```
import numpy as np
arr_a = np.random.rand(10000,2)
arr_b = np.random.rand(20000,50)
arr_c = np.random.rand(20,10000)
np.savez("many_array",arr_a,arr_b,arr_c)
#zipped as zip file saved in sample
```

```
arr_a,arr_b,arr_c
```

```
(array([[0.75202668, 0.5296053 ],
        [0.07620231, 0.13655821],
        [0.48438086, 0.22870396],
        ...,
        [0.75726877, 0.92348433],
        [0.96828337, 0.36706202],
        [0.51675935, 0.52393531]]),
 array([[0.96960362, 0.27712621, 0.10167781, ..., 0.80686473, 0.88201445,
         0.21711351],
        [0.00717427, 0.34799402, 0.11488967, ..., 0.36407085, 0.98060993,
         0.48433852],
        [0.32313477, 0.39173363, 0.04248859, ..., 0.40349942, 0.94001292,
         0.65890232],
        ...,
        [0.29214066, 0.58195224, 0.5571026 , ..., 0.76659108, 0.33066942,
         0.44045301],
        [0.24164372, 0.12784808, 0.50603242, ..., 0.81579179, 0.50001581,
         0.0468197 ],
        [0.27510611, 0.7951284 , 0.43025358, ..., 0.35635729, 0.34365181,
         0.41471756]]),
 array([[0.28722896, 0.90687596, 0.22470241, ..., 0.27362494, 0.26300118,
         0.2944816 ],
        [0.90120601, 0.75748615, 0.44847948, ..., 0.54349758, 0.1180226 ,
         0.65685671],
        [0.57245397, 0.73335697, 0.90276704, ..., 0.9095851 , 0.00742097,
         0.71201205],
        ...,
        [0.78111397, 0.17525652, 0.03580212, ..., 0.1448015 , 0.56251328,
         0.69441002],
        [0.35319205, 0.65290849, 0.39602913, ..., 0.93828409, 0.42378444,
         0.36563841],
        [0.64043817, 0.29770685, 0.26072668, ..., 0.9036335 , 0.97708148,
         0.95399434]]))
```

```
np.load("many_array.npz")
```

```
<numpy.lib.npyio.NpzFile at 0x7f948093fa90>
```

```
arrx=np.load("many_array.npz")
```

```
print(arrx)
```

```
<numpy.lib.npyio.NpzFile object at 0x7f94809e3f90>
```

```
print(type(arrx))
```

```
<class 'numpy.lib.npyio.NpzFile'>
```

```
arrx.files
```

```
['arr_0', 'arr_1', 'arr_2']
```

```
arrx.items
```

```
<bound method Mapping.items of <numpy.lib.npyio.NpzFile object at 0x7f94809e3f90>>
```

```
arrx["arr_0"]
```

```
array([[0.75202668, 0.5296053 ],
       [0.07620231, 0.13655821],
       [0.48438086, 0.22870396],
       ...,
       [0.75726877, 0.92348433],
       [0.96828337, 0.36706202],
       [0.51675935, 0.52393531]])
```

```
np.savez_compressed("comp many array",arr_a,arr_b,arr_c)
#similar to save

```

```
np.save("comp many array",arr_a,arr_b,arr_c)
```

```
  np.savetxt("comp many array",arr_a,arr_b,arr_c)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-23-fdd2a329aa6f> in <module>()
----> 1 np.savetxt("comp many array",arr_a,arr_b,arr_c)

<  array function   internals> in savetxt(*args, **kwargs)
```

```
#stacking of array
a1=np.arange(10).reshape(2,5)
print(a1)
a2=np.repeat(1,10).reshape(2,5)
print(a2)
```

```
[[0 1 2 3 4]
 [5 6 7 8 9]]
[[1 1 1 1 1]
 [1 1 1 1 1]]
        0.21711551],
```

```
a3=np.concatenate([a1,a2],axis=0)
```

```
       [0.32515477, 0.39173365, 0.04248859, ..., 0.40549942, 0.94001292,
```

```
a3
```

```
array([[0, 1, 2, 3, 4],
       [5, 6, 7, 8, 9],
       [1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1]])
        0.41471756]])
```

```
a3=np.concatenate([a1,a2],axis=1)
a3
```

```
array([[0, 1, 2, 3, 4, 1, 1, 1, 1, 1],
       [5, 6, 7, 8, 9, 1, 1, 1, 1, 1]])
```

```
a3=np.hstack([a1,a2])
a3
```

```
array([[0, 1, 2, 3, 4, 1, 1, 1, 1, 1],
       [5, 6, 7, 8, 9, 1, 1, 1, 1, 1]])
```

```
a3=np.vstack([a1,a2])
a3
```

```
array([[0, 1, 2, 3, 4],
       [5, 6, 7, 8, 9],
       [1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1]])
```

```
#stats with numpy
a1 = np.random.rand(100000)
a1
```

```
array([9.83369995e-01, 8.67161644e-01, 5.14316608e-04, ...,
       4.82632974e-01, 7.24642540e-01, 3.74749135e-01])
```

```
np.mean(a1)
```

```
0.4995549645228008
```

```
np.median(a1)
```

```
0.4995604504210791
```

```
np.var(a1)
```

```
0.08337568252605657
```

```
np.std(a1)
```

```
0.28874847623157524
```

```
np.min(a1)
```

```
4.042297034989595e-06
```

```
range = np.max(a1) - np.min(a1)
```

```
np.percentile(a1,25)
```

```
0.24897119335602044
```

```
IQR = np.percentile(a1,75) - np.percentile(a1,25)
IQR
```

```
0.5008934785985248
```

```
#zscore = xi-xbar /std dev
```

```
zscore = (a1-np.mean(a1))/np.std(a1)
zscore
```

```
array([ 1.67555873,  1.27310344, -1.72828842, ..., -0.0586046 ,
        0.77952818, -0.43223026])
```

```
np.mean(zscore)
#mean of this actually
```

```
7.414513447656645e-17
```

```
np.std(zscore)
#ideally 1
```

```
1.0
```

```
np.histogram(a1)
#1st row is binzise pf 10,000 of 1 each
```

```
(array([10004, 10143,  9854, 10036, 10015,  9986,  9975, 10016, 10078,
         9893]),
 array([4.04229703e-06, 1.00002519e-01, 2.00000996e-01, 2.99999473e-01,
        3.99997951e-01, 4.99996428e-01, 5.99994905e-01, 6.99993382e-01,
        7.99991859e-01, 8.99990336e-01, 9.99988813e-01]))
```

```
np.histogram(a1,bins=100)
#100 bins of 1000 each
```

```
(array([1003,  975,  986, 1048, 1016,  988,  981, 1004, 1008,  995, 1049,
         998, 1061, 1037,  952,  997, 1039,  985,  981, 1044,  996, 1008,
         965, 1013,  969, 1004,  970,  953,  957, 1019, 1003, 1094, 1006,
        1022,  950,  986,  994,  996,  929, 1056, 1024,  982,  994, 1035,
         941,  975, 1015, 1010, 1017, 1022, 1011,  980, 1043,  983, 1075,
         935,  972,  987,  966, 1034,  985,  977, 1032,  995,  995,  994,
        1037,  996,  969,  995,  929,  994,  975, 1042, 1062, 1021, 1006,
         990,  988, 1009, 1021,  994,  923, 1074, 1036,  981, 1008, 1014,
        1030,  997, 1026,  950, 1003,  980,  973,  981,  984, 1031,  972,
         993]),
 array([4.04229703e-06, 1.00038900e-02, 2.00037377e-02, 3.00035854e-02,
        4.00034331e-02, 5.00032808e-02, 6.00031285e-02, 7.00029762e-02,
        8.00028239e-02, 9.00026716e-02, 1.00002519e-01, 1.10002367e-01,
        1.20002215e-01, 1.30002062e-01, 1.40001910e-01, 1.50001758e-01,
        1.60001606e-01, 1.70001453e-01, 1.80001301e-01, 1.90001149e-01,
        2.00000996e-01, 2.10000844e-01, 2.20000692e-01, 2.30000540e-01,
        2.40000387e-01, 2.50000235e-01, 2.60000083e-01, 2.69999930e-01,
        2.79999778e-01, 2.89999626e-01, 2.99999473e-01, 3.09999321e-01,
        3.19999169e-01, 3.29999017e-01, 3.39998864e-01, 3.49998712e-01,
        3.59998560e-01, 3.69998407e-01, 3.79998255e-01, 3.89998103e-01,
        3.99997951e-01, 4.09997798e-01, 4.19997646e-01, 4.29997494e-01,
        4.39997341e-01, 4.49997189e-01, 4.59997037e-01, 4.69996884e-01,
        4.79996732e-01, 4.89996580e-01, 4.99996428e-01, 5.09996275e-01,
        5.19996123e-01, 5.29995971e-01, 5.39995818e-01, 5.49995666e-01,
        5.59995514e-01, 5.69995361e-01, 5.79995209e-01, 5.89995057e-01,
        5.99994905e-01, 6.09994752e-01, 6.19994600e-01, 6.29994448e-01,
        6.39994295e-01, 6.49994143e-01, 6.59993991e-01, 6.69993839e-01,
        6.79993686e-01, 6.89993534e-01, 6.99993382e-01, 7.09993229e-01,
        7.19993077e-01, 7.29992925e-01, 7.39992772e-01, 7.49992620e-01,
        7.59992468e-01, 7.69992316e-01, 7.79992163e-01, 7.89992011e-01,
        7.99991859e-01, 8.09991706e-01, 8.19991554e-01, 8.29991402e-01,
        8.39991250e-01, 8.49991097e-01, 8.59990945e-01, 8.69990793e-01,
        8.79990640e-01, 8.89990488e-01, 8.99990336e-01, 9.09990183e-01,
        9.19990031e-01, 9.29989879e-01, 9.39989727e-01, 9.49989574e-01,
        9.59989422e-01, 9.69989270e-01, 9.79989117e-01, 9.89988965e-01,
        9.99988813e-01]))
```

```
np.histogram(a1,bins=[0,0.1,0.15,0.6,0.7,0.9,1])
```

```
(array([10004,  5096, 44939,  9974, 20094,  9893]),
 array([0.  , 0.1 , 0.15, 0.6 , 0.7 , 0.9 , 1.  ]))
```

```
np.histogram(a1,bins=[0,0.2,0.4,0.6,0.8,1])
```

```
(array([20147, 19891, 20001, 19990, 19971]),
 array([0. , 0.2, 0.4, 0.6, 0.8, 1. ]))
```

```
#digitise
np.digitize(a1,bins=[0,0.1,0.15,0.6,0.7,0.9,1],right=True)
```

```
array([6, 5, 1, ..., 3, 5, 3])
```

```
a2=np.random.randint(1,20,10)
a2
```

```
array([13,  1, 18, 10, 19,  9, 10,  7,  3, 19])
```

```
bins=[2,4,6,8]
bins
```

```
[2, 4, 6, 8]
```

```
np.digitize(a2,bins)
```

```
array([4, 0, 4, 4, 4, 4, 4, 3, 1, 4])
```

```
#by defaault , left inclusive
#to make right inclusive
np.digitize(a2,bins,right=True)
```

```
array([4, 0, 4, 4, 4, 4, 4, 3, 1, 4])
```