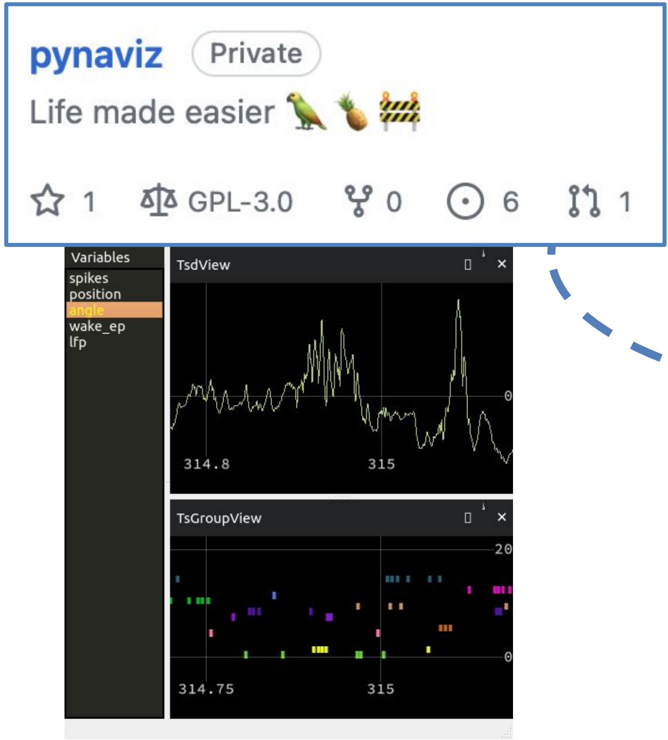


pynalog Public

Logging manager for data analysis with pynapple

☆ 0 GPL-3.0 0 0

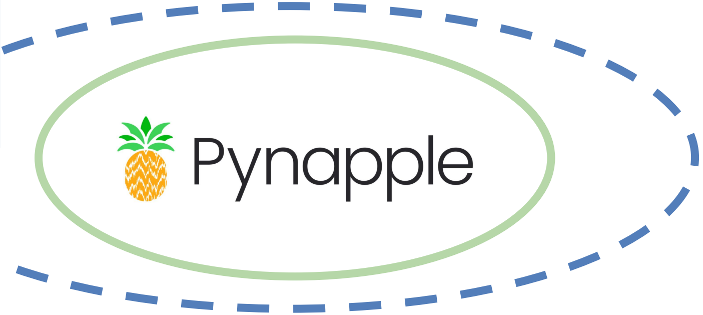
Pynasuite

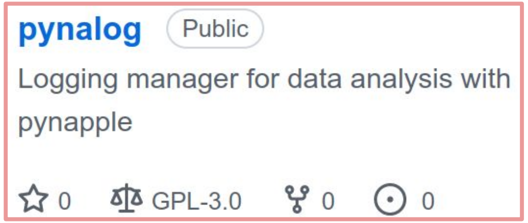


pynaviz Private

Life made easier 🦜🍍🚧

☆ 1 ⚖ GPL-3.0 ⑂ 0 ⊙ 6 ⇅ 1

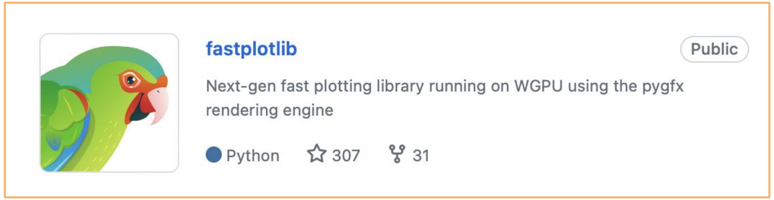




pynalog Public

Logging manager for data analysis with pynapple

☆ 0 ⚖ GPL-3.0 ⑂ 0 ⊙ 0



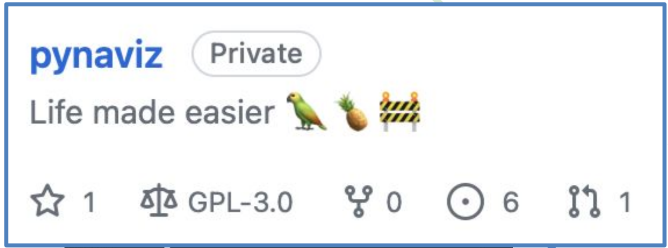
fastplotlib Public

Next-gen fast plotting library running on WGPU using the pygfx rendering engine

● Python ☆ 307 31

Kushal Kolar

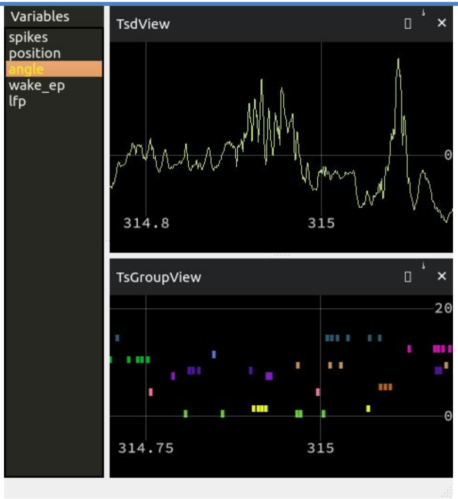
Caitlin Lewis



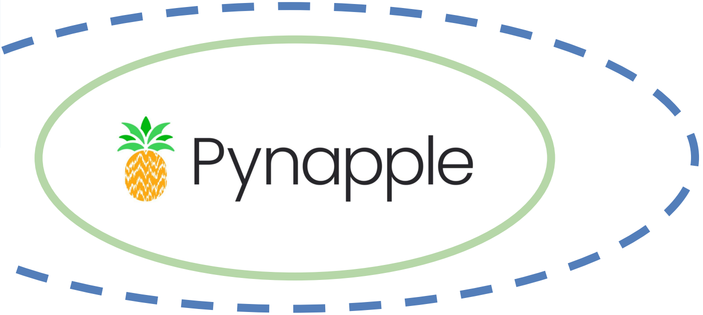
pynaviz Private

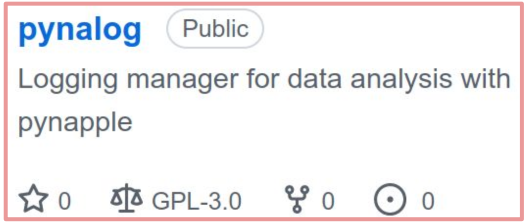
Life made easier 🦜 🍍 🚧

☆ 1 GPL-3.0 0 6 1



Pynasuite

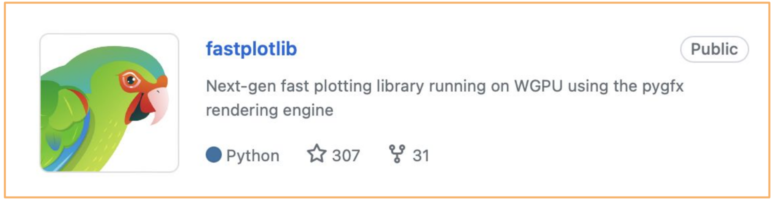




pynalog Public

Logging manager for data analysis with pynapple

☆ 0 GPL-3.0 0 0



fastplotlib Public

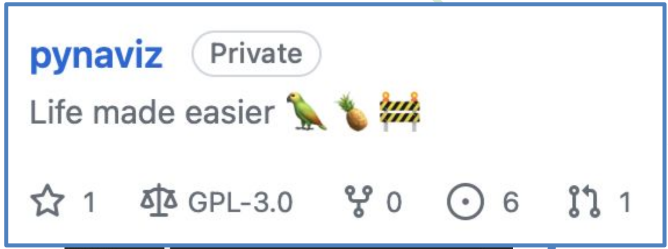
Next-gen fast plotting library running on WGPU using the pygfx rendering engine

● Python ☆ 307 ⑂ 31

Kushal Kolar

Caitlin Lewis

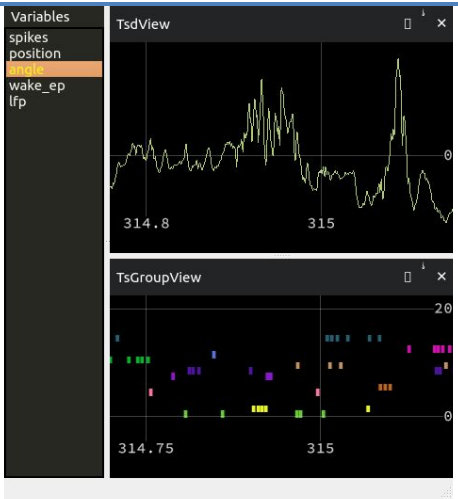
Pynasuite



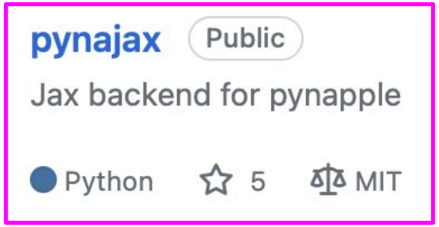
pynaviz Private

Life made easier 🦜 🍍 🚧

☆ 1 ⚖ GPL-3.0 ⑂ 0 ⊙ 6 ⇅ 1



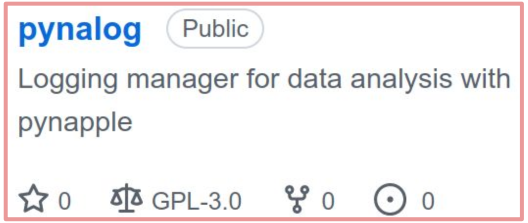
Pynapple



pynajax Public

Jax backend for pynapple

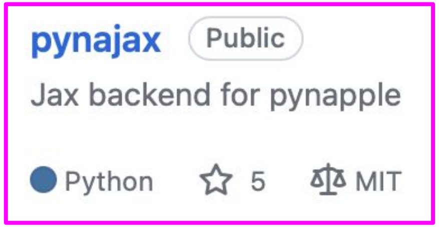
● Python ☆ 5 ⚖ MIT



pynalog Public

Logging manager for data analysis with pynapple

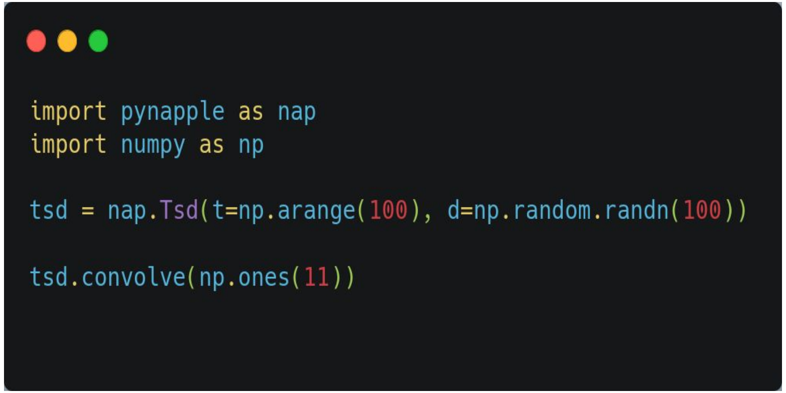
☆ 0 ⚖ GPL-3.0 ⑂ 0 ⊙ 0



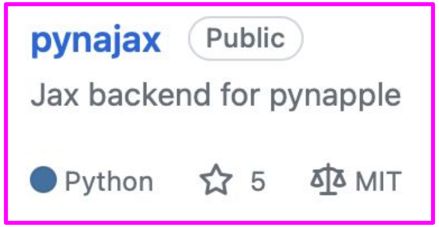
pynajax Public

Jax backend for pynapple

● Python ☆ 5 MIT



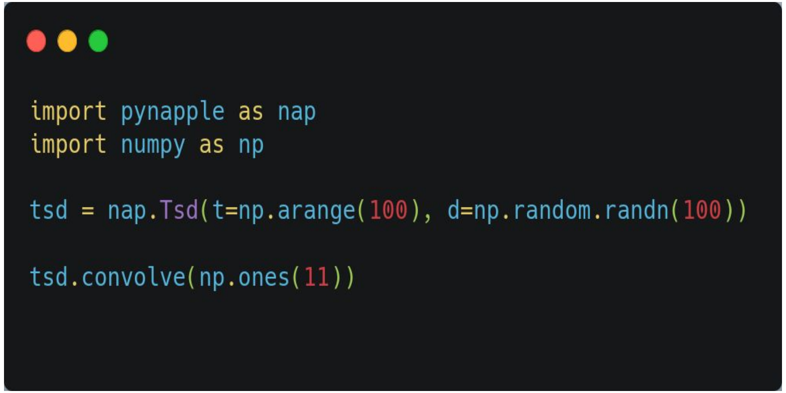
```
import pynapple as nap
import numpy as np

tsd = nap.Tsd(t=np.arange(100), d=np.random.randn(100))

tsd.convolve(np.ones(11))
```



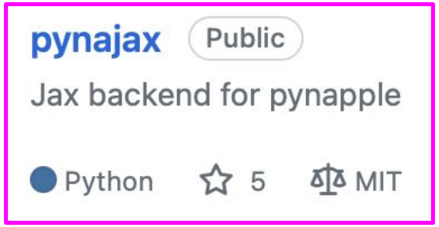
pynajax Public

Jax backend for pynapple

● Python ☆ 5 ⚖ MIT



```
import pynapple as nap
import numpy as np

tsd = nap.Tsd(t=np.arange(100), d=np.random.randn(100))

tsd.convolve(np.ones(11))
```

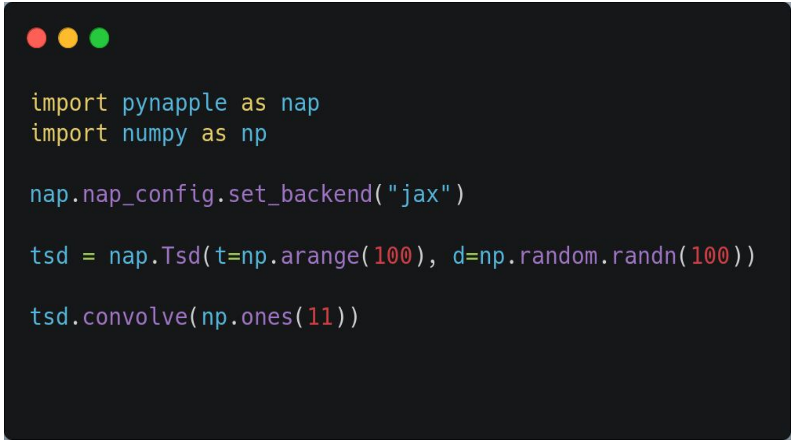


pynajax Public

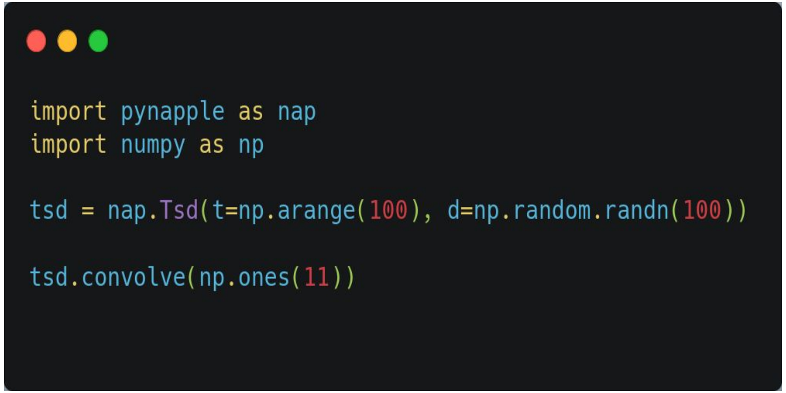
Jax backend for pynapple

● Python ☆ 5 MIT

$ pip install pynajax

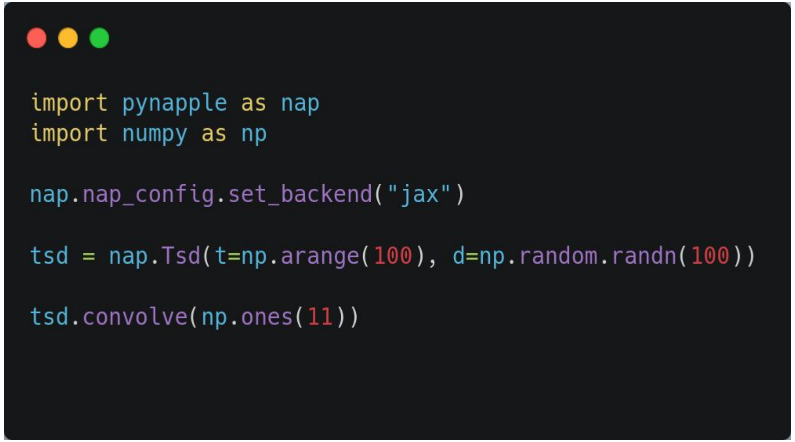


```
import pynapple as nap
import numpy as np

nap.nap_config.set_backend("jax")

tsd = nap.Tsd(t=np.arange(100), d=np.random.randn(100))

tsd.convolve(np.ones(11))
```

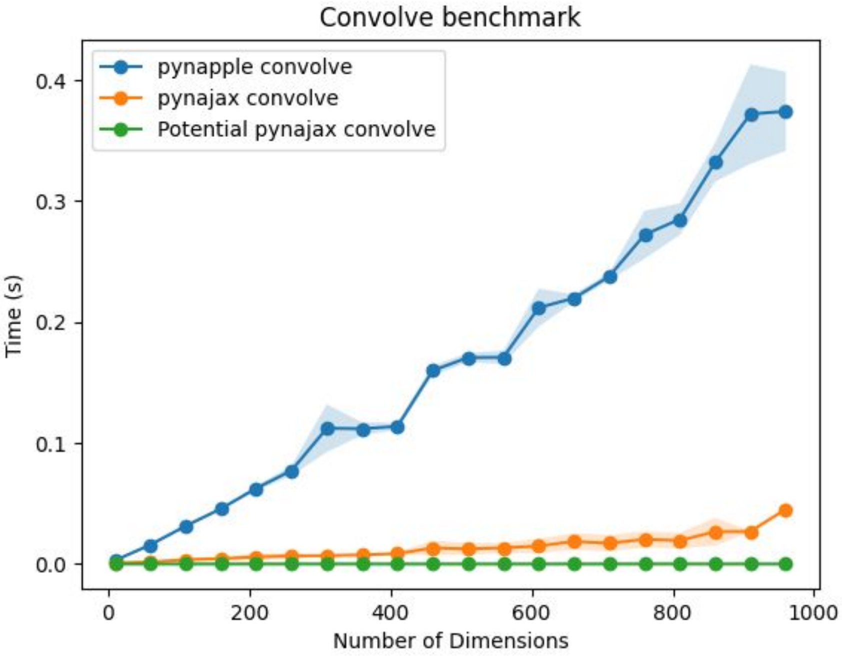


```python
import pynapple as nap
import numpy as np

tsd = nap.Tsd(t=np.arange(100), d=np.random.randn(100))

tsd.convolve(np.ones(11))
```

$ pip install pynajax



```python
import pynapple as nap
import numpy as np

nap.nap_config.set_backend("jax")

tsd = nap.Tsd(t=np.arange(100), d=np.random.randn(100))

tsd.convolve(np.ones(11))
```



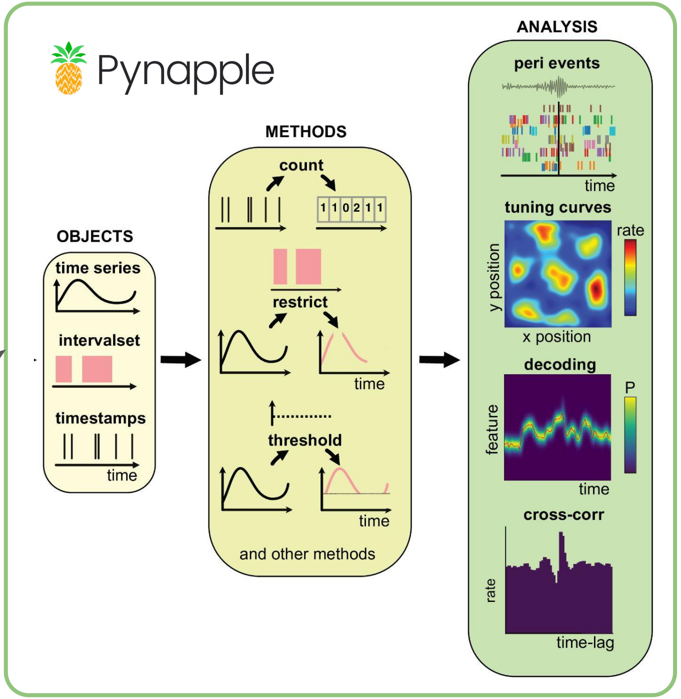
FLATIRON INSTITUTE

Time

Preprocessing
(CaImAn, SpikeInterface, …)

Pynapple

Postprocessing
(GLM, Manifold, …)



FLATIRON INSTITUTE

Pynapple

```
$ pip install pynapple
```

https://twitter.com/thepynapple

https://bsky.app/profile/pynapple.bsky.social

pynapple-org.slack.com

Pynapple

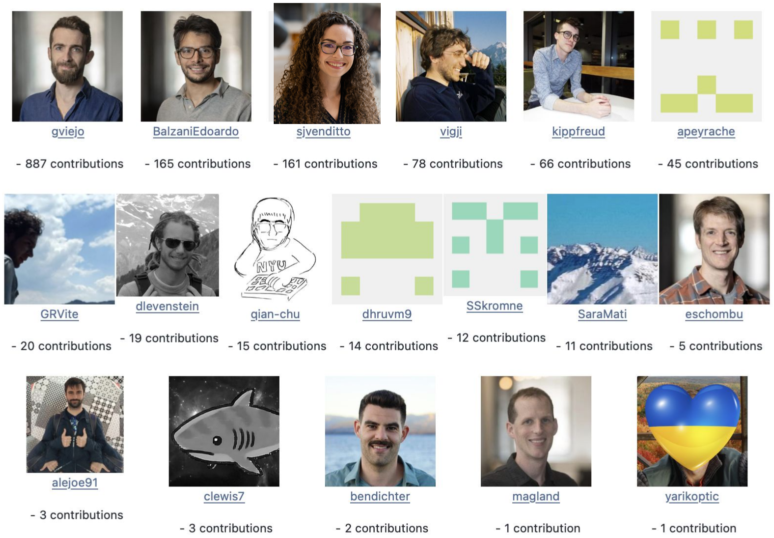
```
$ pip install pynapple
```

https://twitter.com/thepynapple

https://bsky.app/profile/pynapple.bsky.social

pynapple-org.slack.com

Contributors



gviejo - 887 contributions

BalzaniEdoardo - 165 contributions

sjvenditto - 161 contributions

vigji - 78 contributions

kippfreud - 66 contributions

apeyrache - 45 contributions

GRVite - 20 contributions

dlevenstein - 19 contributions

gian-chu - 15 contributions

dhruvm9 - 14 contributions

SSkromne - 12 contributions

SaraMati - 11 contributions

eschombu - 5 contributions

alejoe91 - 3 contributions

clewis7 - 3 contributions

bendichter - 2 contributions

magland - 1 contribution

yarikoptic - 1 contribution

Time for ~~coding~~ lunch!

FLATIRON INSTITUTE