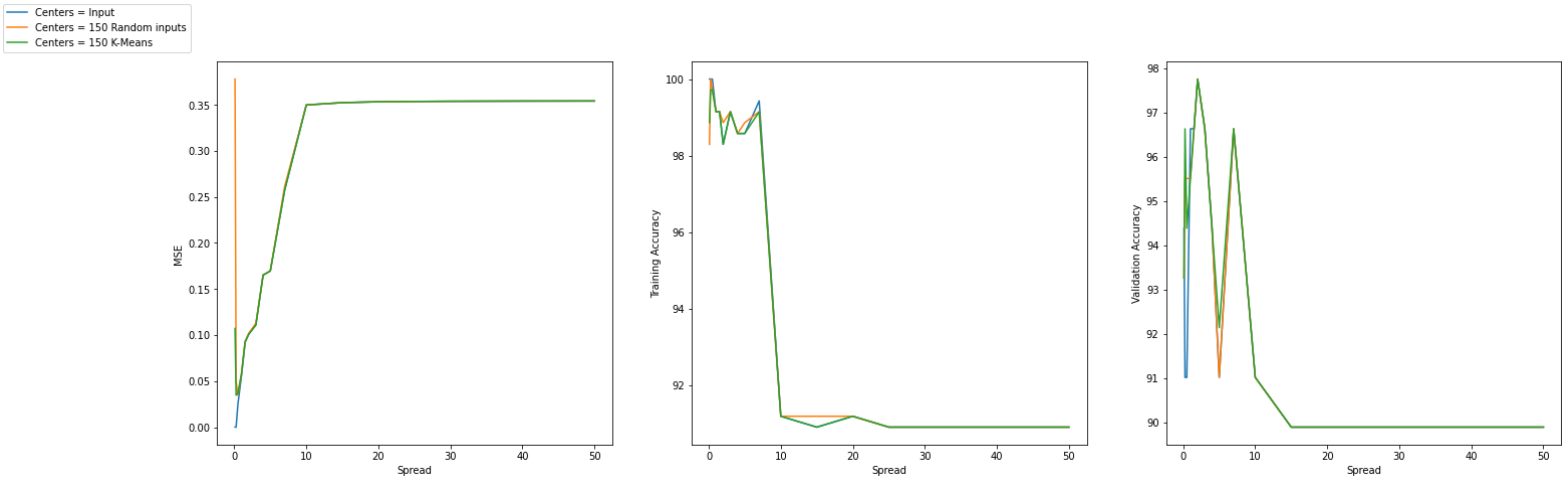
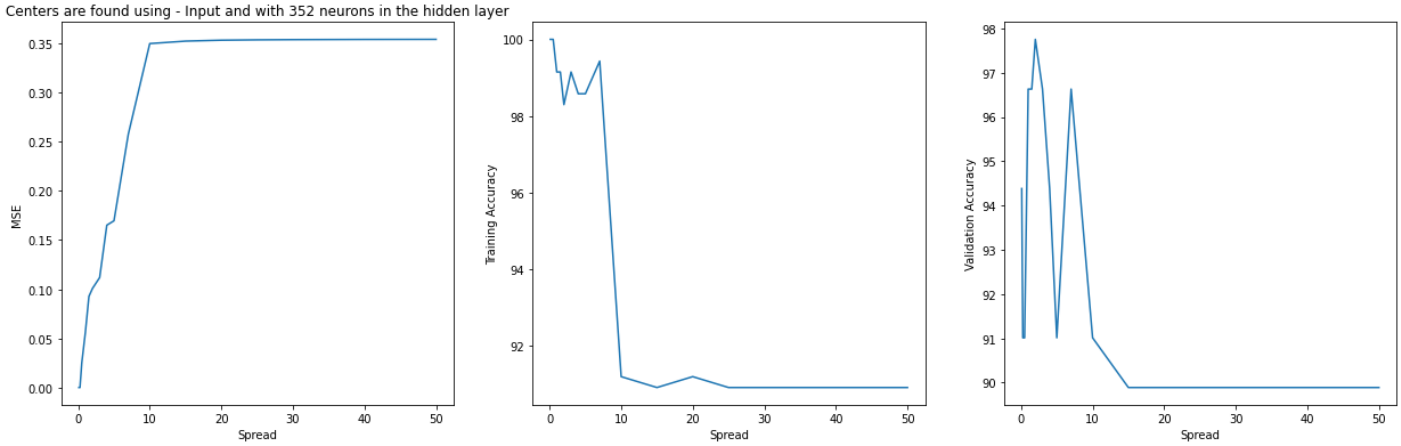
Report



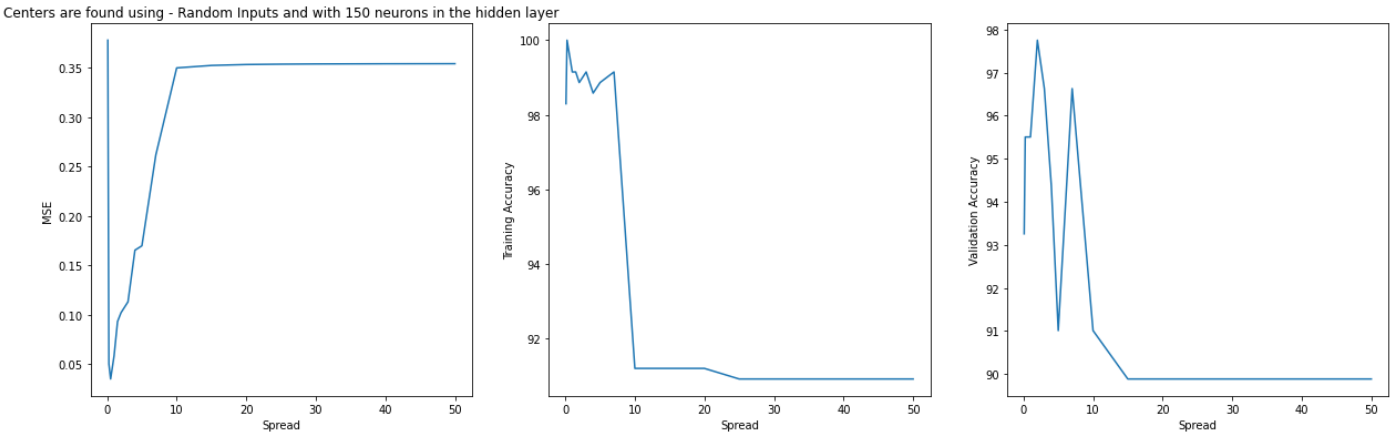
Following are the results achieved for different values of Spread:

Part 1] Centers = 352 Input Points



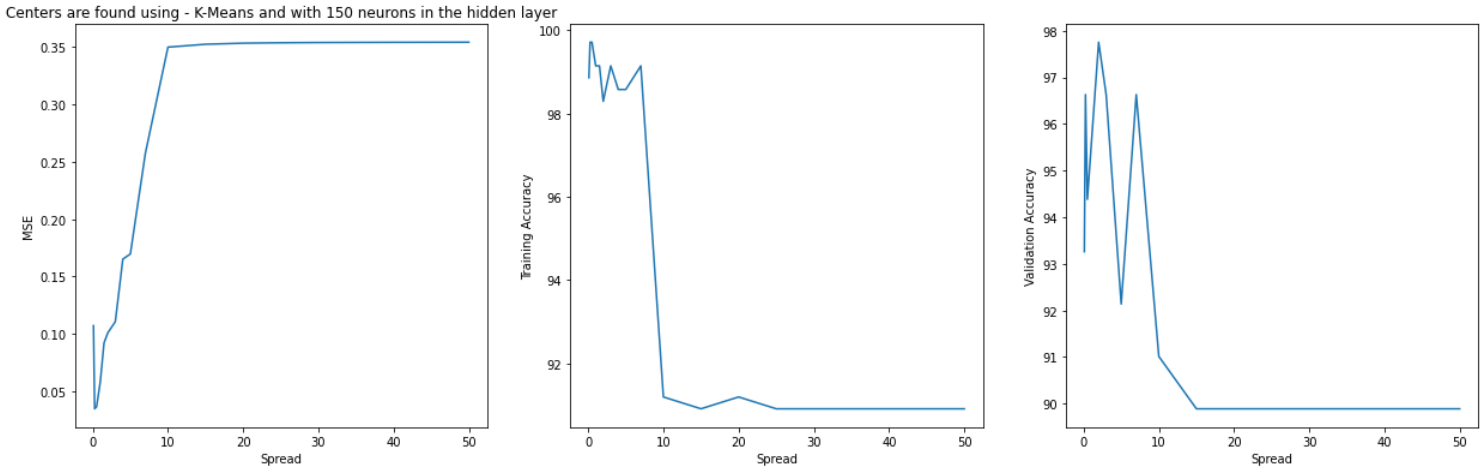
- Mean square Error keeps increasing with increase in 'Spread' value & after 'spread = 10' stays almost constant
- Training Accuracies have very little fluctuations initially, and then suddenly drop to around 90 for spread >= 10.
- Testing (Validation) accuracies follow a trend similar to training, except that the initial fluctuations are large. [The large initial fluctuations are expected, since the validation set may hold new inputs (which were not exposed earlier, during training)]

Part 2].b] Centers = 150 Random Input Points



- Mean square Error keeps increasing with increase in 'Spread' value (With the exception of 'spread = 0.1 to 0.5' where it actually drops between the 2 points) & after 'spread = 10' stays almost constant
- Training Accuracies – Results are again similar to those achieved in Part-1, except for 'spread = 0.1 to 0.5', where it actually goes up by a small margin.
- Testing (Validation) accuracies – Results are similar to those achieved in Part-1, although not exactly the same.

Part 2].c] Centers = 150 Points, Found using K-Means



- The results are similar to those achieved in Part 2.a] and for 'spread = 0.1 to 0.5' the change is relatively smaller in case of all 3 parameters.

Comparison in performance of the network:

- As is evident from the graphs, there is very little difference in performance between the 3 different configurations of centers.
- All the performance parameters (Mean Square Error (MSE), Training Accuracy in %, Validation Accuracy in %) are very similar in each of the configurations.
- A noticeable difference that we can see between the configurations is only for the initial values of spread. Otherwise, the results are very similar.

Comment on Spread Value:

- The above results show the importance of 'Spread' value in Radial Basis Function Neural Networks.
- The performance of the network largely varies depending on the spread value, compared to other parameters.
- A spread value between '1' and '2' appears to provide an efficient result, with good accuracies and low cumulative error.

Following Configuration has been used:

- spreads = [0.1, 0.25, 0.5, 1, 1.5, 2, 3, 4, 5, 7, 10, 15, 20, 25, 30, 40, 50]
- Radial Function = "Gaussian Kernel", Loss Function = "Mean Square Error"

Changing the Spread Values:

The choice of spread values can be changed by changing the value in '**spreads**' parameter inside the constructor call for **assignment** under **'Train and observe the Network'** section.

```
# -------------- Prepare Input Data for the network -----------------
a = assignment(center_methods = ["Input", "Random Inputs", "K-Means"], spreads = [0.1, 0.25, 0.5, 1, 1.5, 2, 3, 4, 5,
7, 10, 15, 20, 25, 30, 40, 50], test_validation_division = 8/10)
a.generate_input_data()
a.train_and_observe()
```

- Simply change the value of '**spreads**'

- **'center_methods'** parameter contains the name of the methods that should be implemented in sequence for the network configuration for Part 1, Part 2a, Part 2b respectively.