Ex3.1 Derive the formulas for (i) number of comparisons, and (ii) average-case number of swaps for bubble sort

1. Number of Comparisons

  - In bubble sort, the algorithm compares adjacent elements during each pass through the list.
  - In the first pass, there are n – 1 comparison.
  - In the second pass, there are n – 2 comparisons, and so on, until only 1 comparison remains in the final pass.
- **Formula**
  - Sum of comparison for each pass.
  - Total comparisons=$(n-1) + (n-2) + \cdots + 1 = (n(n-1))/2$
- **Complexity:**
  - Since the total number of comparisons grows proportionally to $(n(n-1))/2$, the complexity is $O(n^2)$.

2. Average-Case Number of Swaps

- **Explanation:**
  - A swap in bubble sort occurs when two adjacent elements are out of order.
  - The total number of swaps performed during a run of bubble sort is equivalent to the number of **inversions** in the array (an inversion is a pair of elements that are out of order).
- **Average-Case Analysis:**
  - For a random permutation of **n** elements, the expected number of inversions is $n(n-1)/4$
  - Therefore, on average, bubble sort performs approximately $n(n-1)/4$ swaps.
- **Complexity:**
  - Since the number of swaps also grows quadratically with the input size, the average-case complexity for swaps is $O(n^2)$.

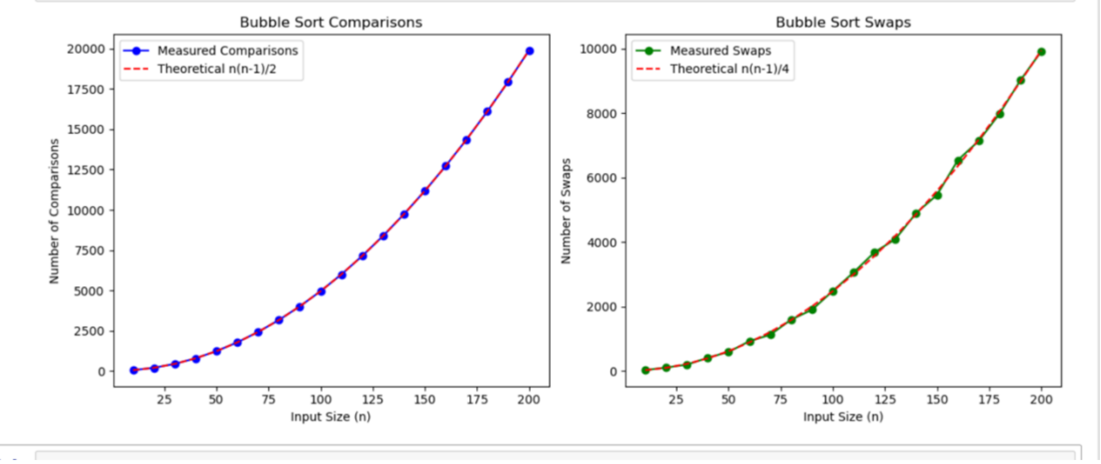
3. 4. Separately plot the results of #comparisons and #swaps by input size, together with appropriate interpolating functions. Discuss your results: do they match your complexity analysis?

Discussion:

- **Comparisons:**
  - The measured comparisons closely match the theoretical curve n(n–1)/2.
  - The interpolated function aligns well with the quadratic prediction, confirming the expected O(n²) behavior.
- **Swaps:**
  - The measured swaps similarly follow a quadratic trend.
  - Although slight fluctuations may occur due to randomness, the average trend aligns with n(n–1)/4, as predicted by the theoretical analysis.

Conclusion

- The plots validate the expected O(n²) complexity for both comparisons and swaps.
- Theoretical and measured results match, confirming Bubble Sort's inefficiency for large inputs.


Bubble Sort Comparisons
Measured Comparisons
Theoretical n(n-1)/2
Number of Comparisons
Input Size (n)
Bubble Sort Swaps
Measured Swaps
Theoretical n(n-1)/4
Number of Swaps
Input Size (n)