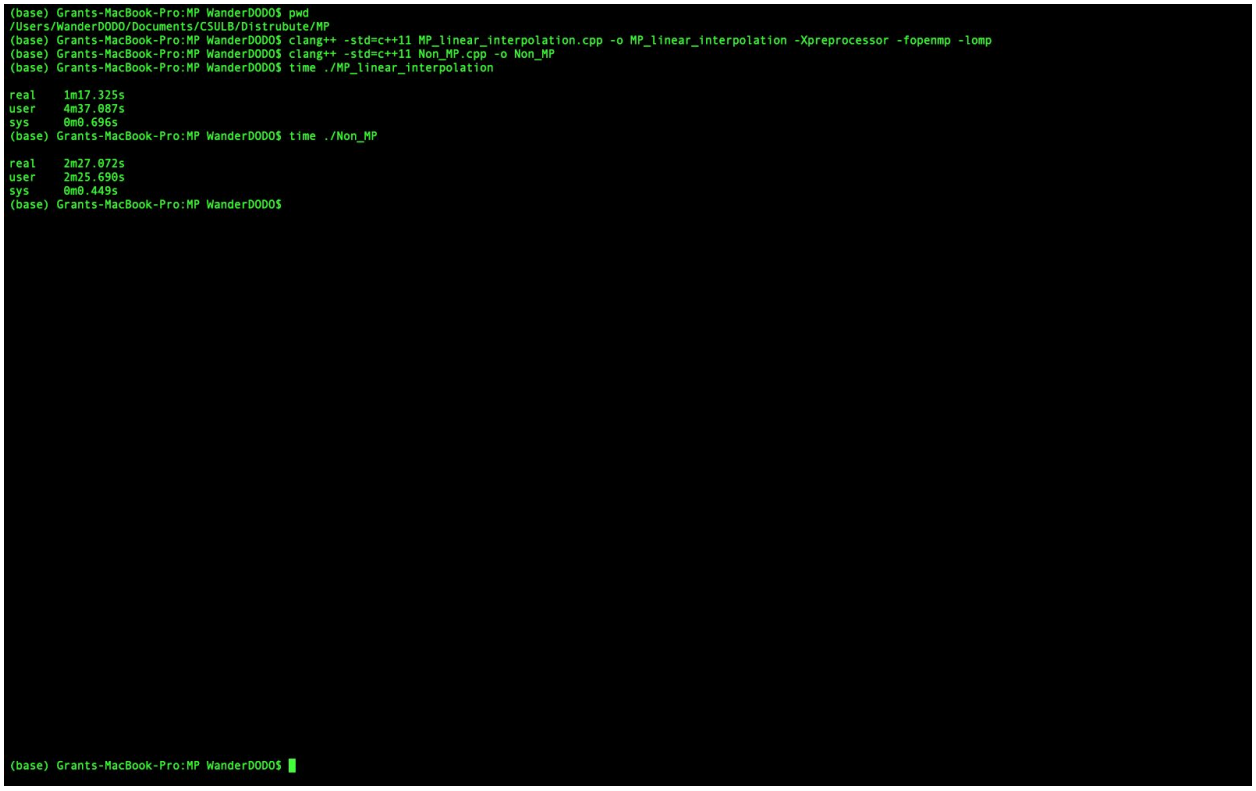
For this project I rewrite the code of linear interpolation.

First I used #pragma omp parallel with shared memory vector<double> yVal is where to store the output information, and NPTS which is the size of xVal and yVal also the number to do iterations.



We can find out by using multi-thread, the real processing time decrease because the work splict to the CPUs. And the total user time increases due to the total processing time of CPUs increased, to explain clearierly in about 1m17 the CPUs did the work that required 4m37 to process which means multi-threads can use CPU more efficiently.

Then I try the different omp_set_num_threads from 16, 8 to 4.

By decreasing the threads, the real processing time increased because each thread need to serve more work.

One step ahead, I tried to let my program do more multithreading process. Now I use multi-thread to initialize my vector variables.

But unfortunately, the performance does not speed up. For the reason I concluded is because the vector/array initialization is more about memory accessing/storing than the computation, so even using multi-thread to accelerating the computation part doesn't affect the real processing time much.