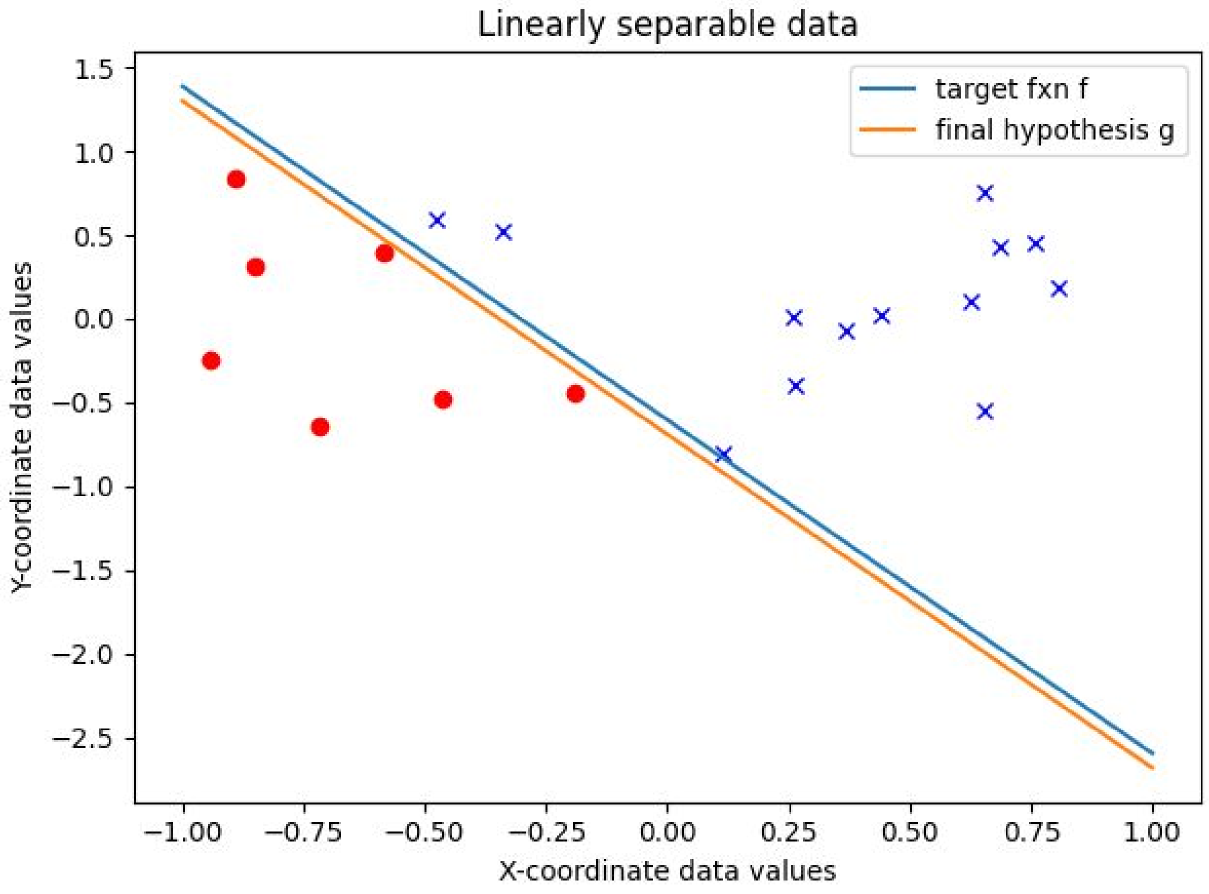
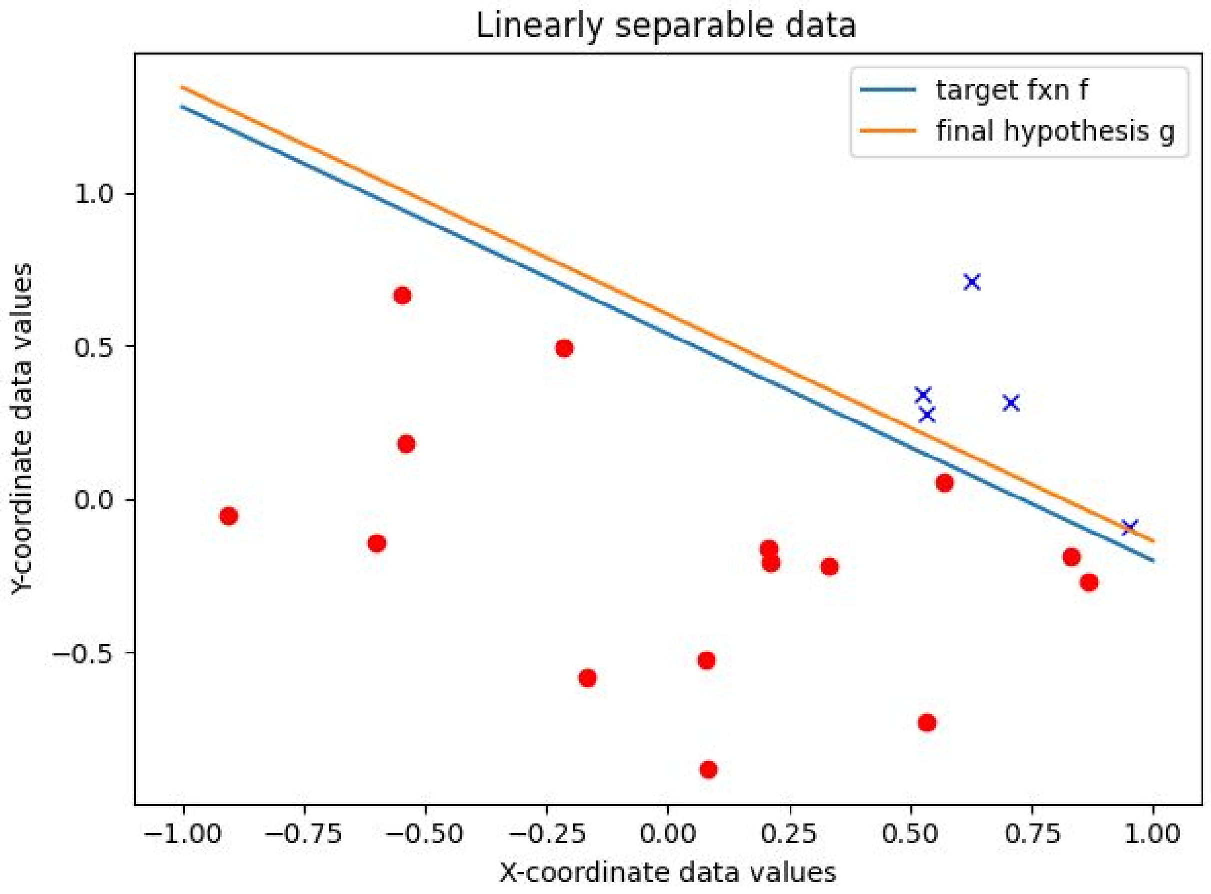
Results from parts A and B: (N=20, d = 2)



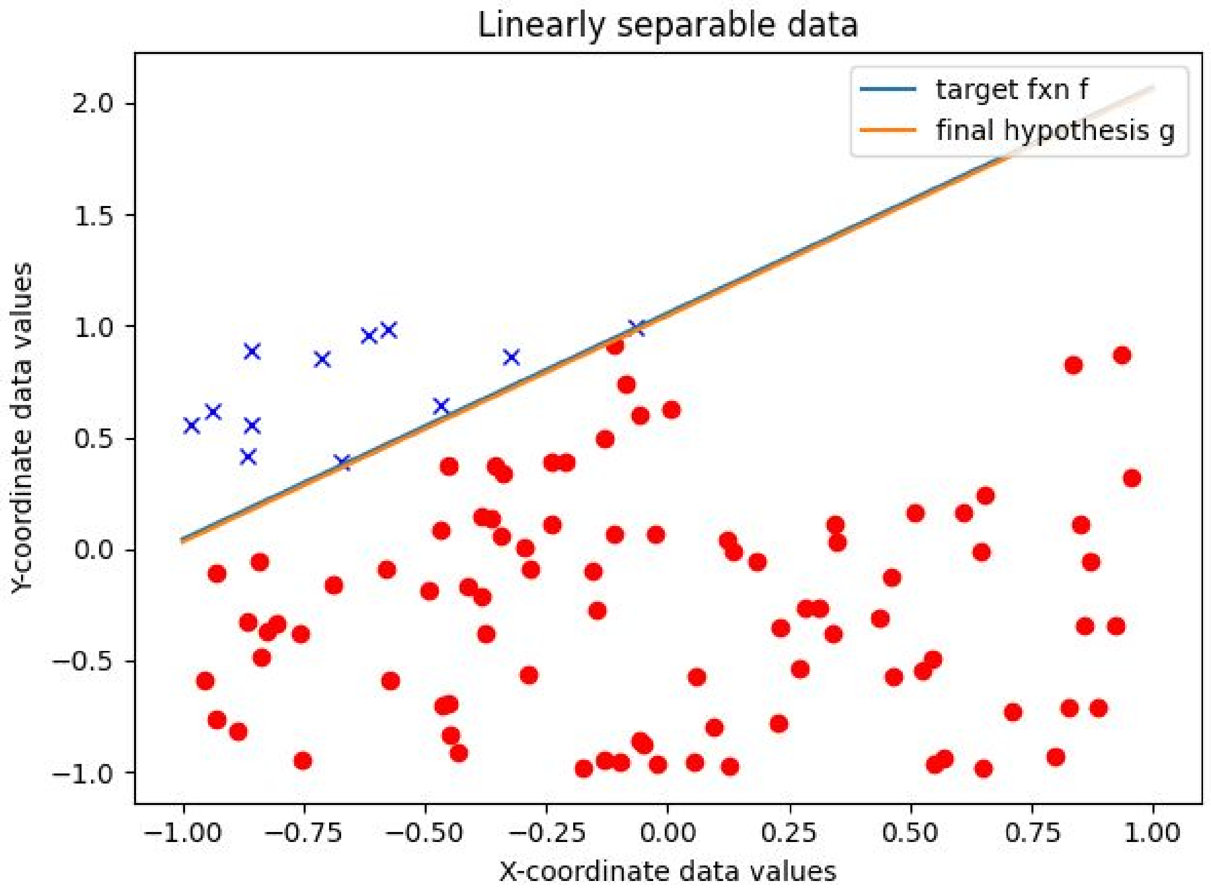
In this generated plot, the target function is in blue and the final hypothesis is in orange. It took the perceptron 28 updates before converging. The target function f in this case appears to be very close to the final hypothesis g.

Results from part C: (N=20, d = 2)



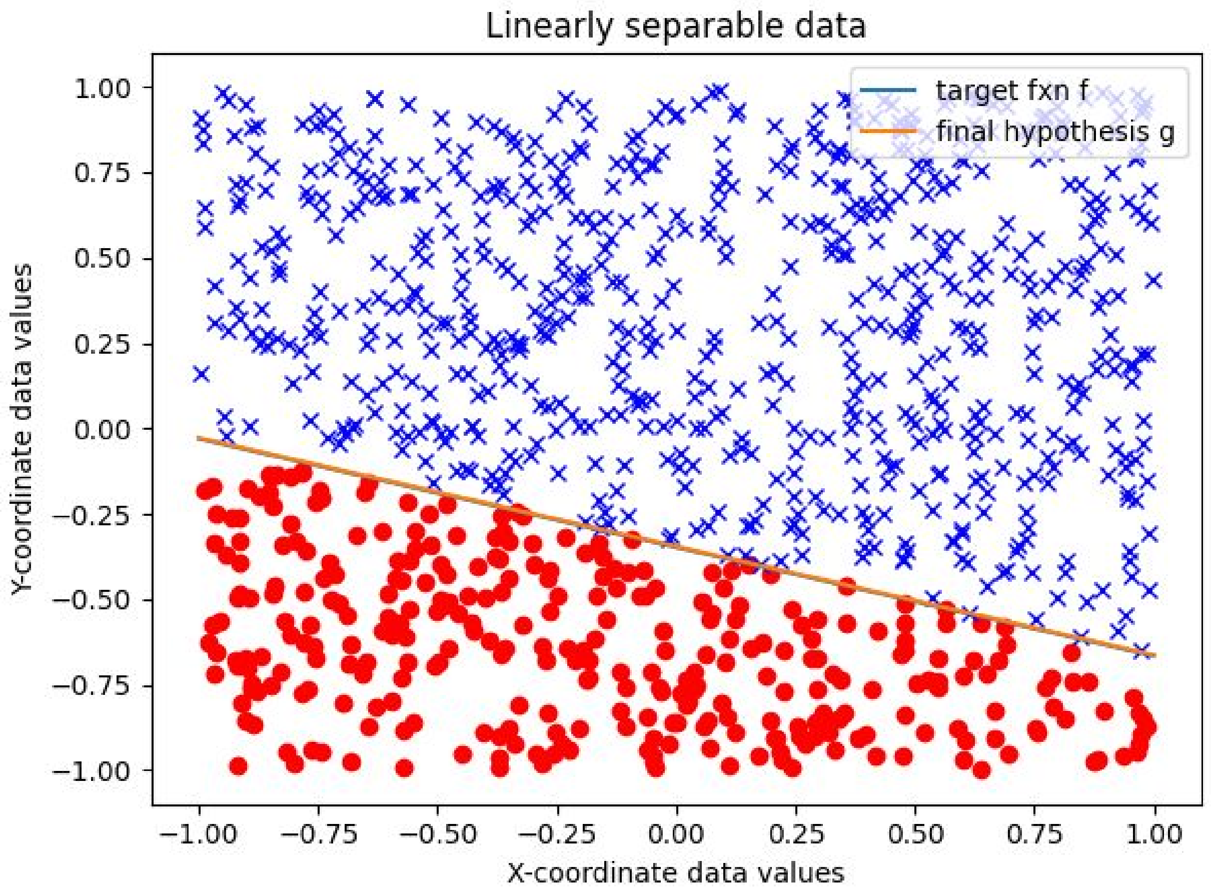
The perceptron was executed once more on a data set of size 20. This time, the perceptron took 16 updates before converging. Once again, the target function is very close to the final hypothesis, with a very minor increase in difference compared to part B.

<u>Results from part D: (N=100, d = 2)</u>



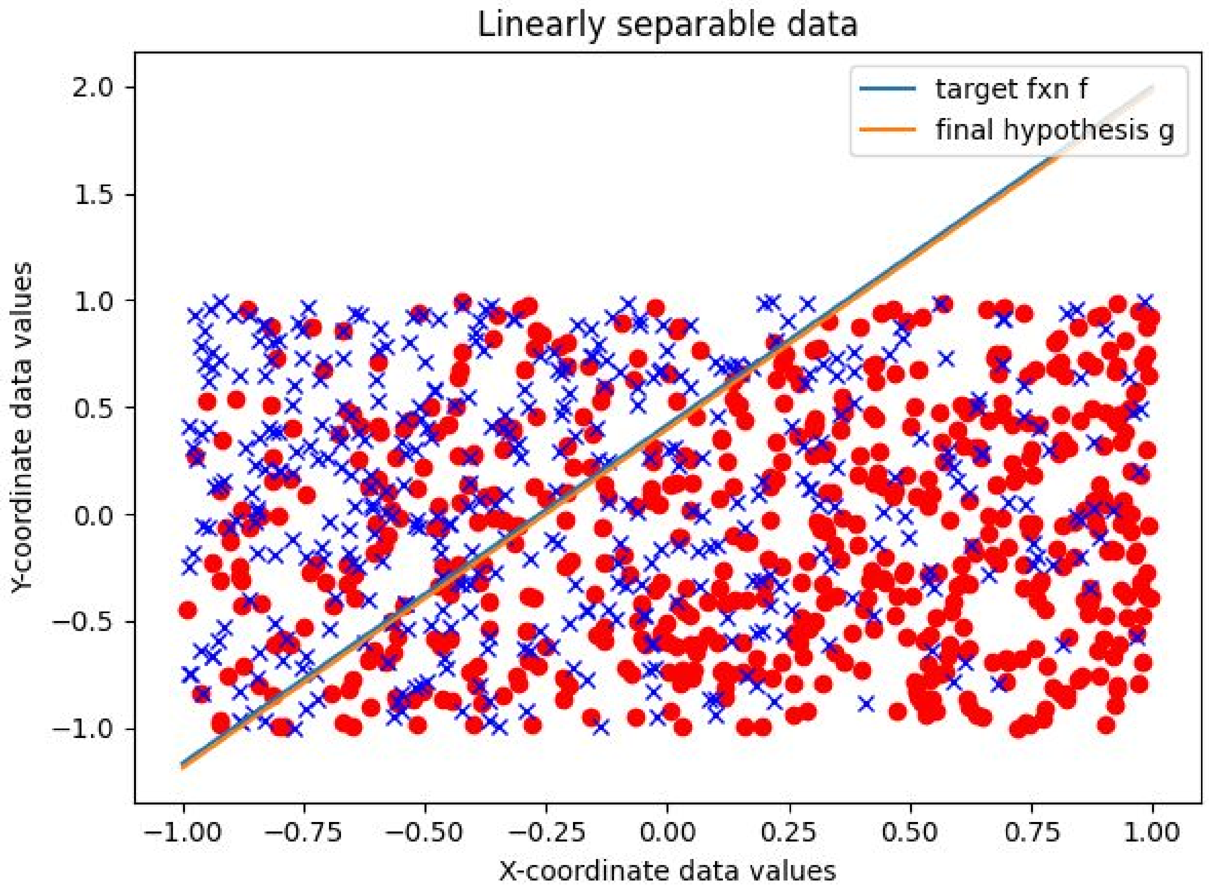
This time, the perceptron was executed with a data set of size 100. The target function and final hypothesis are very, very close together, especially compared to the results from part B. The perceptron converged after 126 updates.

<u>Results from part E: (N=1000, d = 2)</u>



In this case, the perceptron was executed with a data set of size 1000. The target function and final hypothesis are indistinguishable from each other, which is really good. This is an immense improvement from the results of part B. In this case, the perceptron converged after 667 updates.

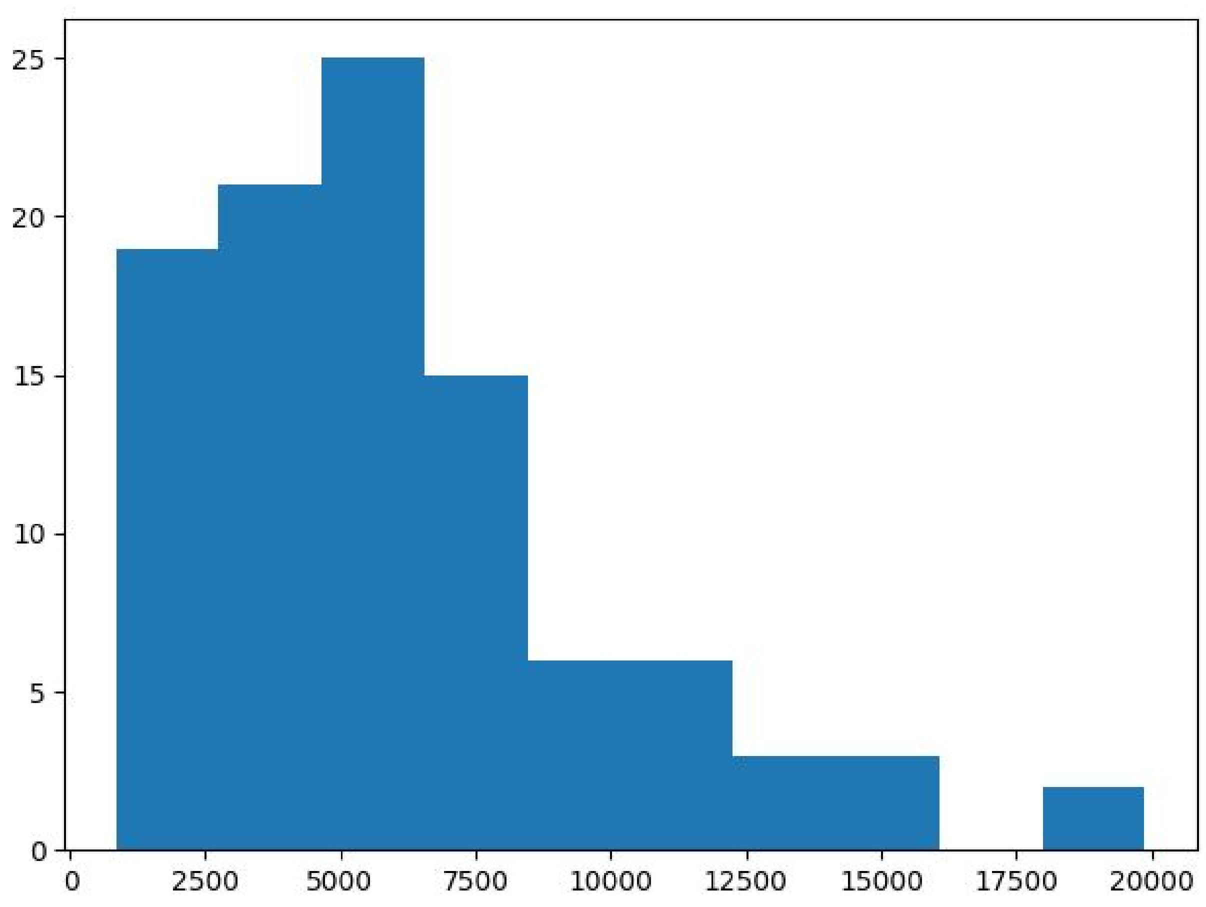
Results from part F: (N=1000, d = 10)



In this case, the perceptron was run on a data set of size 1000 with a dimension of 10. It took the algorithm 8238 updates to converge, which is much higher than any of the instances where the perceptron was executed on a 2-dimensional data set.

Results from part G: (N=100, x(t) picked randomly)

Frequency vs. Number of updates for convergence



Note: This part of the code took my computer about an hour to run. For this reason, I have commented out the code that accompanies part G. In order to test this part for a grade, please uncomment that section of my code. Thank you!

Part H: Runtime Analysis

The runtime of the perceptron algorithm increases very, very slightly with respect to N, to a point where it is nominal (milliseconds). However, the runtime largely depends on d. When d increased from 2 to 10, it took my computer a couple minutes to run the code. Furthermore, when I set d to 10 and ran 100 iterations of the perceptron, it took my computer about an hour to run the code.