11/29/2021

Aaryan

Last Meeting Discussion

```python
def produceCFFs(numReplicas, data, Wsave):
    '''
    :param numSamples: number of replicas to produce
    :param data: whole DvcsData
    :param Wsave: saved weights

    :returns: numpy array of shape (numSets, numReplicas, 3)
    '''

    by_sample = []

    for i in tqdm(range(max(data.df['#Set'])+1)):

        globalModel.set_weights(Wsave) # reset weights to original value

        setI = data.getSet(i) #DvcsData object containing specific set

        by_set = []

        for sample in range(numReplicas):


            globalModel.fit([setI.Kinematics, setI.XnoCFF], setI.sampleYforInterference(), #  ue interference term
                            epochs=2500, verbose=0)

            cffs = uts.cffs_from_globalModel(globalModel, setI.Kinematics) # get cffs from middle model

            by_set.append(cffs)

        by_sample.append(by_set)

    return np.array(by_sample)
```

Weights being reset for each new sample

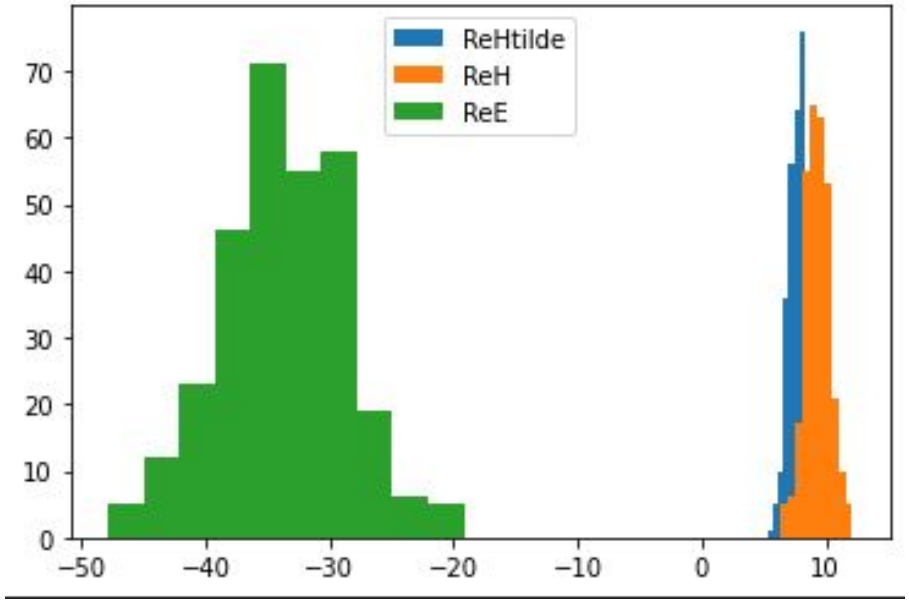
Weights being continually updated for each replica

Resetting Weights For Each Sample

Samples: 300

Means:

- ReH: 9.28
- ReE: -33.67
- ReHtilde: 7.69

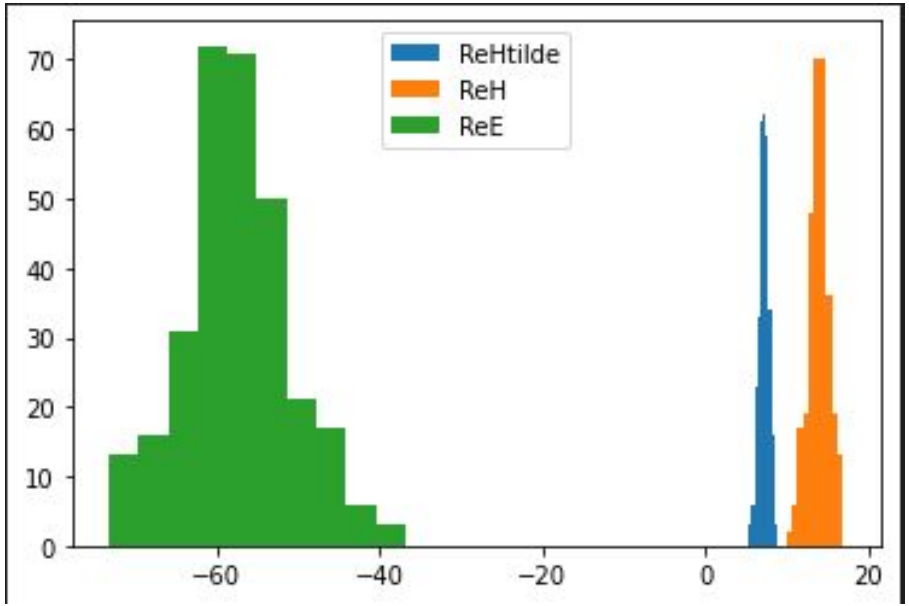


Weights not Reset After Each Sample

Samples: 300

Means:

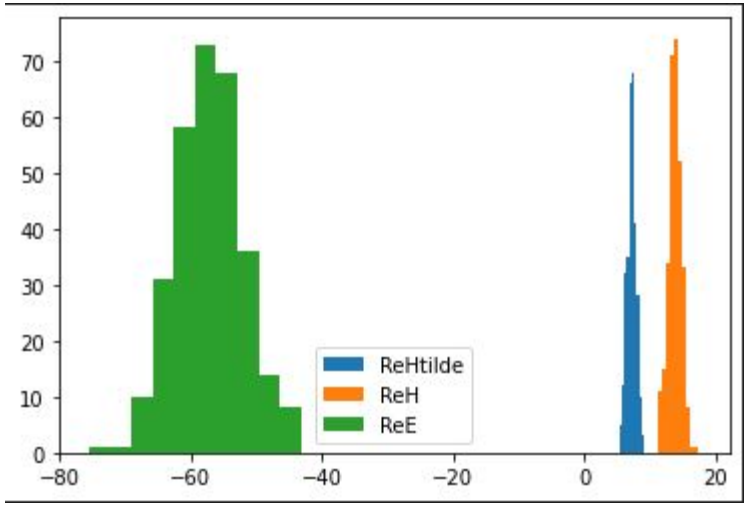
- ReH: 13.90
- ReE: -57.44
- ReHtilde: 7.16



Middle Ground - Resetting For Each Sample After Optimal Starting Weights are Found

```
#Creating a model that has better starting conditions --> getting weights slightly better trained
for learningIteration in range(15):
        globalModel.fit([setI.Kinematics, setI.XnoCFF], setI.sampleY(),
                        epochs=2500, verbose=0)

Wsave = globalModel.get_weights() #Getting Weights from better trained models

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

        globalModel.set_weights(Wsave)

        globalModel.fit([setI.Kinematics, setI.XnoCFF], setI.sampleY(),
                        epochs=2250, verbose=0)

        cffs = cffs_from_globalModel(globalModel, setI.Kinematics)

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

Middle Ground - Resetting For Each Sample After Good Starting Weights are Found



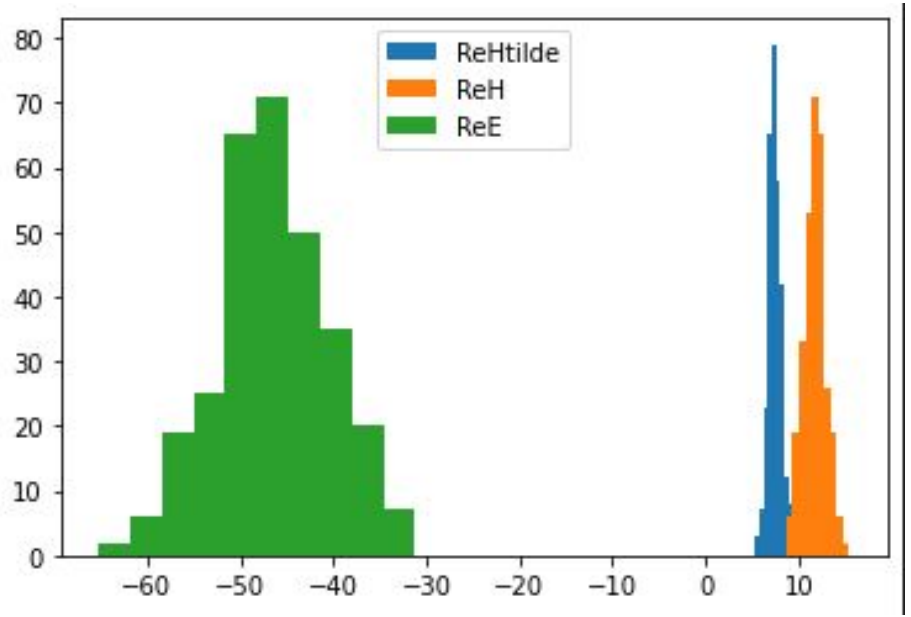
Maybe Resetting After Each Sample Needs More Time?

Samples: 300

#Epochs: 5000

Means:

- ReH: 11.78
- ReE: -46.52
- ReHtilde: 7.41



Full Average Comparison

- Reset to Random Starting Weights After Each Sample
  - 9.2823 -33.6674 7.6931
- No Resetting of Weights at All
  - 13.902 -57.4433 7.1559
- Resetting to ‘trained/optimal’ weights after each sample
  - 13.8245 -57.0429 7.173
- Reset to Random Starting with More Epochs Per fitting
  - 11.7781 -46.5203 7.4143
- Actual True Values
  - 13.0554 -53.0554 7.25302

Benefits of Resetting to Trained/optimal weights:

- Much quicker (not starting from scratch so less epochs required)
  - Weights are already in a good starting position to then fit to each sample and predict instead of starting from a fully random weight initialization