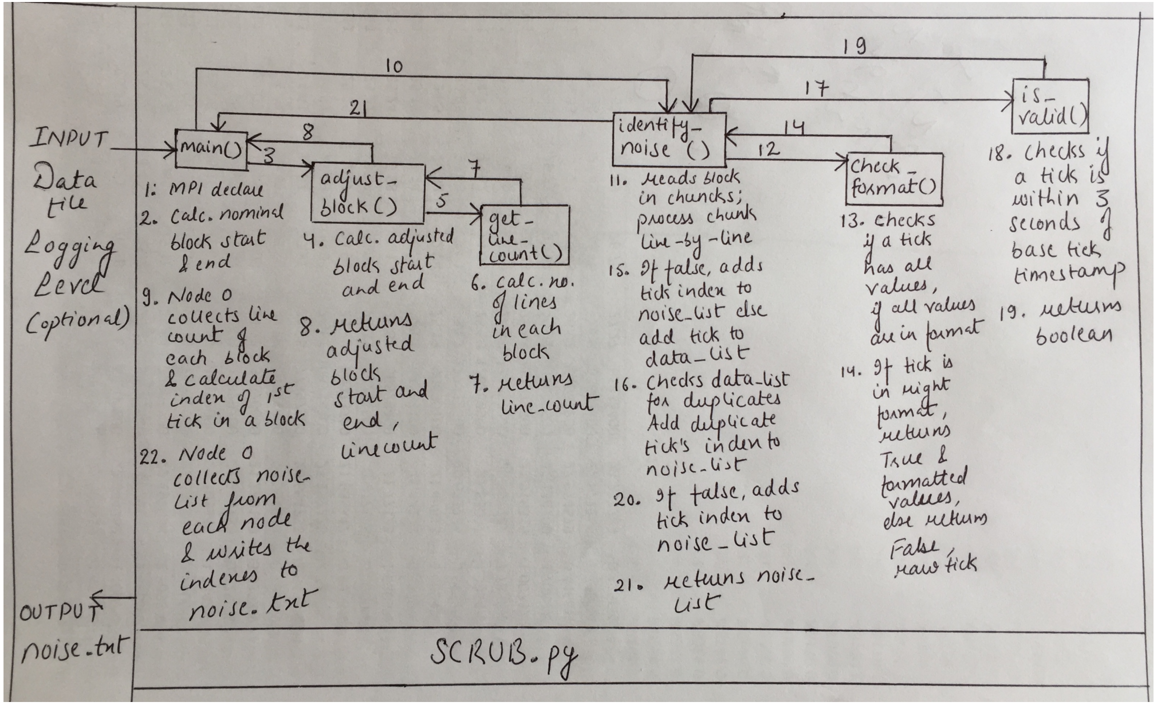
The block diagram below shows the workflow of my SCRUB.py program. It was found that all the timestamps were in right format and out of order; prices and units traded had negative and zero values. Some of the price and units values were outliers with very high positive values. According to this data, we do not need to check for timestamps format. Since future data can have formatting issues, this check is included.

In my SCRUB program, ticks are identified as noise if:

1. Number of items in a tick are less than 3 or items are not in right format
2. Price or units are zero or negative
3. Difference between current timestamp and base timestamp is more than three seconds.

My code does not identify ticks with very high price/ units as outliers since it is not the right approach. There may be genuine reason for them to be that high, so it should be investigated before categorizing them as noise.

The code is scalable for large datasets. Due to issues with Penzias I could run it only on my laptop. For data-small.txt it takes around 3 seconds when run with logging in info mode (includes times and memory heap) or debug mode (also includes reason for a tick to be noise). For data-big.txt it takes much more time but runs to completion. Using regex execution time can be decreased which would be an improvement in my program. Also, my program allows user to define the logging level through command line because of which no change in code is required to run it in different environments.



1: MPI declare
2. Calc. nominal block start & end
9. Node 0 collects line count of each block & calculate index of 1st tick in a block
22. Node 0 collects noise list from each node & writes the indexes to noise.txt

4. Calc. adjusted blocks start and end
8. returns adjusted blocks start and end, linecount

6. calc. no. of lines in each block
7. returns line-count

11. reads block in chuncks; process chunk line-by-line
15. If false, adds ticks index to noise_list else add tick to data_list
16. Checks data_list for duplicates Add duplicate tick's index to noise_list
20. If false, adds tick index to noise_list
21. returns noise_list

13. checks if a tick has all values, if all values are in format
14. If tick is in right format, returns True & formatted values, else returns False, raw tick

18. checks if a tick is within 3 seconds of base tick timestamp
19. returns boolean