Pynapple : Python Neural Analysis package

FENS 2024 workshop

Guillaume Viejo

FLATIRON INSTITUTE

When do I need pynapple?

Time

Preprocessing
(CaImAn, SpikeInterface, …)

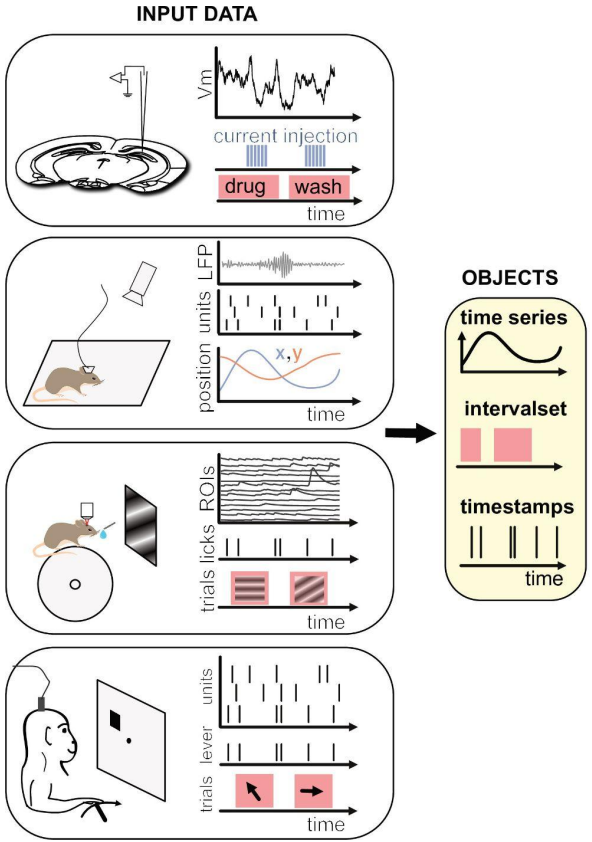
Postprocessing
(GLM, Manifold, …)

Time

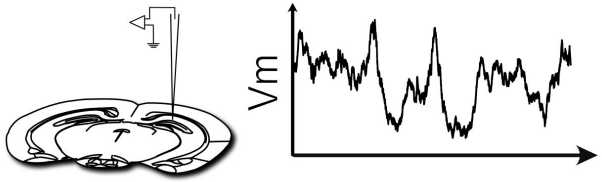
Preprocessing
(CaImAn, SpikeInterface, …)

Pynapple

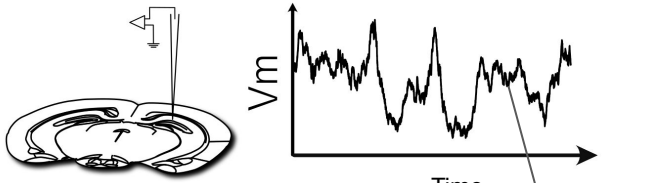
Postprocessing
(GLM, Manifold, …)



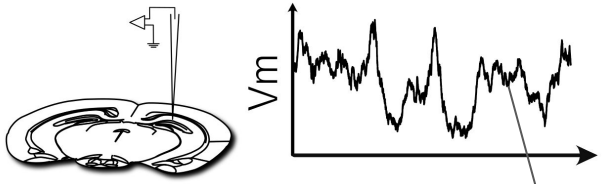
Time Series Data : Tsd, TsdFrame and TsdTensor



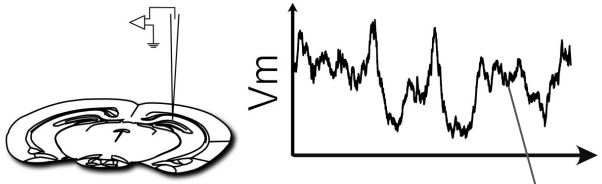
FLATIRON INSTITUTE




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



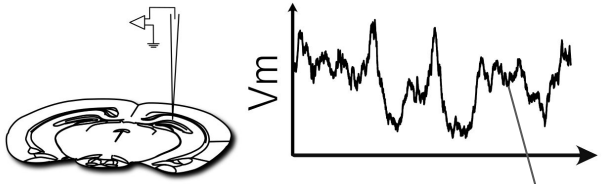

```
In [11]: tsd = nap.Tsd(t=t, d=d)

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,)
```



Numpy array

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

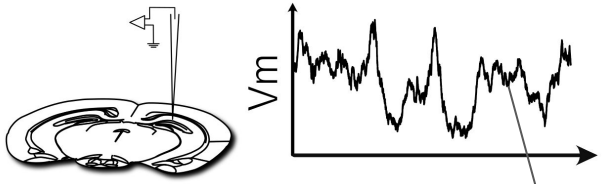


Numpy array

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

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



Numpy array

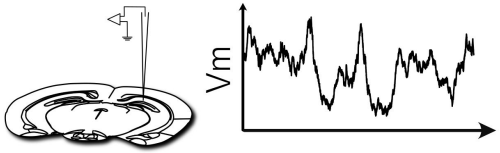
```
In [15]: tsd.index.values
Out[15]:
array([ 0.,  1.,  2.,  3.,
```

```
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

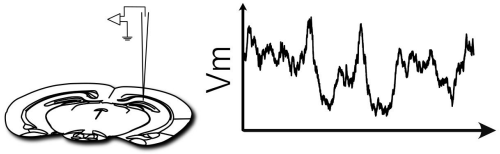
- Automatic handling of time units
- Epoch restriction
- Binning
- Time support
- …

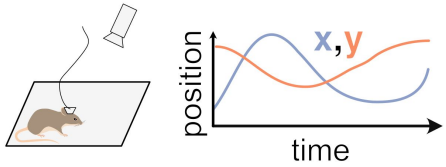


FLATIRON INSTITUTE

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,)
```





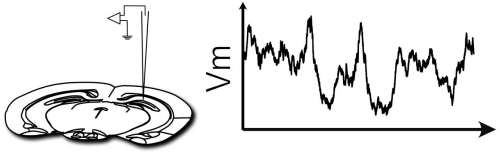
FLATIRON INSTITUTE

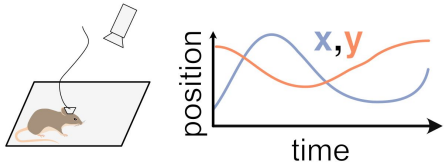
Tsd: 1-dimension

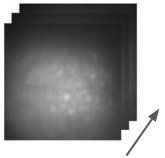
```
In [11]: tsd = nap.Tsd(t=t, d=d)

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)
```







FLATIRON INSTITUTE

Tsd: 1-dimension

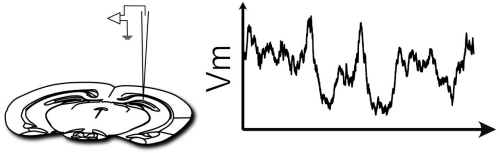
```
In [11]: tsd = nap.Tsd(t=t, d=d)

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,)
```

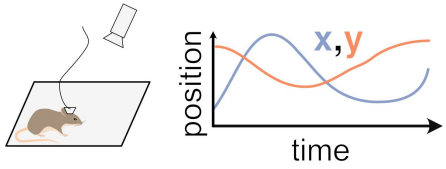
TsdFrame: 2-dimensions

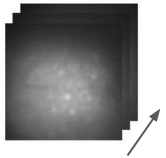
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:                         columns = ['x', 'y'])

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)
```







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)
```

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

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

pandas

Doing math: the numpy array container approach

Numpy array

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

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

FLATIRON INSTITUTE

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)
```

FLATIRON INSTITUTE

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]])
```

```
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)
```

FLATIRON INSTITUTE

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,)
```

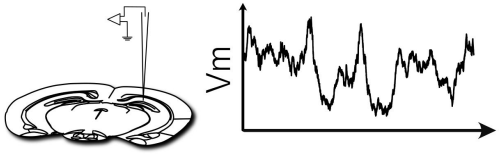
```
In [17]: np.concatenate((tsd1, tsd2))
Out[17]:
Time (s)
----------  ----------
0            0.910503
1           -0.0110368
2            0.496159
3           -0.956014
4           -0.592748
5           -0.711171
6           -0.370642
7            0.424684
8            0.718995
9            0.616419
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.
```

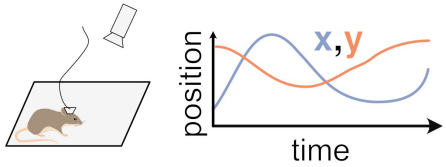
FLATIRON INSTITUTE

Time series without data : the timestamps object



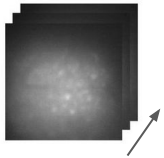
Tsd: 1-dimension

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



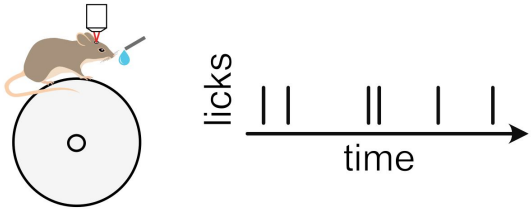
TsdFrame: 2-dimensions

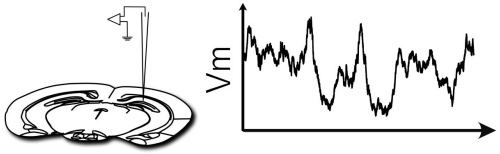
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:                         columns = ['x', 'y'])
```



TsdTensor: n-dimensions

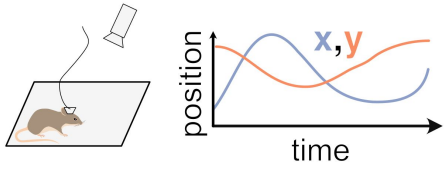
```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)
```





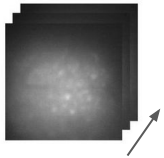
Tsd: 1-dimension

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



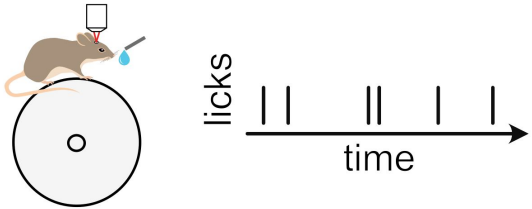
TsdFrame: 2-dimensions

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



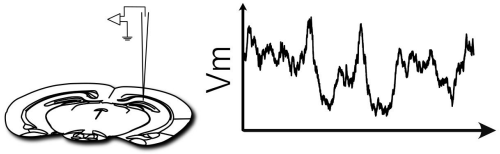
TsdTensor: n-dimensions

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



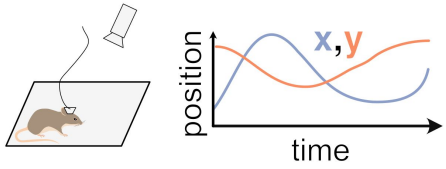
Ts: Timestamps

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



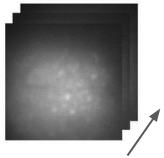
Tsd: 1-dimension

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



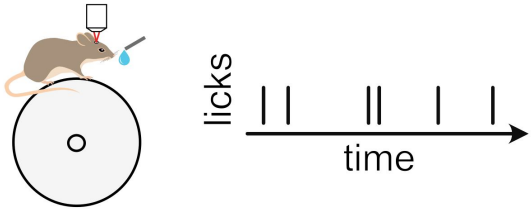
TsdFrame: 2-dimensions

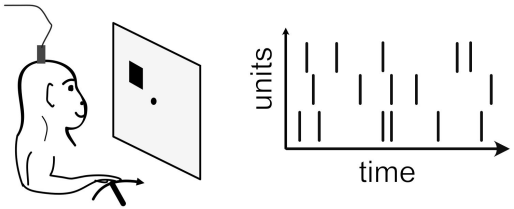
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:                         columns = ['x', 'y'])
```



TsdTensor: n-dimensions

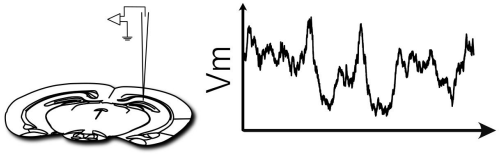
```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)
```





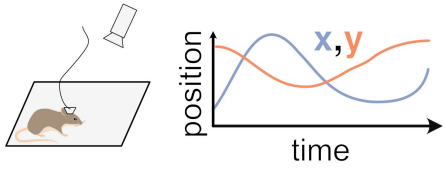
Ts: Timestamps

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



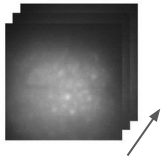
Tsd: 1-dimension

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



TsdFrame: 2-dimensions

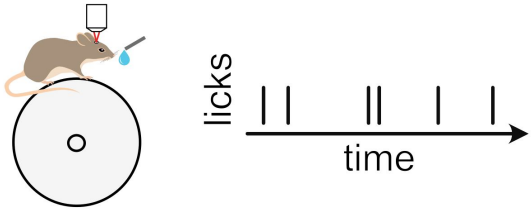
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:                         columns = ['x', 'y'])
```



TsdTensor: n-dimensions

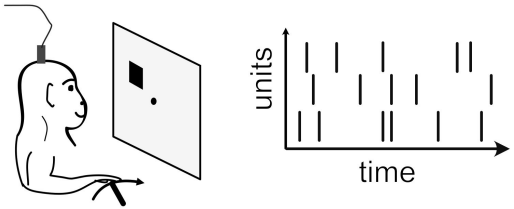
```
In [34]: tsdtensor = nap.TsdTensor(t=t, d=d)
```

FLATIRON INSTITUTE



Ts: Timestamps

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

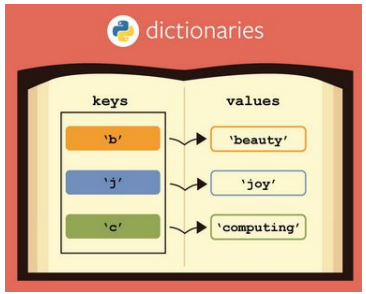


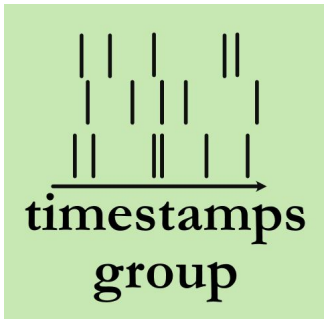
TsGroup: group of timestamps

```
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

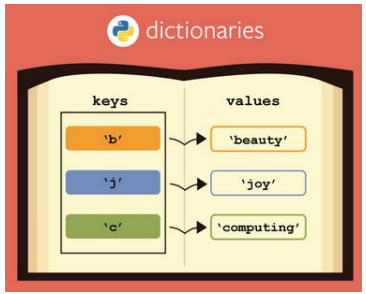
TsGroup manipulation

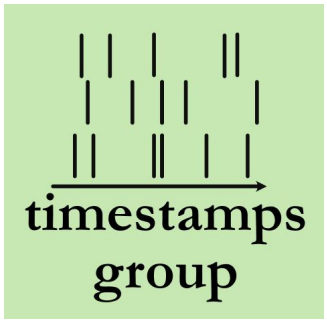




TsGroup manipulation

```
ts_group = nap.TsGroup(
    data = {
        0: neuron_thalamus,
        1: neuron_ca1,
        2: neuron_cerebellum
        },
    group = pd.Series(data=['thalamus','ca1','cerebellum']),
    theta_modulation = pd.Series(data=[1,1,0])
    )
```

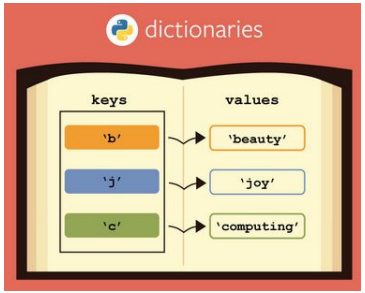


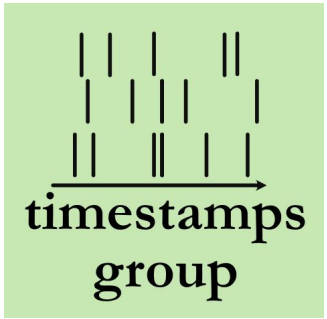


TsGroup manipulation

```
ts_group = nap.TsGroup(
    data = {
        0: neuron_thalamus,
        1: neuron_ca1,
        2: neuron_cerebellum
        },
    group = pd.Series(data=['thalamus','ca1','cerebellum']),
    theta_modulation = pd.Series(data=[1,1,0])
    )
```

```
In [7]: ts_group
Out[7]:
  Index    Freq. (Hz)  group         theta_modulation
-------  ------------  ----------  ------------------
      0           0.5  thalamus                     1
      1             1  ca1                          1
      2         10.01  cerebellum                   0
```

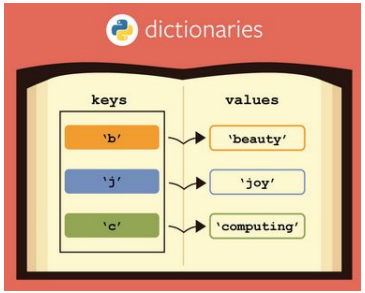


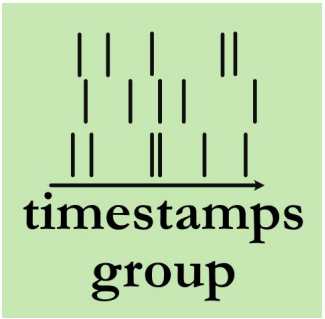


TsGroup manipulation

```
In [9]: ts_group[[0,2]]
Out[9]:
  Index    Freq. (Hz)  group          theta_modulation
-------  ------------  -----------  ------------------
      0           0.5  thalamus                      1
      2         10.01  cerebellum                    0
```

```
In [7]: ts_group
Out[7]:
  Index    Freq. (Hz)  group          theta_modulation
-------  ------------  -----------  ------------------
      0           0.5  thalamus                      1
      1           1    ca1                           1
      2         10.01  cerebellum                    0
```



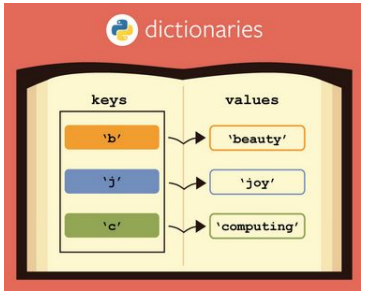


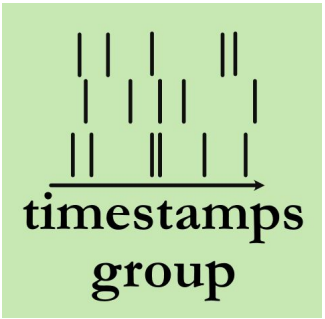
Select by :

- Threshold
- Category
- Intervals

```
In [11]: ts_group.getby_threshold('freq', 1)
Out[11]:
  Index    Freq. (Hz)  group          theta_modulation
-------  ------------  ----------  ------------------
      1             1  ca1                          1
      2         10.01  cerebellum                   0
```

```
In [7]: ts_group
Out[7]:
  Index    Freq. (Hz)  group          theta_modulation
-------  ------------  ----------  ------------------
      0           0.5  thalamus                     1
      1             1  ca1                          1
      2         10.01  cerebellum                   0
```



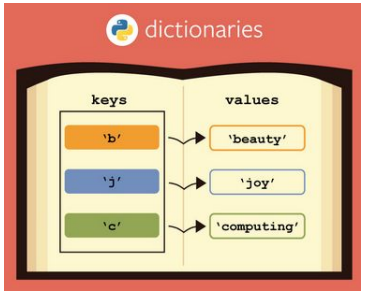


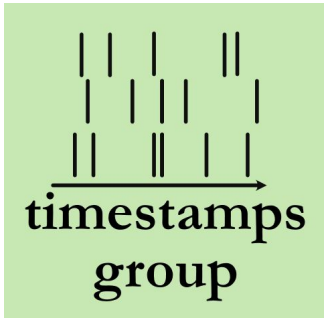
Select by :

- Threshold
- Category
- Intervals

```
In [12]: ts_group.getby_category('group')['thalamus']
Out[12]:
  Index    Freq. (Hz)  group       theta_modulation
-------  ------------  --------  ------------------
      0           0.5  thalamus                   1
```

```
In [7]: ts_group
Out[7]:
  Index    Freq. (Hz)  group          theta_modulation
-------  ------------  ----------  ------------------
      0           0.5  thalamus                    1
      1             1  ca1                         1
      2         10.01  cerebellum                  0
```



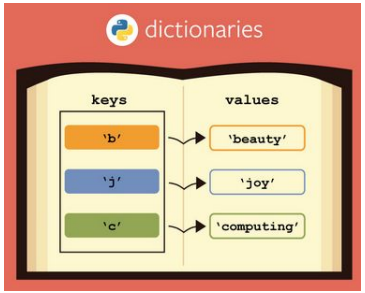


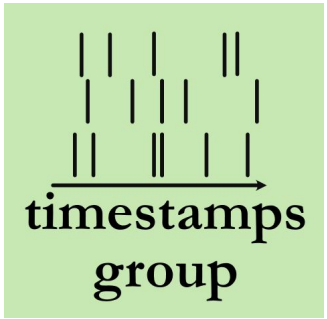
Select by :

- Threshold
- Category
- Intervals

```
In [14]: ts_group.getby_intervals('theta_modulation', [0.5, 1.5])[0][0]
Out[14]:
  Index    Freq. (Hz)  group       theta_modulation
-------  ------------  --------  ------------------
      0           0.5  thalamus                   1
      1           1    ca1                        1
```

```
In [7]: ts_group
Out[7]:
  Index    Freq. (Hz)  group          theta_modulation
-------  ------------  ----------  ------------------
      0           0.5  thalamus                    1
      1           1    ca1                         1
      2          10.01 cerebellum                  0
```



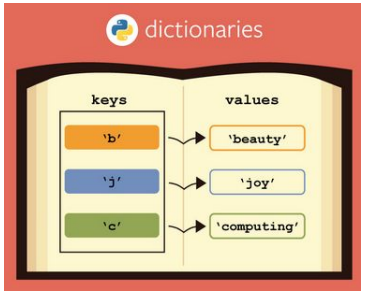


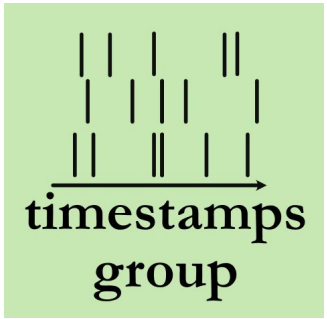
Operations :

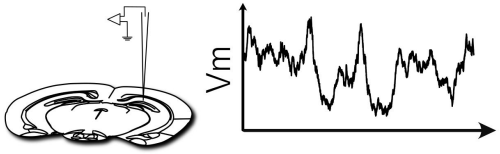
- restrict
- binning

```
In [10]: ts_group.count(bin_size=2, time_units='s')
Out[10]:
           0  1   2
Time (s)
1.0        1  2  20
3.0        1  2  20
5.0        1  2  20
7.0        1  2  20
9.0        1  2  20
11.0       1  2  20
13.0       1  2  20
```

```
In [7]: ts_group
Out[7]:
  Index    Freq. (Hz)  group         theta_modulation
-------  ------------  ----------  ------------------
      0           0.5  thalamus                     1
      1             1  ca1                          1
      2         10.01  cerebellum                   0
```

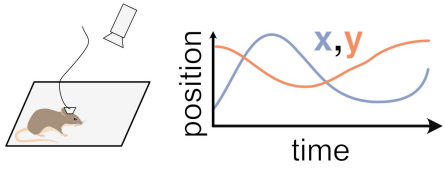






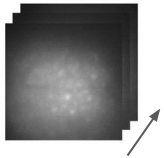
Tsd: 1-dimension

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



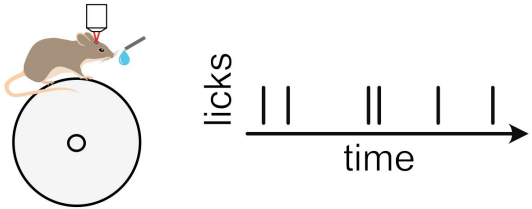
TsdFrame: 2-dimensions

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



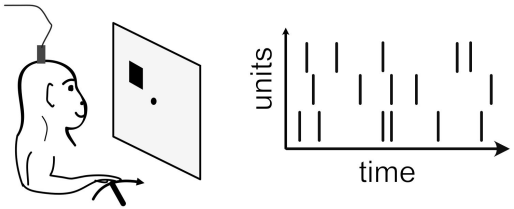
TsdTensor: n-dimensions

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



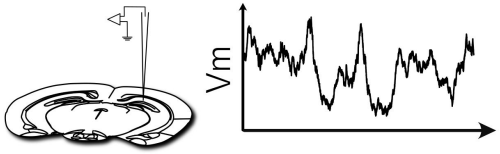
Ts: Timestamps

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



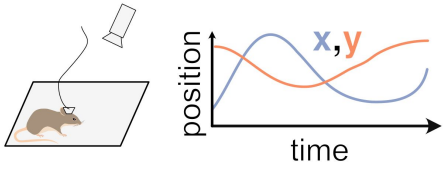
TsGroup: group of timestamps

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



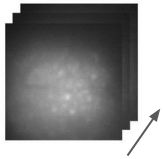
Tsd: 1-dimension

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



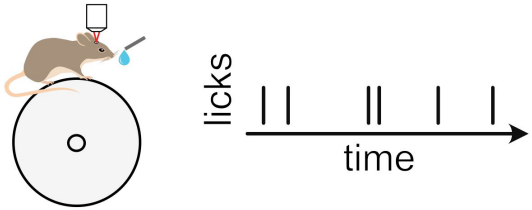
TsdFrame: 2-dimensions

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



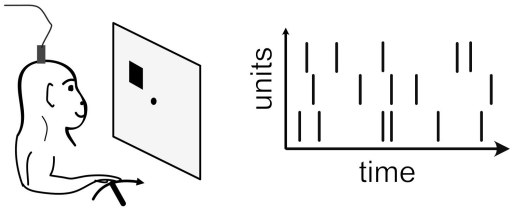
TsdTensor: n-dimensions

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



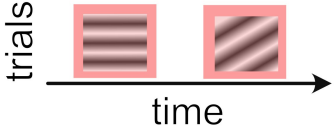
Ts: Timestamps

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

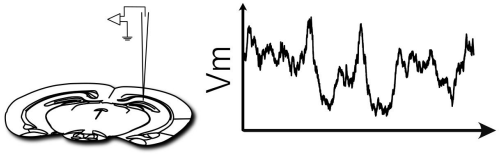


TsGroup: group of timestamps

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

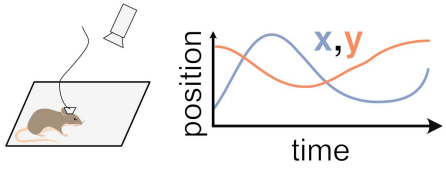


IntervalSet: set of epochs



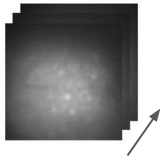
Tsd: 1-dimension

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



TsdFrame: 2-dimensions

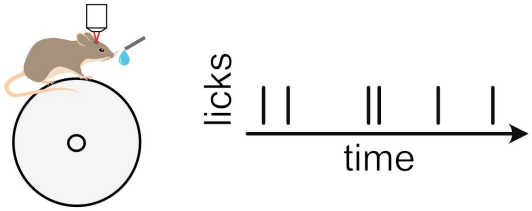
```
In [20]: tsdframe = nap.TsdFrame(t=t, d=d,
    ...:                 columns = ['x', 'y'])
```



FLATIRON INSTITUTE

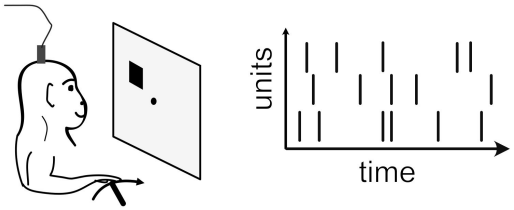
TsdTensor: n-dimensions

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



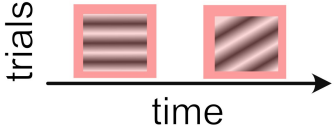
Ts: Timestamps

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



TsGroup: group of timestamps

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

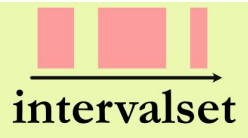


IntervalSet: set of epochs

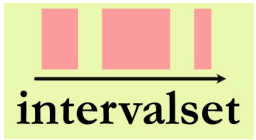
```
In [23]: nap.IntervalSet(
    ...:     start=start,
    ...:     end = end)
Out[23]:
   start   end
0    0.0   1.0
1    3.0   5.0
2    9.0  12.0
```

Manipulating time : the IntervalSet object

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

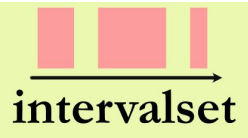


- Sleep/wake
- Stimulus on/off
- Lick start/end



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

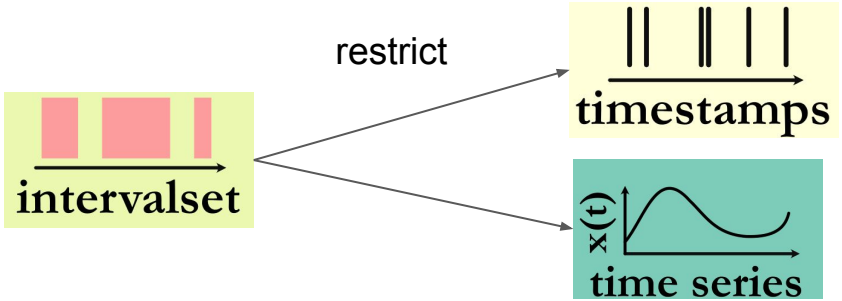
- Sleep/wake
- Stimulus on/off
- Lick start/end



	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
```

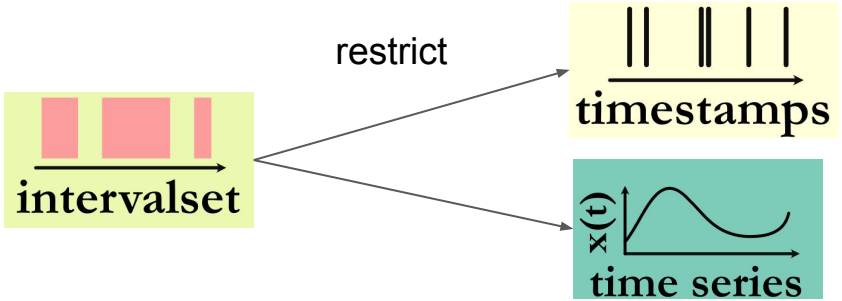
- Sleep/wake
- Stimulus on/off
- Lick start/end



	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

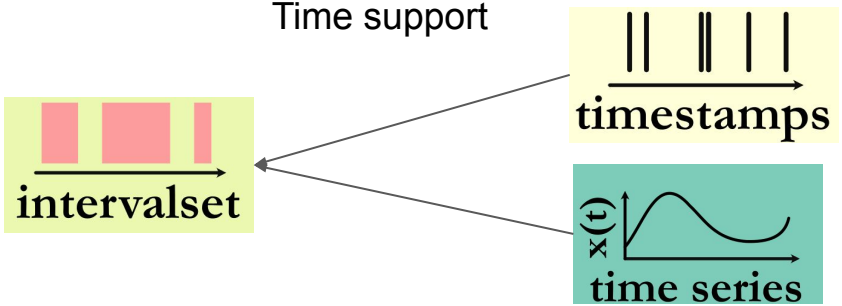


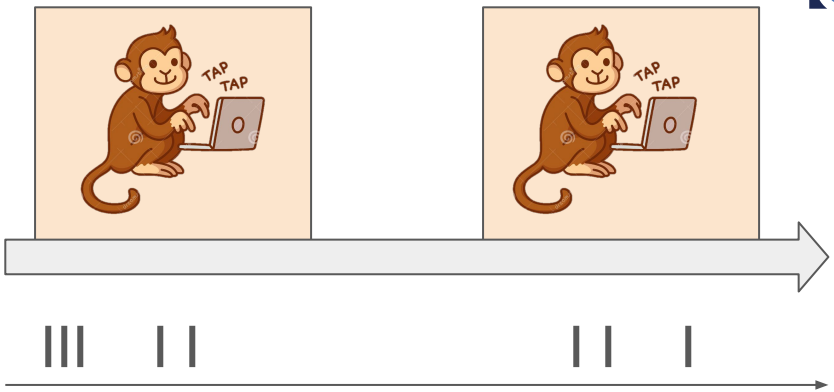

```
In [26]: nap.IntervalSet(start=[0, 3], end=[1, 5])
Out[26]:
   start  end
0    0.0  1.0
1    3.0  5.0
```

```
In [40]: ts
Out[40]:
Time (s)
0.297820   NaN
3.472073   NaN
5.145722   NaN
5.876591   NaN
5.897732   NaN
7.573689   NaN
7.577931   NaN
7.703431   NaN
8.405479   NaN
9.736642   NaN
dtype: float64
```

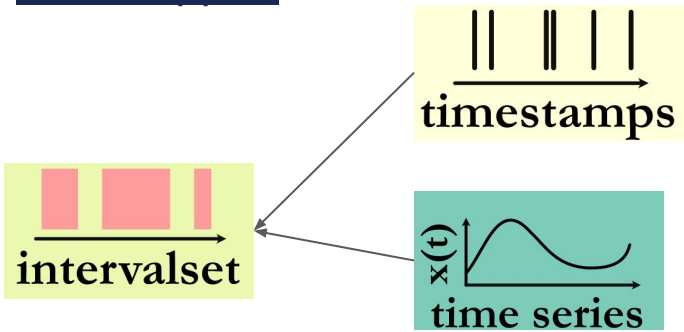
```
In [41]: ts.restrict(ep)
Out[41]:
Time (s)
0.297820   NaN
3.472073   NaN
dtype: float64
```

Time support

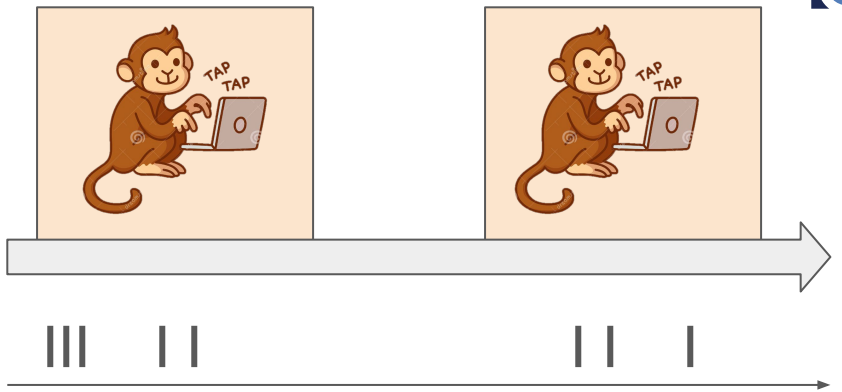




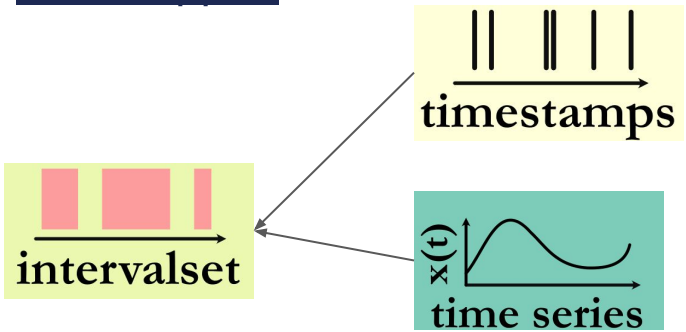
Time support



```
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
```

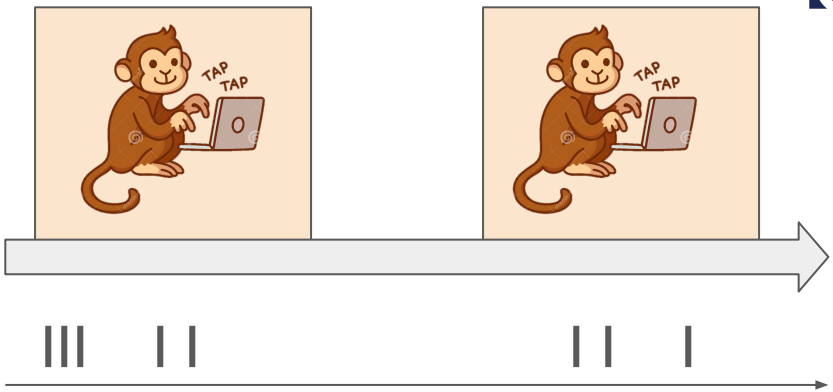


Time support

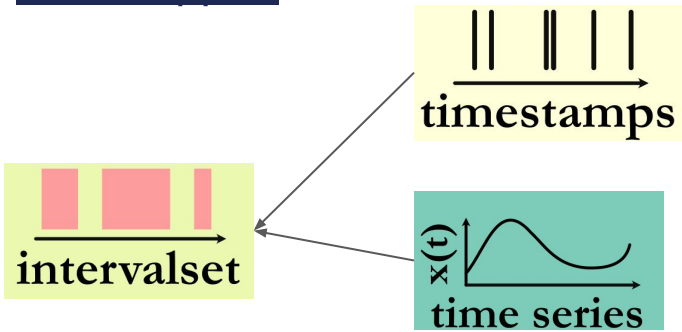


Tweet frequency?

```
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
```

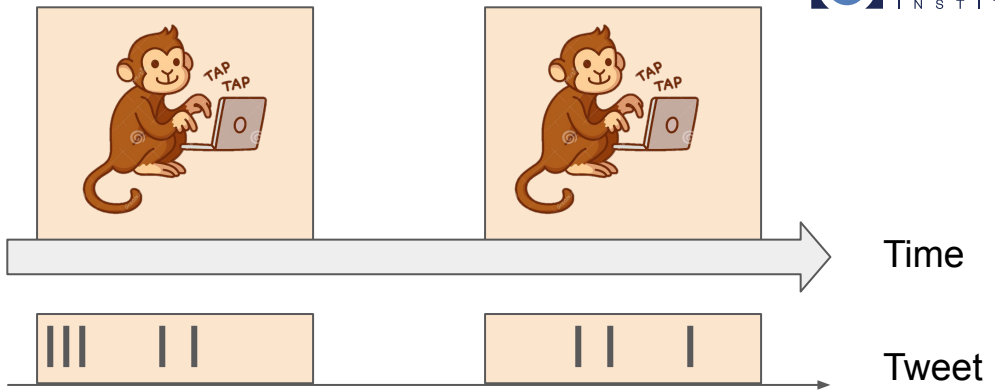


Time support

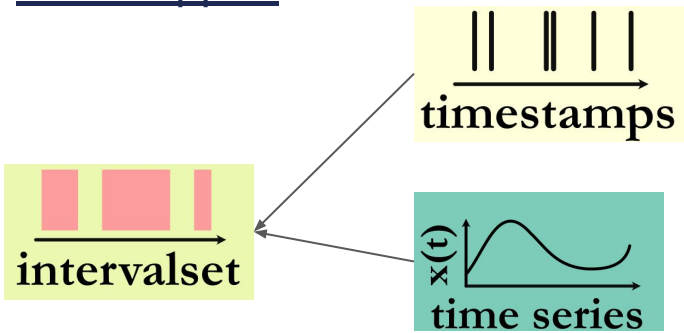


Tweet frequency?

```
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
```

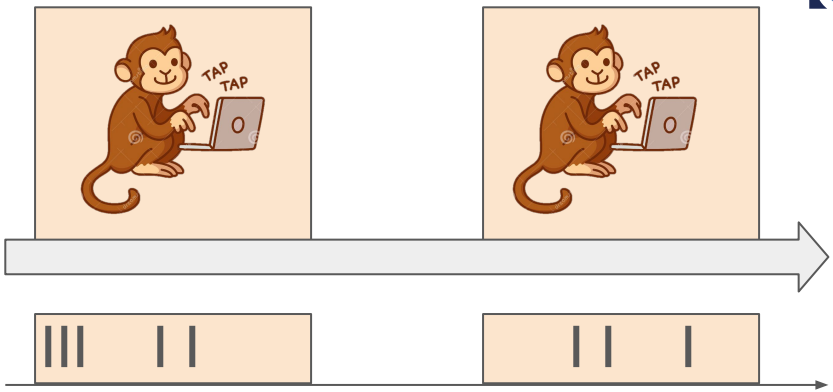


Time support



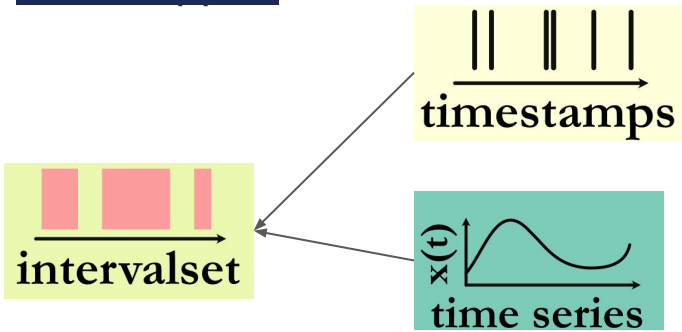
Tweet frequency?

```
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
```



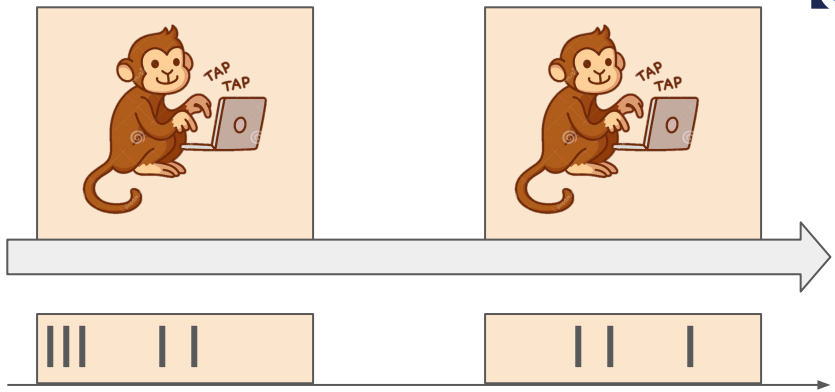

```
In [42]: tweeting_monkey.time_support
Out[42]:
   start    end
0    0.0   40.0
1   60.0  100.0
```

Time support



Tweet frequency?

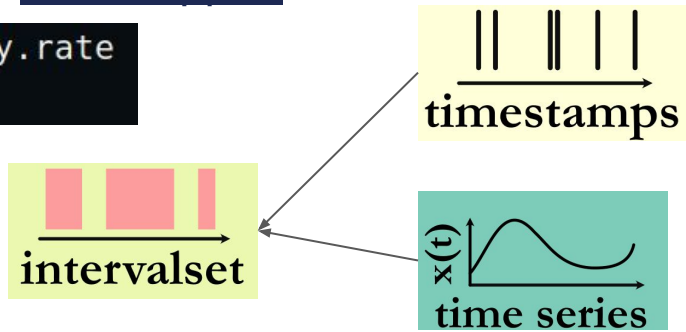
```
In [30]: tweeting_monkey
Out[30]:
Time (s)
0.126937916
0.320358519
0.707764437
1.942286662
3.163258247
...
96.895533151
97.002109352
98.01970871
98.842008394
99.609694697
shape: 100
```



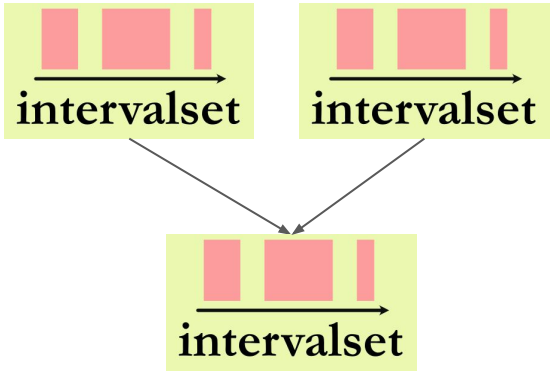

```
In [43]: tweeting_monkey.rate
Out[43]: 0.8125
```

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

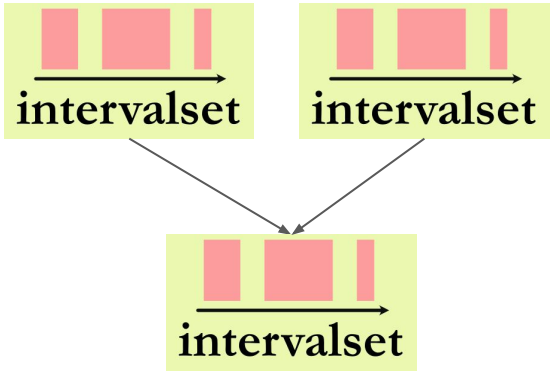
Time support



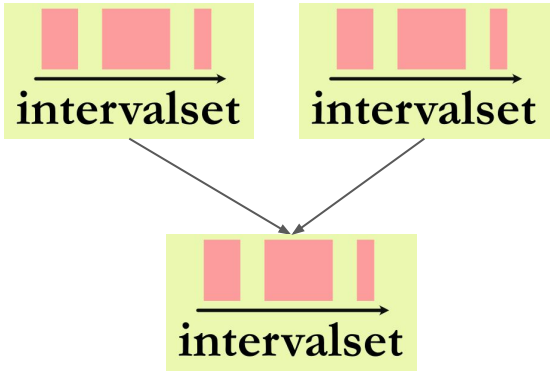
IntervalSet operations



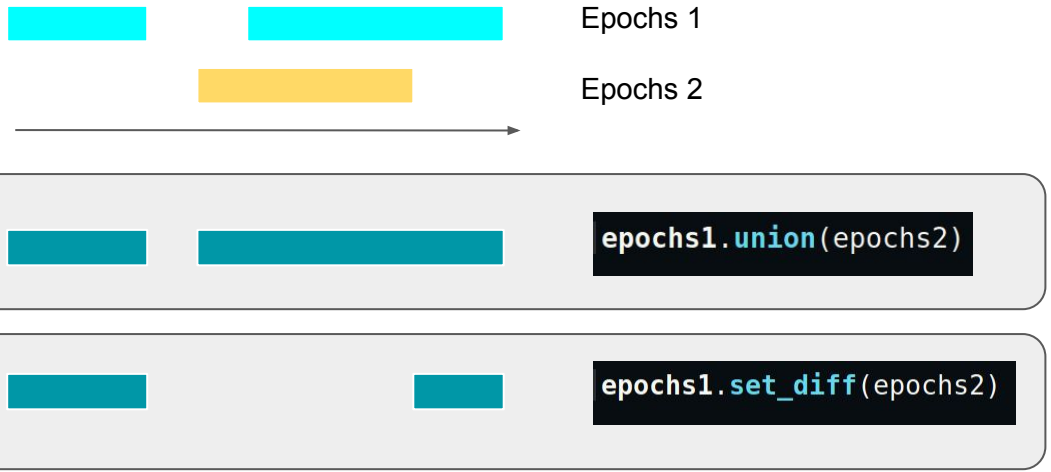
IntervalSet operations

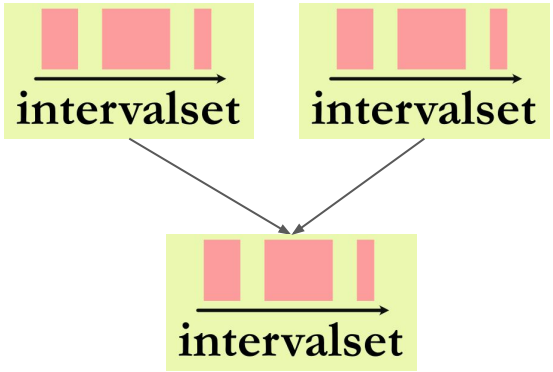


IntervalSet operations

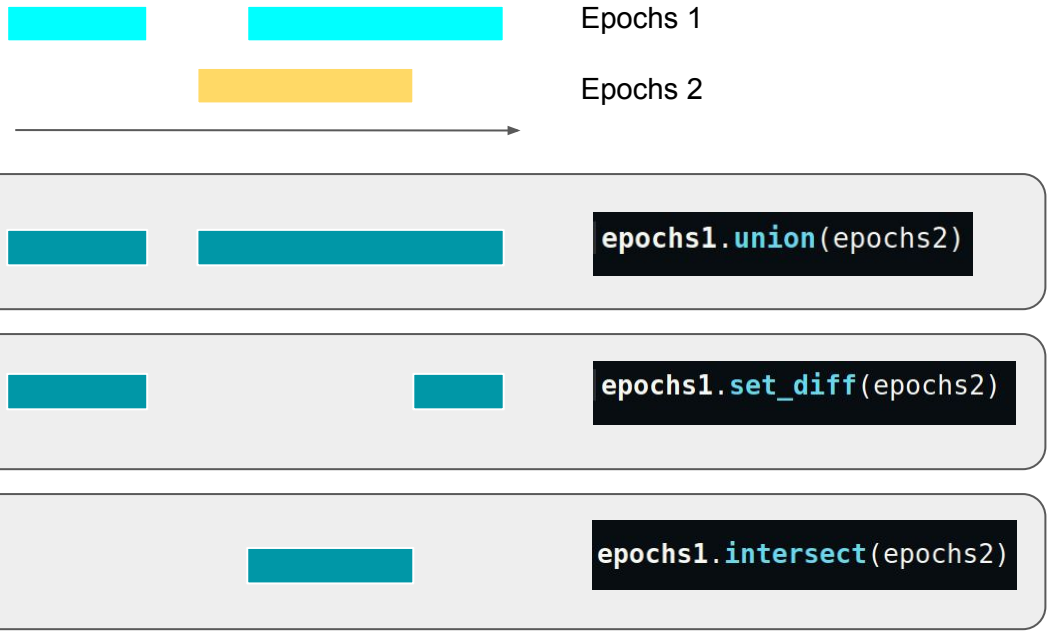


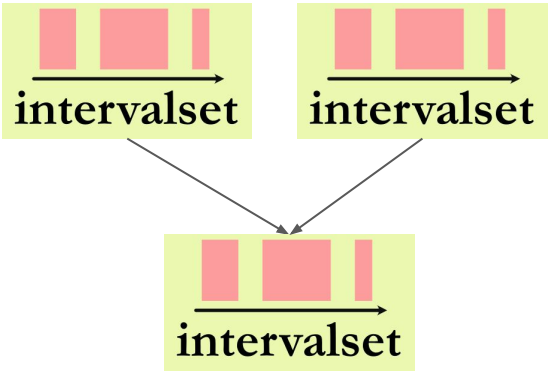
IntervalSet operations





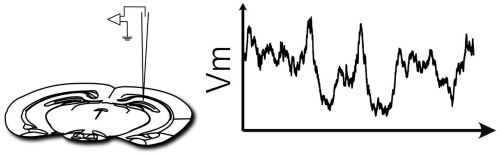
IntervalSet operations





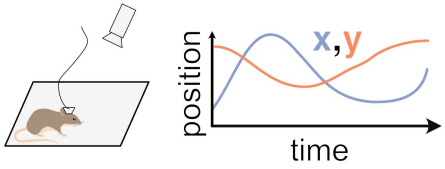
FLATIRON INSTITUTE

Summary : 6 objects to represent data



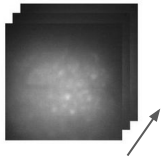
Tsd: 1-dimension

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



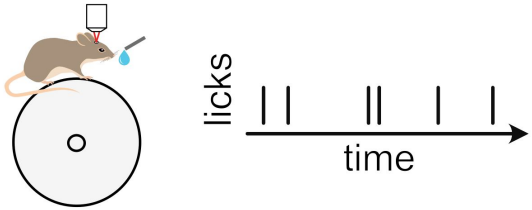
TsdFrame: 2-dimensions

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



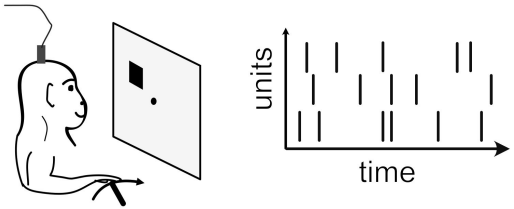
TsdTensor: n-dimensions

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



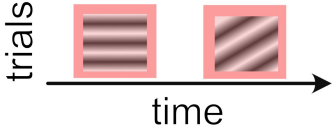
Ts: Timestamps

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



TsGroup: group of timestamps

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

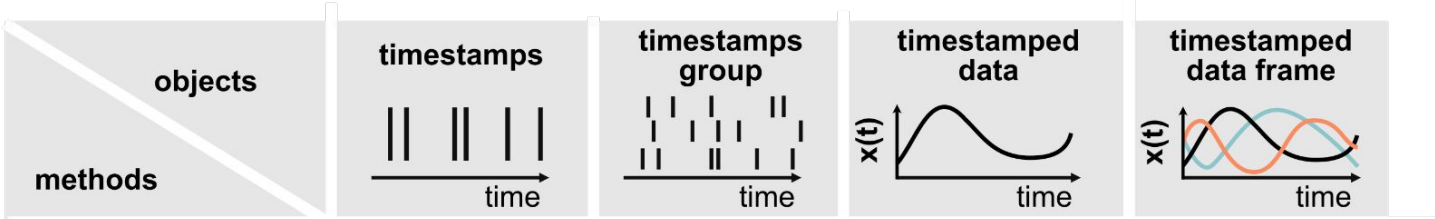


IntervalSet: set of epochs

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

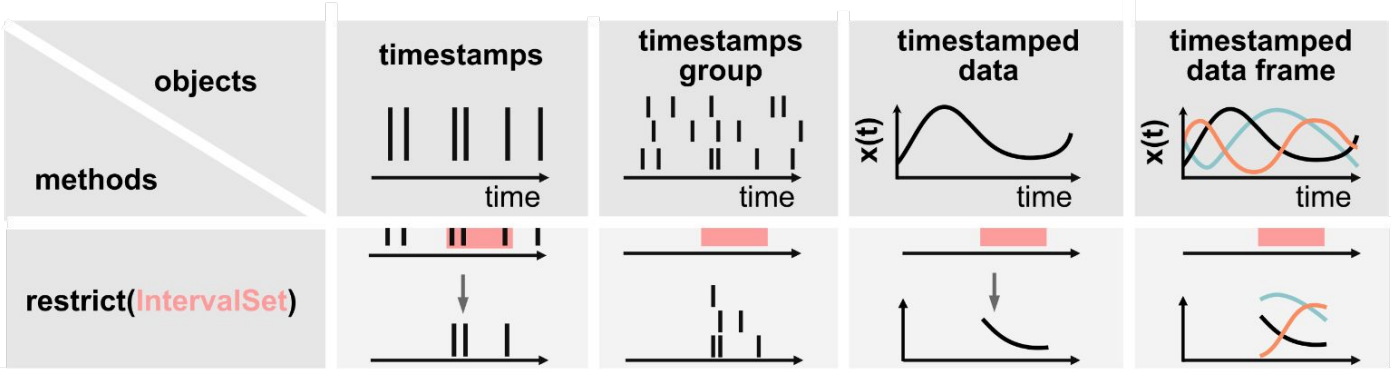
FLATIRON INSTITUTE

Core functions of pynapple



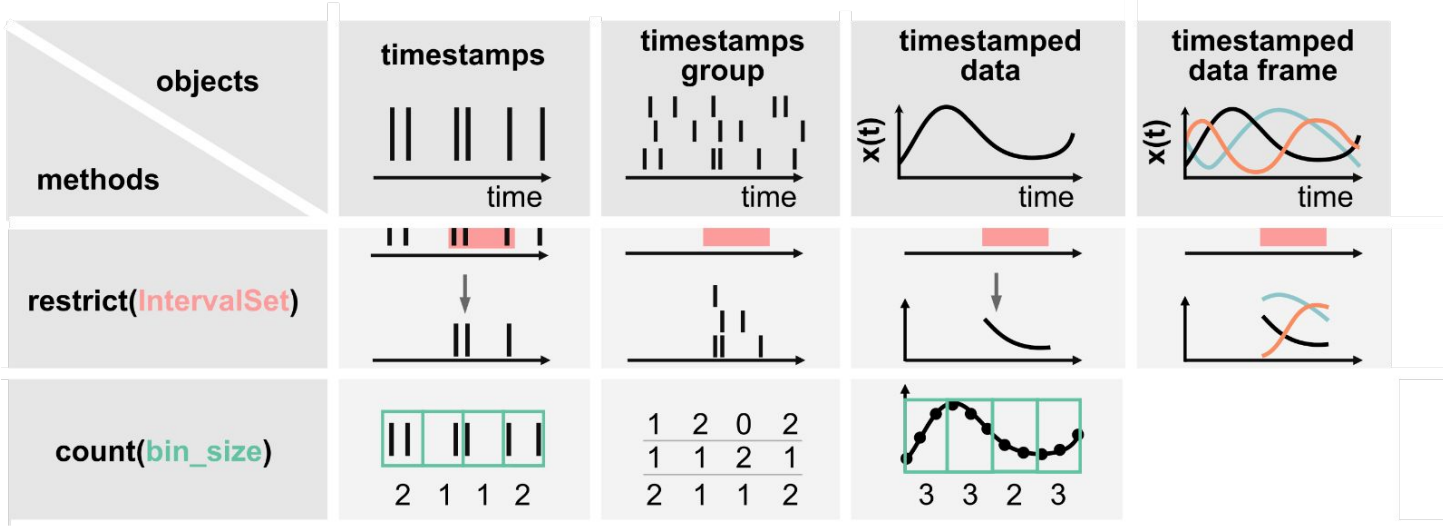
FLATIRON INSTITUTE

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.



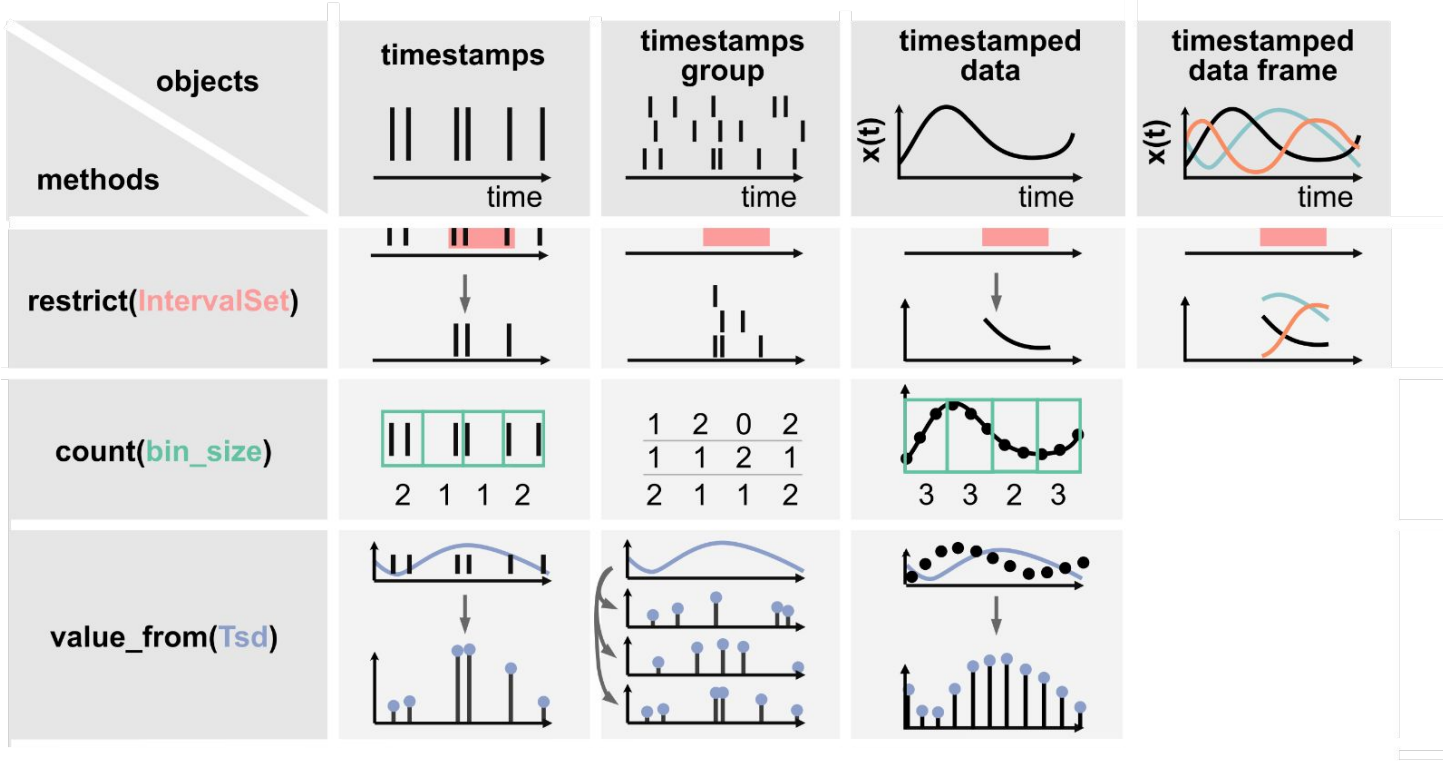
FLATIRON INSTITUTE

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.

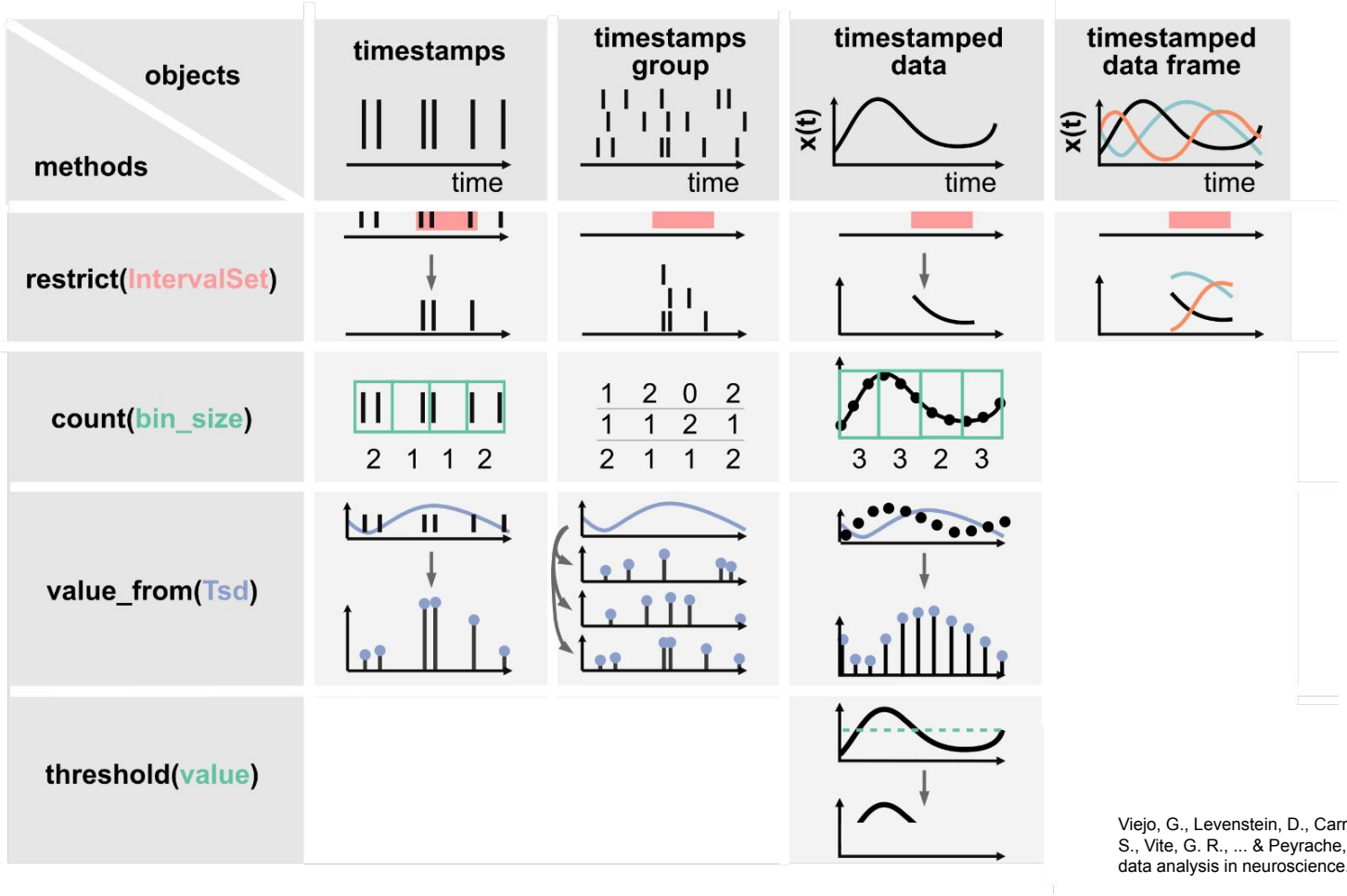


FLATIRON INSTITUTE

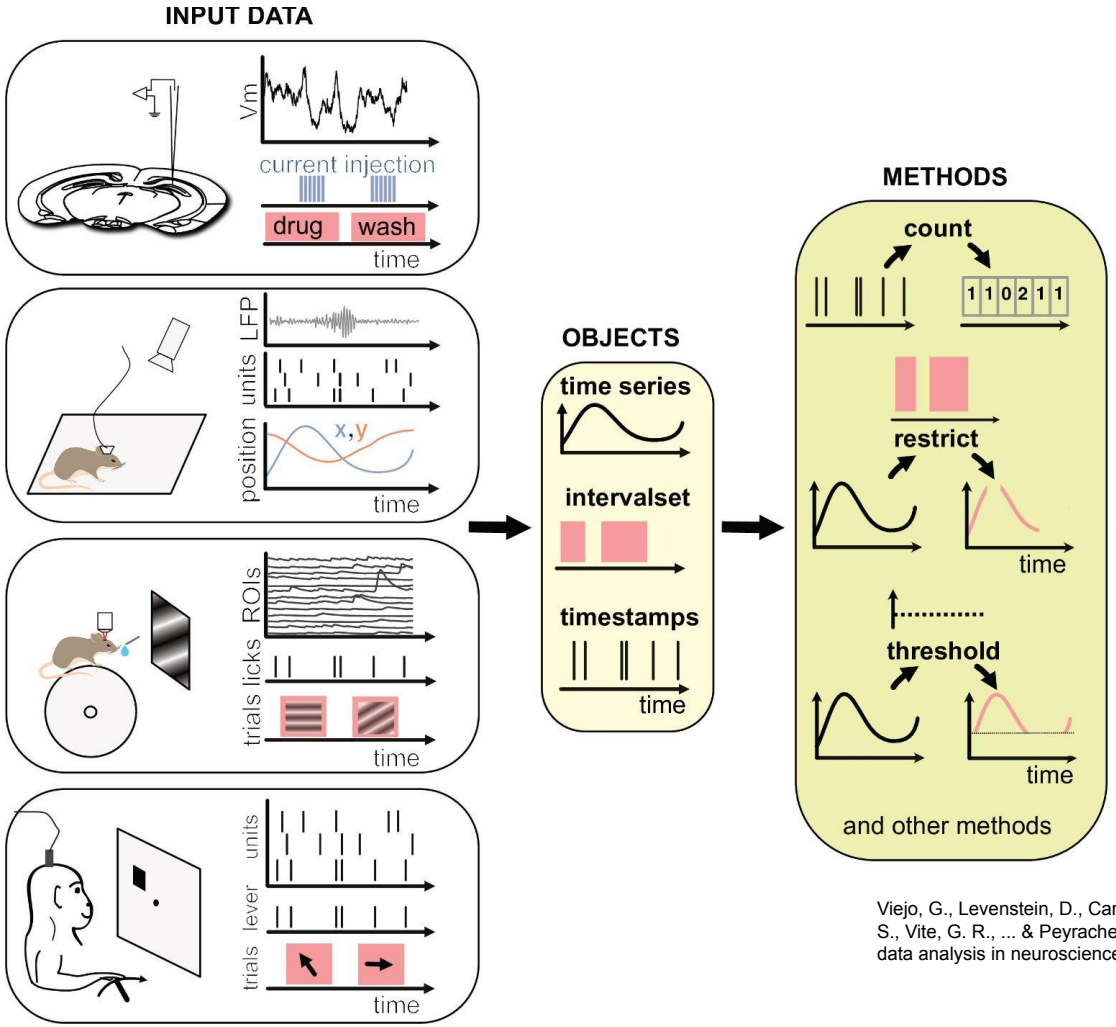
Viejo, G., Levenstein, D., Carrasco, S. S., Mehrotra, D., Mahallati, S., Vite, G. R., ... & Peyrache, A. (2023). Pynapple, a toolbox for data analysis in neuroscience. *eLife*, *12*, RP85786.



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.

Time to code


FLATIRON
INSTITUTE