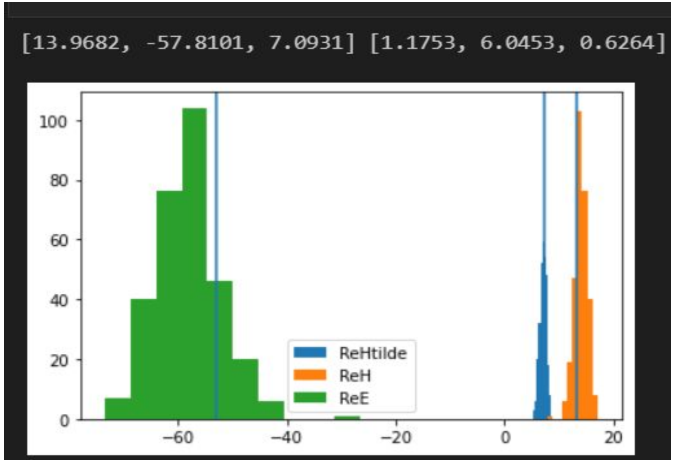
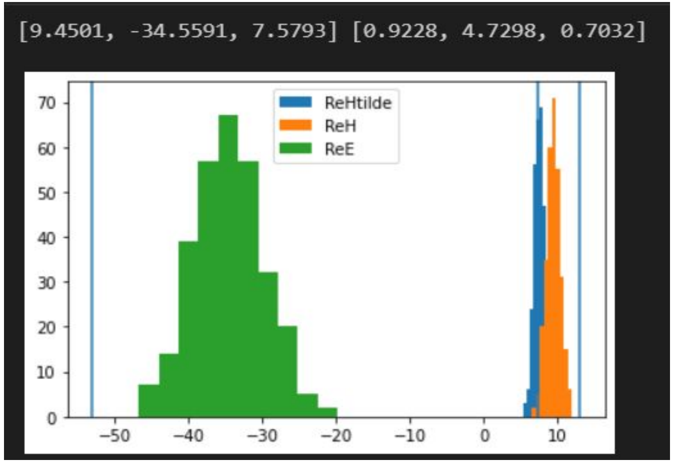
01/04/2022

Aaryan

Last Meeting Discussion

- Nicks Method - No Weight Reset
  a. ✓: Good Estimates
  b. X: Doesn't really propagate error, Wider Distribution
- Pure Bootstrapping - Resetting Weights to Random Initial Configuration
  a. ✓: Mathematically sound, Narrower Distribution, Error Propagation
  b. X: Horrible Estimates (especially for ReE)

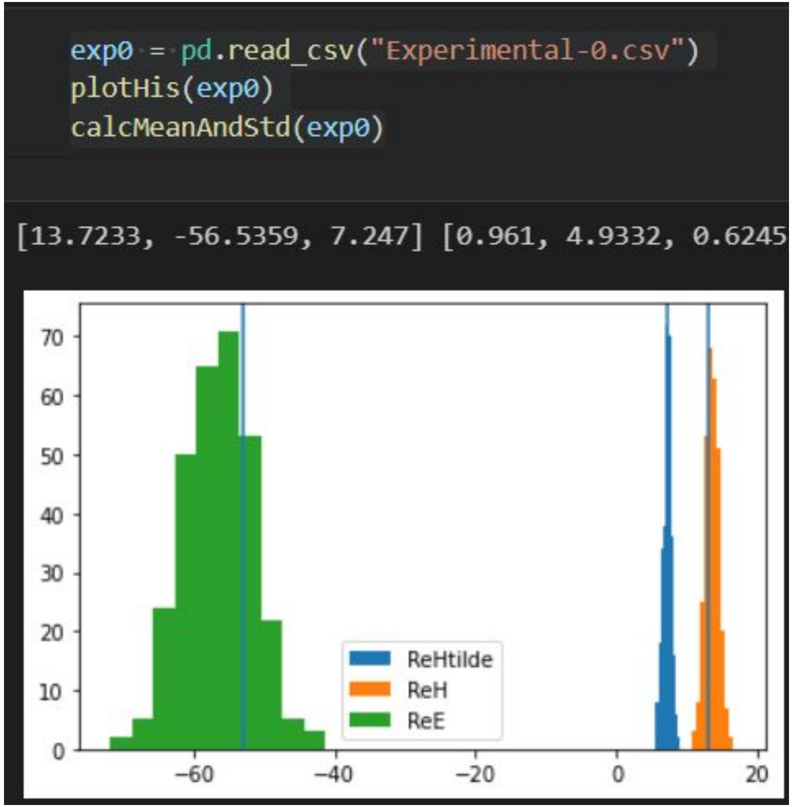




Experimental Method - Combining Two Methods

- Get random weights
- Train them a bit on the input data
    - 10% of total replicas
- Save those weights as the starting weights
- Do bootstrapping with those weights (resetting weights after each replica)

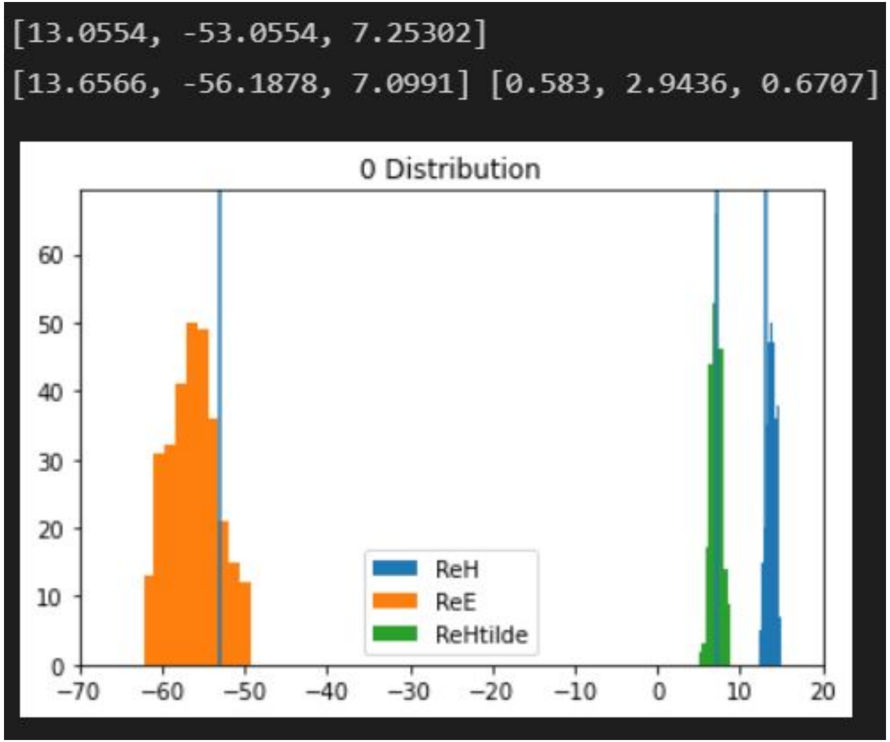
Saves a lot of computation time as each replica doesn’t need 2500 epochs and still gets good accuracy with lower deviation

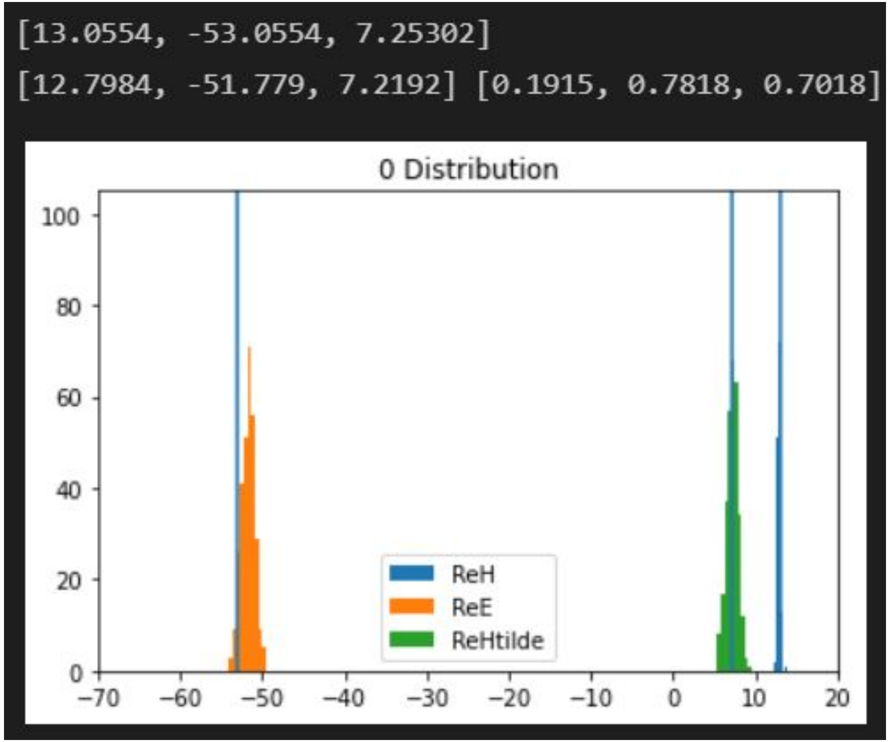


Code Changes

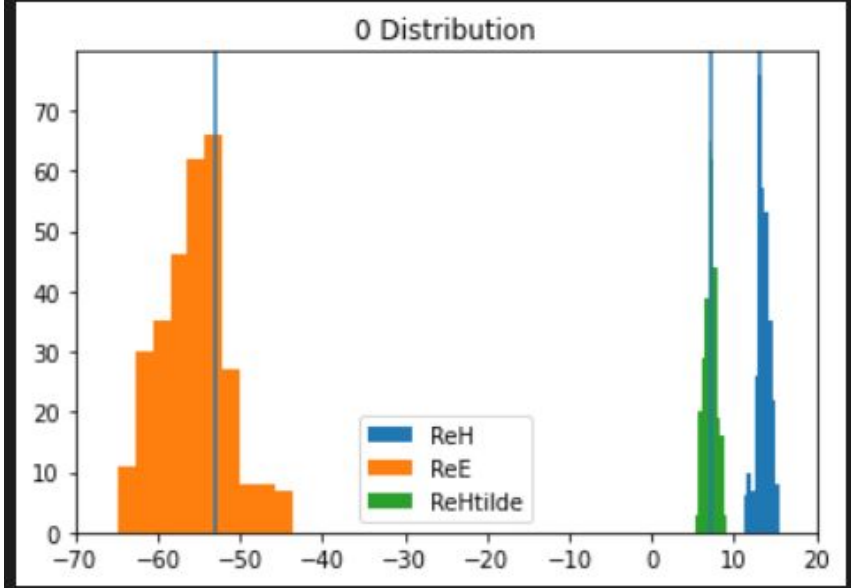
```
#Creating a model that has better starting conditions --> getting weights slightly better trained
#10%
for learningIteration in range(int(numSamples/10)):
        globalModel.fit([setI.Kinematics, setI.XnoCFF], setI.sampleY(),
                    epochs=2500, verbose=0)

Wsave = globalModel.get_weights()

#Using unrelated bootstrapping method
for sample in range(numSamples):

        globalModel.set_weights(Wsave)

        globalModel.fit([setI.Kinematics, setI.XnoCFF], setI.sampleY(),
                    epochs=300, verbose=0)

        cffs = cffs_from_globalModel(globalModel, setI.Kinematics)

        for num, cff in enumerate(['ReH', 'ReE', 'ReHtilde']):
            results.loc[sample, cff] = cffs[num]
```

Reproducibility - Set 0 with Same Settings

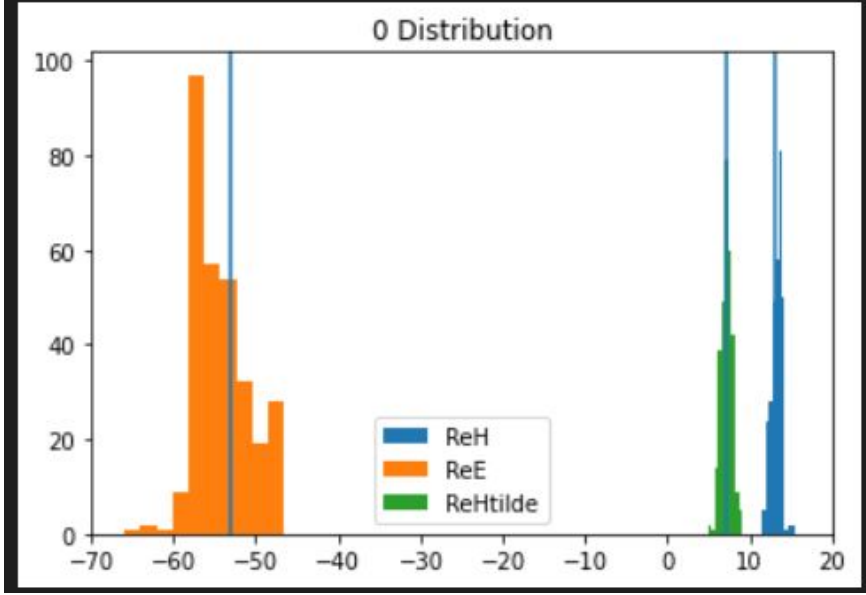




```
[13.0554, -53.0554, 7.25302]
[13.5353, -55.5744, 7.1366] [0.8061, 4.1507, 0.7153]
```




```
[13.0554, -53.0554, 7.25302]
[13.2787, -54.2628, 7.1756] [0.6467, 3.3251, 0.6656]
```

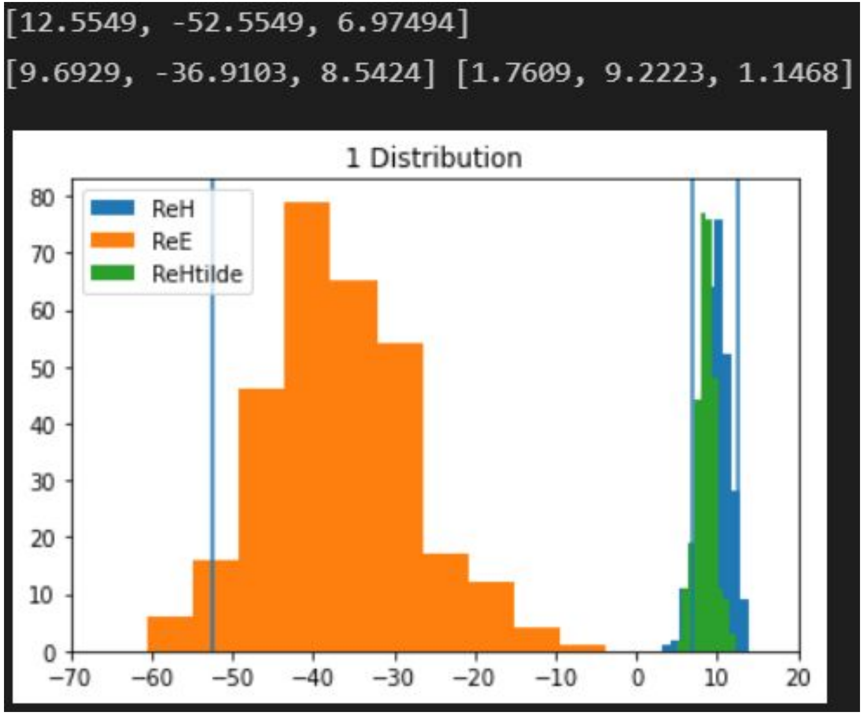


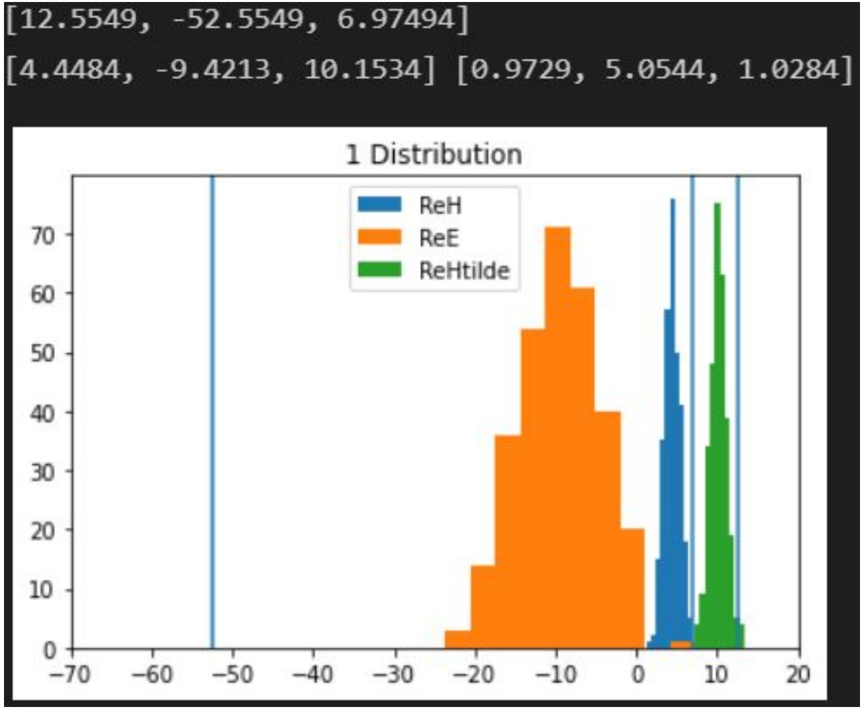
Takeaways?

- Method does not produce same distributions (inconsistent)

- Accuracy and spread are still much better than pure bootstrapping or method 2 implementation

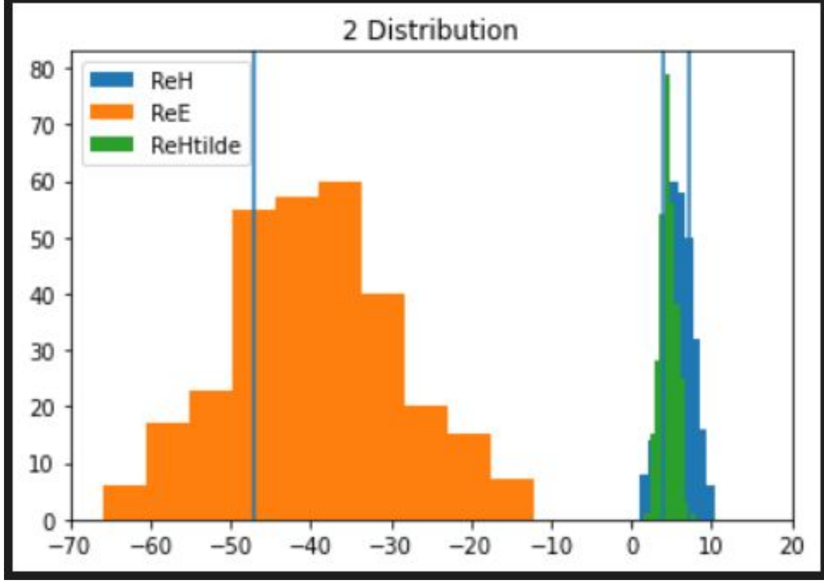
Control - Set 1



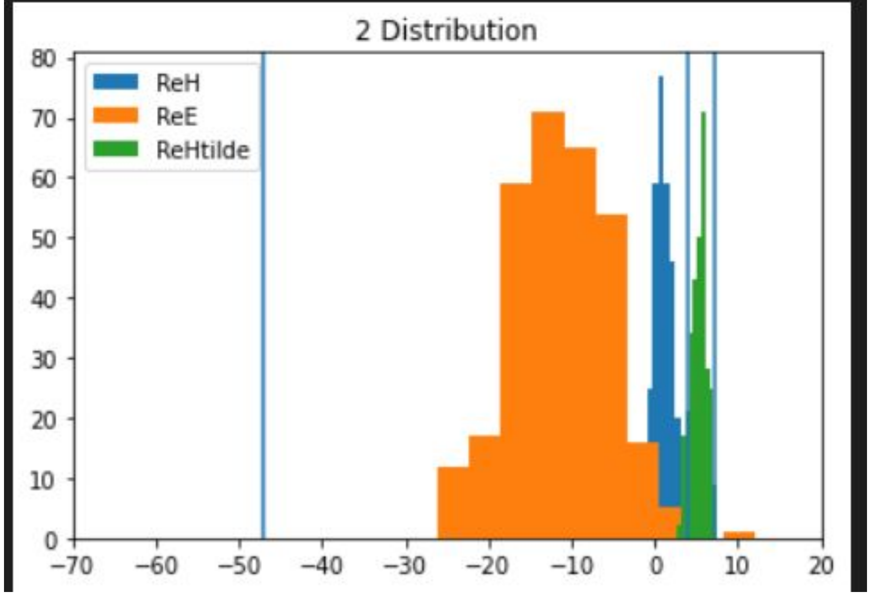


Control - Set 2

```
[7.22424, -47.2242, 4.01347]
[5.8964, -39.5538, 4.6308] [1.8475, 10.5635, 1.0519]
```

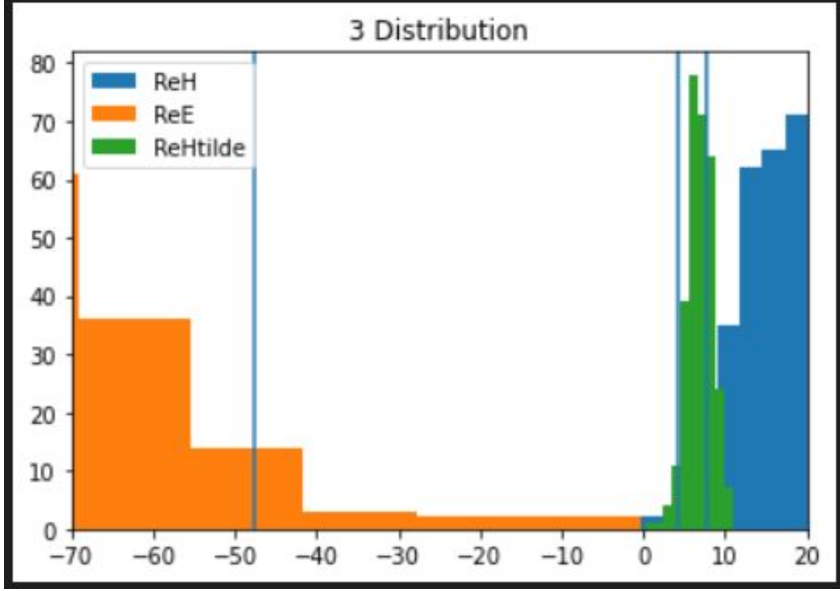



```
[7.22424, -47.2242, 4.01347]
[1.0002, -11.3166, 5.325] [1.0407, 5.9148, 0.9662]
```

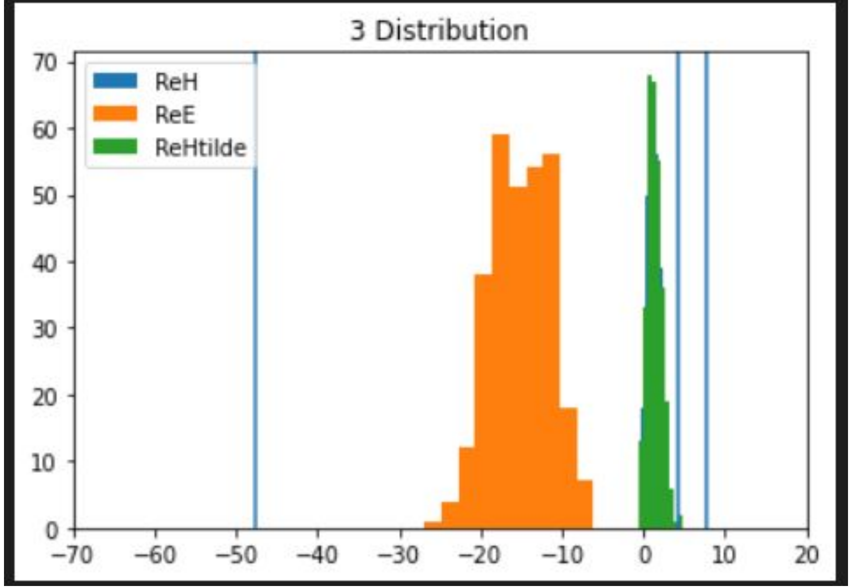


Control - Set 3

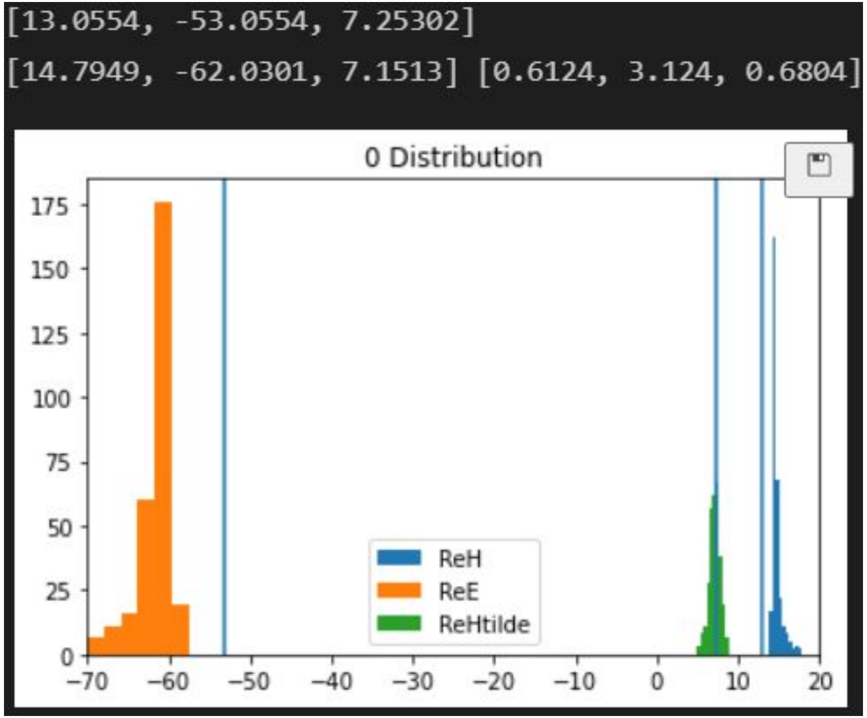
```
[7.65272, -47.6527, 4.25151]
[15.6387, -88.2363, 6.9415] [4.4156, 22.256, 1.6134]
```

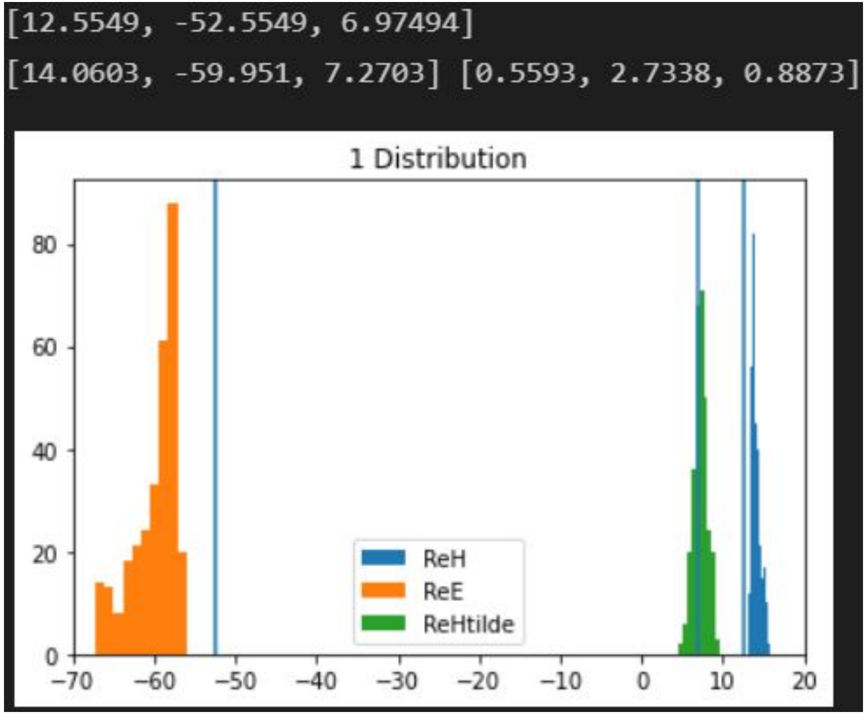



```
[7.65272, -47.6527, 4.25151]
[1.2076, -15.088, 1.4335] [0.7542, 3.6452, 0.9155]
```

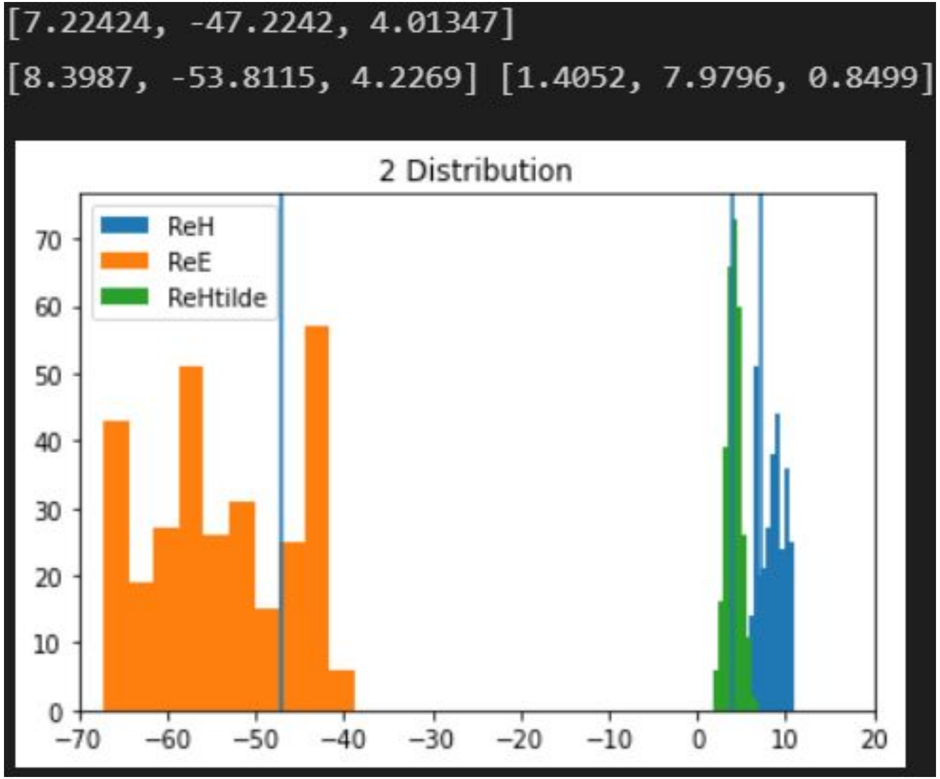


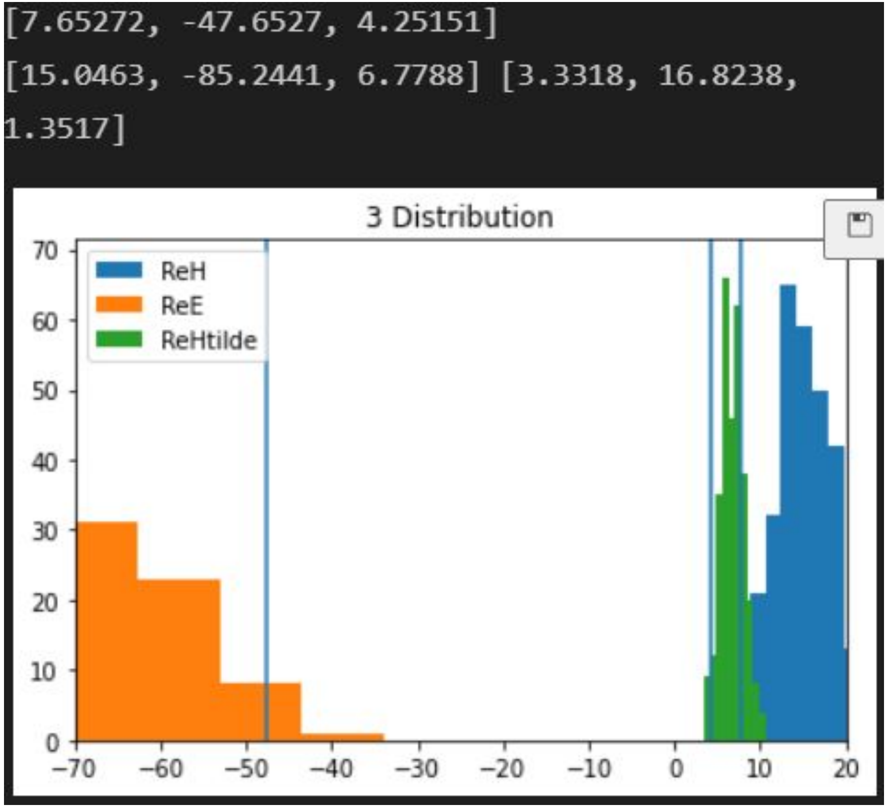
Experimental Set: 25% Initial Training





Experimental Set: 25% Initial Training





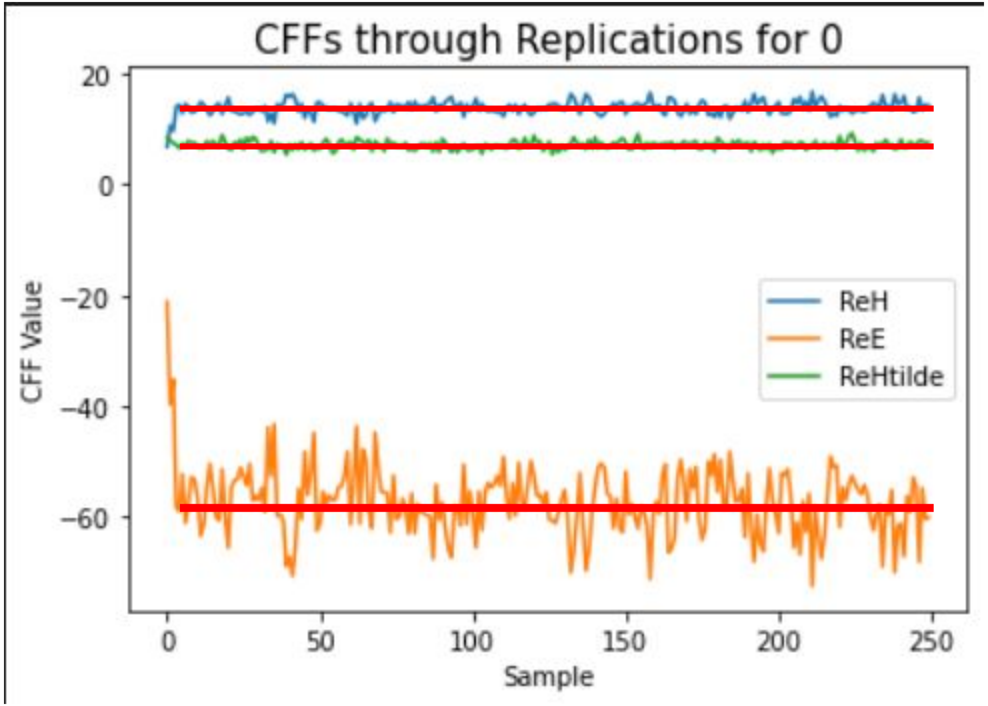
Other Percentages + Observations

- Tried other initial training percentages, no real improvement
  - 10%, 25%, 33%, 50%

- Set 0 usually had the best fit while Set 3 usually had the worst fit (farthest off mean and widest distribution)

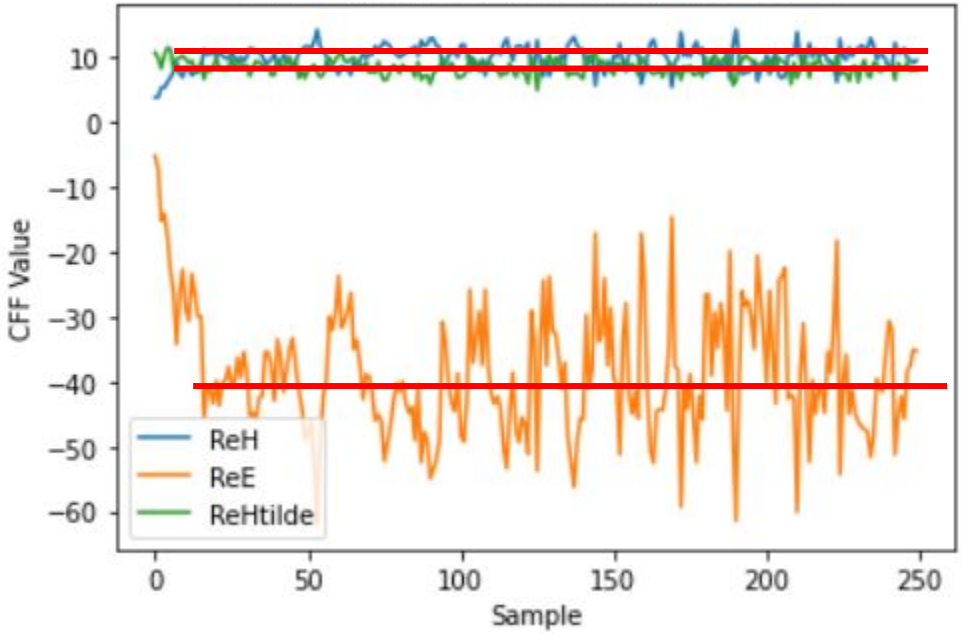
- Questions:
  - Does CFF remain stagnant after a certain number of learning iterations?
  - Are certain sets inherently have more fluctuation in predictions?

CFFs Through Different Sets



- Set 0 has good fits because CFF prediction stays relatively stagnant at the ‘correct’ value
- Arrives at the correct value fairly quickly (within 10 learning iterations)

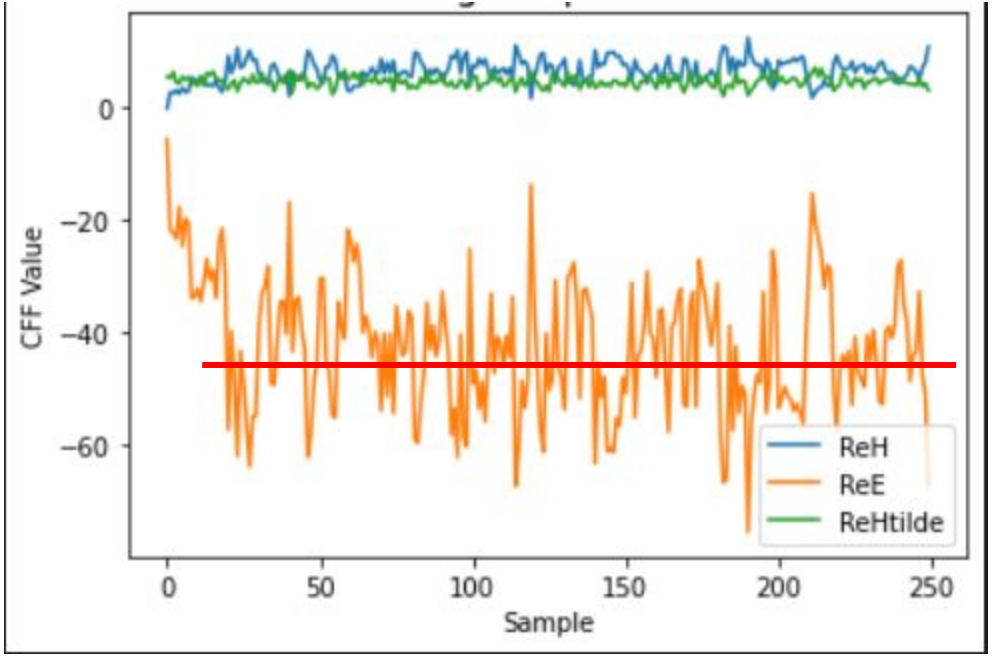
CFFs Through Different Sets



- Set 2 has even wider fluctuations for ReE

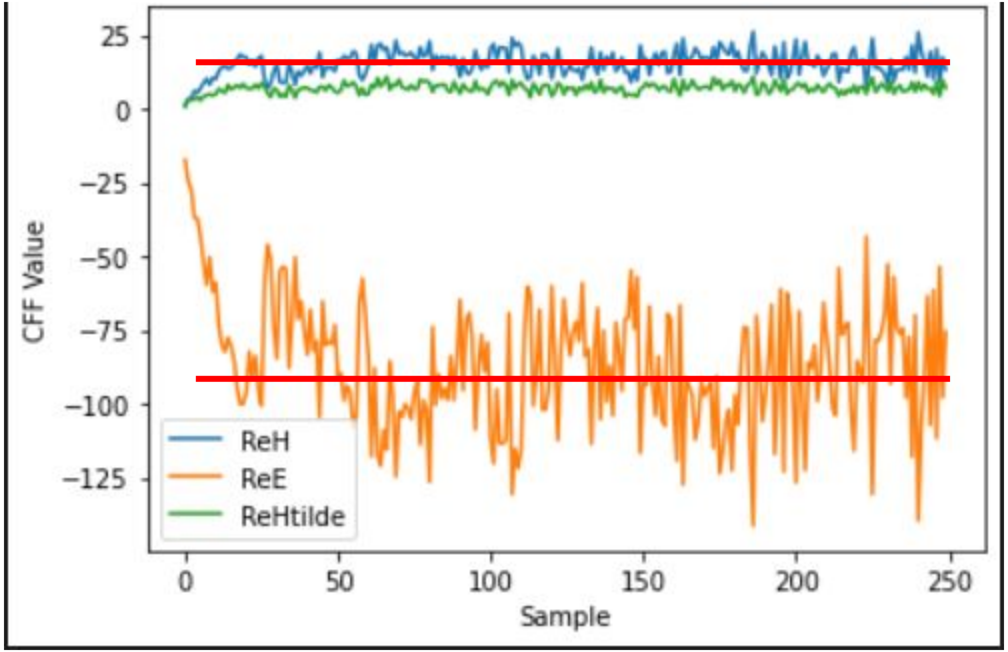
- ReE fluctuates around the correct value

CFFs Through Different Sets



- Set 1 has much wider fluctuations so it is not feasible to choose one model to take weights from (could take from one of the fluctuations)
  - Weight averaging?
- ReE fluctuates around -40 when the real value is around -50
- Why are there such wide fluctuations?
  - Learning rate too high?
  - Overfitting?

CFFs Through Different Sets



- Large fluctuations around -85 for ReE (correct value ~ -47)
  - Explains why Set 3 ReE has such a large standard deviation

- What about set 3 makes it so far off from the correct value?

Improve Consistency in Method

- Instead of using the weights of the last model in the initial learning stage, we use the averaged weights of the last five models in the learning stage
  - Decreases chance that a largely fluctuated weight will be used as WSave

- Implement a decay in the learning rate to see whether it impacts the extent of the fluctuations
  - If fluctuations still are large then the value of ReE has minimal impact on loss function (ReE can be anything and loss won't be impacted as much)