Results:

After completing the brute force and quick hull algorithms, my results for timing, while not being completely accurate, did show the difference in time complexity that is described in the book. Brute force exponentially increased as the file size increased, while the quick hull algorithm tended to lean a lot more towards slightly worse than n\*log(n) time. Specifically, my times are as follows in the table and graph below:

|  |  |  |
| --- | --- | --- |
| Points | Brute Force Time | Quick Hull Time |
| 25 | 0.755 | 1.365 |
| 250 | 4.835 | 2.989 |
| 1250 | 35.865 | 5.599 |
| 2500 | 112.215 | 11.305 |
| 5000 | 484.575 | 18.155 |

It is important to recognize while running the tests that the values of time fluctuate, and there are most likely many variables that stood in the way of making these times perfect. For example, storing strings and printing add an absurd amount of execution time.

As it pertains to how large a file can be for each algorithm, you can most certainly get away with the Brute Force algorithm until around 2500 points. At this point it isn’t beneficial to take the shortcut. Using Quick Hull would prove to be the viable with much larger file sizes.