1. Hardware verification
   1. What,why,how?
2. UVM
   1. Structure
   2. Coverages
3. DUT)brief explanation about it)
4. Implemented Verification environment
   1. UVM used classes
   2. Ref Model
5. Test Lines
   1. Basic - randomly generate:
      1. Number of points
      2. Points values
      3. Initial Centroid values

Insert these 3 parameters to DUT and REF Model, check for differences

Run this test 10 times?

* 1. Back to back - randomly generate:
     1. Number of points
     2. Points values
     3. Initial Centroid values

Get the result and do another iteration using it as initial centroids (with the same generated points)

Run this test 10 times?

* 1. Robustness - randomly generate:
     1. Number of points
     2. Points values
     3. Initial Centroid values

And run the algorithm multiple times in a row.

* 1. One iteration run - randomly generate eight data points which will be used also as the eight initial centroids, multiple times in a row. Verify that in all runs convergence is reached in one iteration and final centroids are equal to initial centroids.
  2. Threshold - randomly generate:
     1. Number of points
     2. Points values
     3. Initial Centroid values
     4. Threshold value

Insert these 3 parameters to DUT and REF Model, check for differences

* 1. Isolated centroid(“K value change”) - randomly generate:
     1. Number of points
     2. Points values
     3. Initial Centroid values
        1. Where one of the centroids is constrained to be far away from the all the data points. Verify its values does not change (no points are assigned to it)
        2. Where all of the centroids (except from one) are constrained to be far away from the all the data points. Verify their values does not change (no points are assigned to it)

1. Bug Fixes
2. Conclusions
3. Bibliography