Python Exercises II

Write one script for each of the following exercises.

Exercise 1: Taking the average

Write a function called `compute_average` that accepts the following arguments:

1. `data`: A `numpy` array of floats.
2. `weights`: An optional argument with a default value of `None`. If specified, the argument should be a `numpy` array of floats (the same length as `data`) that will be used to "weight" the average.

If `weights` is `None`, the function should return the unweighted average of `data`: the sum of all data points divided by the total number of data points. If `weights` is specified, the function should return the weighted average of `data`. To calculate the weighted average, first multiply each data point by its weight, then sum all the results. Finally, divide by the sum of the weights. (Don't use `np.mean()`. Other `numpy` functions are allowed!)

Show that your function works by calculating the weighted and unweighted averages of the following dataset:

```python
data = np.array([10.2, 11.3, 10.8, 12.1, 9.7, 10.0, 10.5, 11.7, 9.5, 11.0])
weights = np.array([0.5, 1.2, 0.8, 0.6, 1.0, 1.1, 1.3, 0.7, 1.5, 0.9])
```

Exercise 2: Sigma-clipping

A common way to remove outliers from data is by *sigma-clipping*. In this method, the user provides a `cutoff` for how many standard deviations (σ) a data point is allowed to deviate from the mean. The sigma-clipping procedure is:

1. Calculate the mean of the data.
2. Remove all data points that are greater than σ * `cutoff` away from the mean.
3. Repeat from step 1 until no data points are removed.

Define a function called `sigma_clipping` that takes in the following arguments:

1. `data`: A `numpy` array of floats.
2. `cutoff`: An optional integer with a default value of 3.
3. `max_iters`: An optional integer with a default value of 10.

Your function should implement the sigma-clipping procedure described above and return a new `numpy` array with the outliers removed. Use `cutoff` * (the standard deviation of `data`) as your threshold for deciding whether or not to discard a data point. Your function should clean the data until no data points are removed OR it reaches `max_iters` iterations.

Show that your function works by sigma-clipping the following dataset:

```
Python
data = np.array([10.4967141530, 9.8617356988, 10.6476885381, 11.5230298564,
24.8870861000, 9.7658466253, 9.7658630431, 11.5792128155, 10.7674347292,
26.4056428790, 9.5305256141, 10.5425600436, 10.5365823072, 10.5342702464,
24.2087945460, 9.2419622716, 9.0867197553, 11.2750821675, 9.4377124708,
19.0887926320, 9.9871688836, 11.3142473326, 10.0919759245, 9.5876962987,
17.6904167760, 11.4656487689, 9.7742236995, 10.0675282047, 8.5752518138,
21.6903950680, 9.4556172755, 10.1109225897, 9.8490064226, 9.3756980183,
20.6022752500, 10.3993613101, 10.7083062502, 10.3982933878, 11.8522781845,
-1.6704414650, 9.9865027753, 9.9422890710, 10.8225449121, 10.7791563500,
30.0910035088, 10.2088635950, 9.0403298761, 10.6718139511, 10.1968612359,
15.1872653220])
```

Fun fact: there are functions in both the `scipy` and `astropy` packages that do sigma-clipping for you! Don't use them for this exercise, though.

Exercise 3: Zooming in on bright pixels

Write a function called `zoom` that takes in a 2D `numpy` array and finds the location of the largest value in the array. (Hint: Try using `np.where()` to find where the array is equal to its maximum, which you can get from `np.max()`.)

Once you've found the largest value, slice the array to select the 3x3 region surrounding that value and return the resulting slice.

Show that your function works by running it on the following array:

```
Python
image = np.array([[37.45401188, 95.07143064, 73.19939418, 59.86584842,
15.60186404, 15.59945203, 5.80836122, 86.61761458, 60.11150117, 70.80725778], [
2.05844943, 96.99098522, 83.24426408, 21.23391107, 18.18249672, 18.34045099,
30.42422430, 52.47564316, 43.19450186, 29.12291402], [61.18528947, 13.94938607,
29.21446485, 36.63618433, 45.60699842, 78.51759614, 19.96737822, 51.42344384,
59.24145689, 4.64504127], [60.75448519, 17.05241237, 6.50515930, 94.88855373,
```

```
96.56320331, 80.83973481, 30.46137692, 9.76721140, 68.42330265, 44.01524937],
[12.20382348, 49.51769101, 3.43885211, 90.93204021, 25.87799816, 66.25222844,
999.00000000, 31.17110761, 52.00680212, 54.67102793], [18.48544555,
96.95846278, 77.51328234, 93.94989416, 89.48273504, 59.78999788, 92.18742350,
8.84925021, 19.59828624, 4.52272889], [32.53303308, 38.86772897, 27.13490318,
82.87375092, 35.67533267, 28.09345097, 54.26960832, 14.09242250, 80.21969808,
7.45506437], [98.68869366, 77.22447693, 19.87156815, 0.55221171, 81.54614285,
70.68573438, 72.90071680, 77.12703467, 7.40446517, 35.84657285], [11.58690595,
86.31034259, 62.32981268, 33.08980249, 6.35583503, 31.09823217, 32.51833220,
72.96061783, 63.75574714, 88.72127426], [47.22149252, 11.95942459, 71.32447872,
76.07850486, 56.12771976, 77.09671799, 49.37955964, 52.27328294, 42.75410184,
2.54191267]])
```