In-class exercise for tutorial 010

In this exercise we will practice making datasets. We will on the one hand simulating data, on the other hand not quite as we will be making some asumptions and simplifaction on the data generation process that will not make the data.

Nestor lives right under the ATX airport, along the routes of airliners from major companies (United, Delt, etc).

Because of that Nestor hears noise. The noise is generated by the airplanes landing and taking off. There is a lot of noise that is generated by these airplanes. One day Nestor decides to simulate the noise generated by the airplanes and reaching his years. [Nestor is worried about his hearing loss (https://www.cdc.gov/niosh/topics/aircrew/noise.html)](https://www.cdc.gov/niosh/topics/aircrew/noise.html).

The problem to simulate

Nestor is interested in calculating how much noise he is being exposed to. So he goes and looks up the timetable during the 3 hours window in the morning, when he is at home, and during the 7 hours window in the evening, when he is at hom, and before midnight when the airport shuts down and no more flights land or takeoff.

Nestor lives about 2 miles away from the airport. At that distance each jet generates about 75-80 dB (Decibels). We will say 75dB. In the morning there are about 45 airplanes that land/takeoff. Each airplane can be heard consistently for about 7 minutes (we will assume a flat top distribution of the dBs, no ramp up, no decay, a simple flat distribution).

In the 3 hours window of the morning, Airplanes depart and land every 3 minutes, so their noise overlap for about 4 minutes. The dB of an airplane is corrupted by noise due to the city and nature around around, the cars, trains, trucks (all add some noise, randomly) and the position of the moving cloud in the sky, the wind and humidity (all diminish the noise randomly). So the noise is never 75dB but it stays on average around 75 dB while being corrupted by noise.

In the 7 hours window of the evening, Airplanes depart and land about every 4 minutes, so their noise overlaps for about 3 minutes.

Nestor will assume a linear summation of the airplane noise in a single day. This is not the best way especially when dealing with dB, but this is a simple exercise and we can break some fundamental physics rules to make things easier for us. We will want to simulate the situation.

How many airplanes depart/land in the morning widow of a single day. How many in the evening window?

In [1]:
```python
noyalevel =75
noyavar =2
planeintervalam = 3
planeintervalpm = 4
noysminperplane = 7


mornmins = 3*60
evemins = 7*60

mornplanes = mornmins/planeintervalam
eveplanes = evemins/planeintervalpm
print(mornplanes, '.......')
```

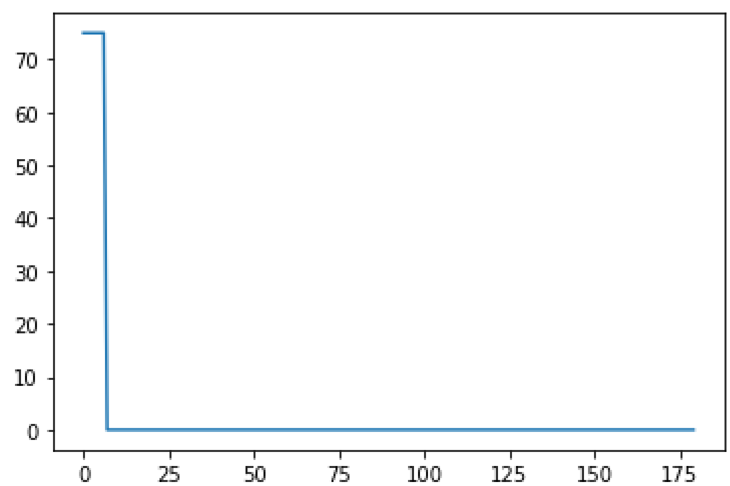
```
60.0 .......
```

Show the noise profile of one airplane in the morning (pure for the moment no corruption but other exsternal noise)

In [2]:
```python
import numpy as np
import matplotlib.pyplot as plt
```

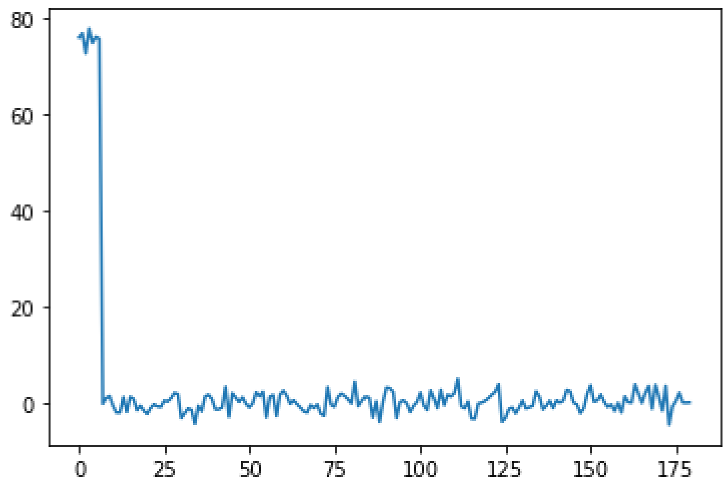
In [3]:
```python
planenoys = np.zeros([mornmins,1])
planenoys[0: noysminperplane] = noyalevel
plt.plot(planenoys)
```

Out[3]:
```
[<matplotlib.lines.Line2D at 0x7fd0ea2e1b20>]
```



Show the noise profile of the same airplane corrupted by some small noise.

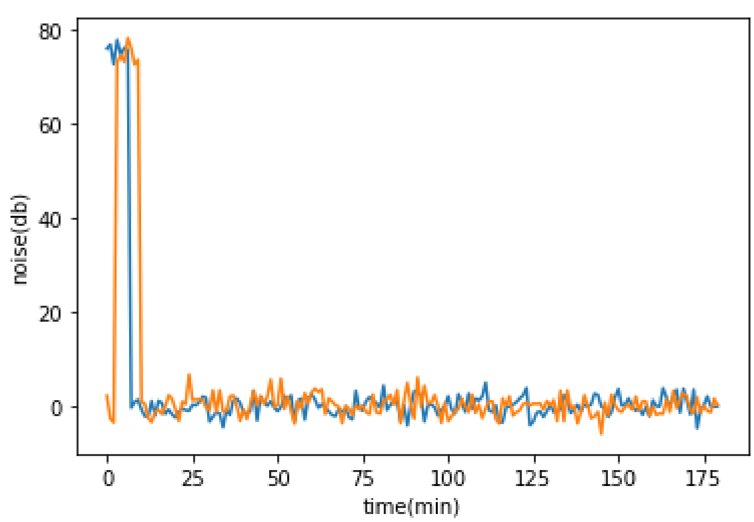
In [4]:
```
envnoys = noyavar*np.random.randn(mornmins,1)
planenoys = planenoys + envnoys
plt.plot(planenoys);
```



Now add a second airplane, corrupted by noise but departing/landing at a different time. Make a plot of the two airplanes together.

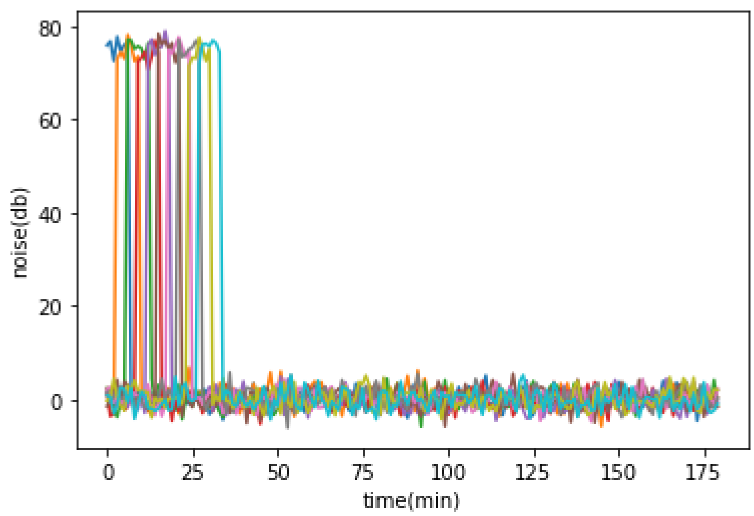
In [5]:
```
start = planeintervalam
stop = start +noysminperplane

plane2noys = np.zeros([mornmins,1])
plane2noys[start:stop,:] = noyalevel

envnoys = noyavar*np.random.randn(mornmins,1)
plane2noys = plane2noys + envnoys

planenoys = np.hstack((planenoys, plane2noys))
plt.plot(planenoys)
plt.xlabel('time(min)')
plt.ylabel('noise(db)')
```

Out[5]:
```
Text(0, 0.5, 'noise(db)')
```

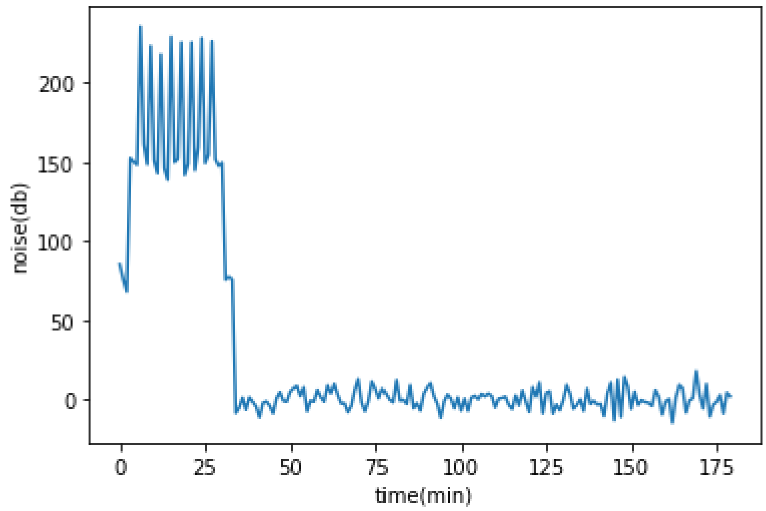


Finally, simulate all the the airplanes that you have estimated to land/depart in the morning. Plot them on the same figure.

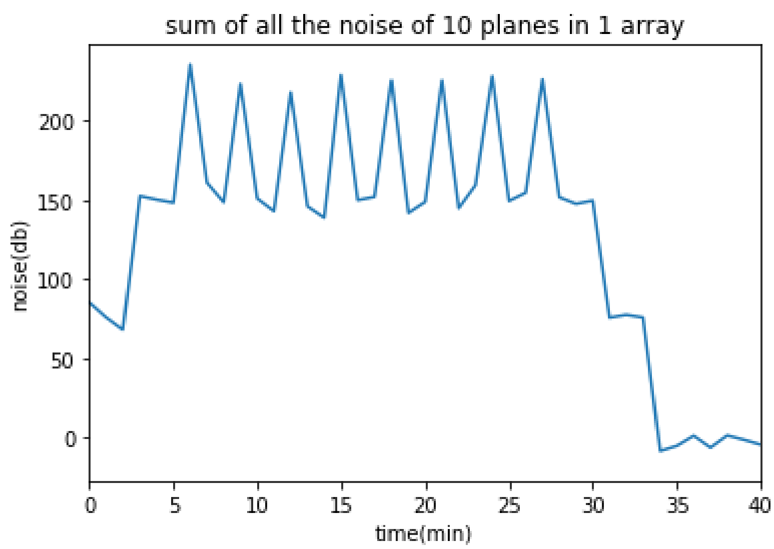
In [6]:
```
for i in range (2, 10):
    planeindex = i
    start = planeintervalam*planeindex
    stop = start +noysminperplane

    nextplanenoys = np.zeros([mornmins,1])
    nextplanenoys[start:stop,:] = noyalevel
    envnoys = noyavar*np.random.randn(mornmins,1)
    nextplanenoys = nextplanenoys + envnoys

    planenoys = np.hstack((planenoys, nextplanenoys))

plt.plot(planenoys);
plt.xlabel('time(min)');
plt.ylabel('noise(db)');
```



In [9]:
```
# combining all the arrays so the gragh will have one variable of noise thr

plt.plot(np.sum(planenoys,1))
plt.xlabel('time(min)');
plt.ylabel('noise(db)');
```



In [12]:
```
plt.plot(np.sum(planenoys,1))
plt.title('sum of all the noise of 10 planes in 1 array');
plt.xlabel('time(min)');
plt.ylabel('noise(db)');
plt.xlim([0,40]);
```



Next, let's start with the evening airplanes. Plot 20% of the total airplanes you estimated to depart/land in the evening. Make a plot.

In [ ]:
```
```

Plot all the evening airplanes.

In [ ]:

Combine the full day. Add the morning airplanes to the evening airplanes (hint: make a single numpy array). Plot all airplanes.

In [ ]:

Now the simple theory Nestor is using is that the noise will add across all airplanes. So what is the average noise across the morning and evening hours that Nestor is exposed to?

In [ ]:

Now use imshow on the full numpy array of all airplane noise. How does it look like?

In [ ]:

Simulate the dataset of tutorial009. For this final exercise you can use tutorial009 and repurpose some of the code (for example for plotting). Also, you can use and should the dataset given to youin tutorial 090. Indeed, you will need to figure out how to generate that very same dataset (up to noise differences). Importantly, the dataset you will generate will need to keep a very similar similar Signal-to-Noise ratio of the given dataset. To do this all, try exploring the properties of the given dataset. Then implement code that generates a new dataset with similar dimensions and properties of signal and noise.

In [ ]:

So here's our mission: See if you can write some similar code in which you can generate 10 planes of noise data in a single 180x10 numpy array, **where only one number needs to change across planes**. Then compute the total noise over time and make sure you get something like the above.

If you have that handled and have spare time or brain cycles, see if you if you can do this **using only one numpy array** – a 180 min by 10 plane array – and no additional arrays to hold individual plane noise or environmental noise.

Brain challange: we have used different "environmental" noise for each plane over the *same* 180 minutes? Is this reasonable? Or might there be environmental noise that should be common to all of the planes' noise profiles?

In [ ]: