Results Explain:

+ **Testing data**: NBA Players record (no duplicate) from Regular Season 2017 - 2018, using only total points (PTS) and total field goals attempted (FGA), so 2 dimensions for both data and queries.

+ **Parameters and Hashing**: We set parameters, using 4 hash tables/layers and within each hash table/layer there are 2 hash functions. The hashing method is set to be Angular Hashing. The algorithm will first use Angular Hashing to retrieve candidate data points against a query then re-ranking them using L2 (Euclidean) distance, then output the points and the distances.

+ **Process**: first apply Onion technique to extract convex hull – there are 5 layers in total for this testing data. For testing purpose, we only use first three layers. Both data and queries are normalized – it is not necessary but both data and queries should be bounded with unit sphere.

+ **Test**: We tested out three sample queries, little changes have been made for query\_2 from query\_1, same as from query\_2 to query\_3.

+ **Comparison**: I have compared it with naïve method – aggregating results and rank then output. To rank the results I used two methods:

* Euclidean distance between data points and queries, smaller distances indicate more similarity
* Using weight combination, query\_dimension\_1 \* data\_dimension\_1 + query\_dimension\_2 \* data\_dimension\_2 and rank query results in descending order – higher scores indicate more similarity.

In this case, since both data and queries are normalized in the unit sphere, a query for instance (0.4, 0.6) indicates

* Searching for the best scores after giving 0.4 weight for dimension-1 and 0.6 weight for dimension-2, or
* After normalization, it is (0.5547, 0.8321), which indicates that within an unit sphere, which of the data vectors are the closest to the direction of (0.5547, 0.8321)