Cole Weis – 11/12

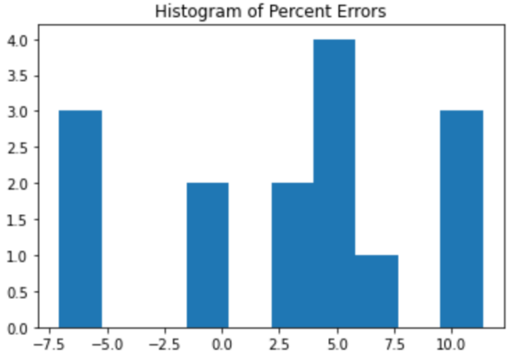
CFF Bounds Local Fit Experimentation

(Initial analysis of possible overfitting to F)

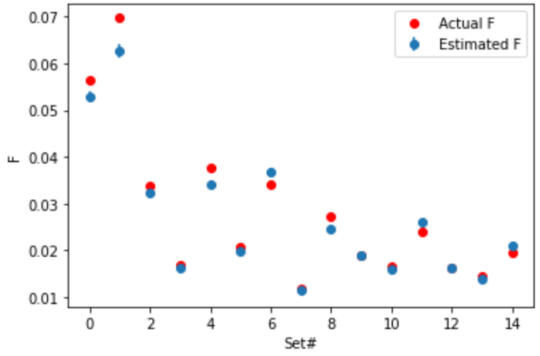
Propagated Fs at 180

```
[44] y_yhat, err = uts.y_yhat_errFs(results, data)
     uts.evaluate(y_yhat)
```

```
Mean percent error:  5.74123579141666
RMSE:  0.0025506729448490236
RMSE w yhat=mean:  0.015855649879790133
R-squared:  0.9741213595508924
```



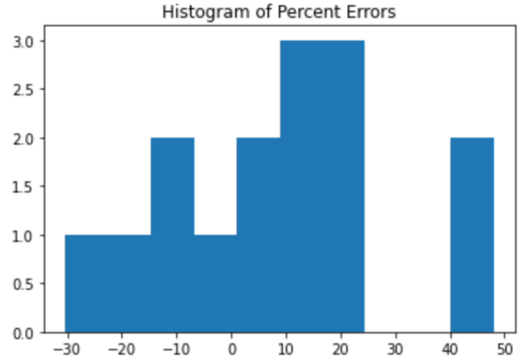

```
[45] uts.plotError(y_yhat, err, "F")
```



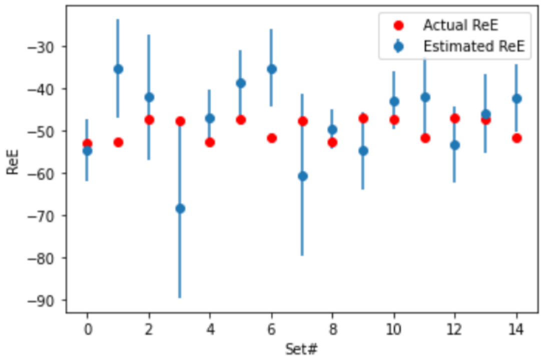

```
39] y_yhat, err = uts.y_yhat_errCFFs(data, results, 1)
```

```
40] uts.evaluate(y_yhat)
```

```
Mean percent error:  18.940298870034535
RMSE:  10.320108481902537
RMSE w yhat=mean:  2.525480507907797
R-squared:  -15.698616205656997
```




```
41] uts.plotError(y_yhat, err, "ReE")
```

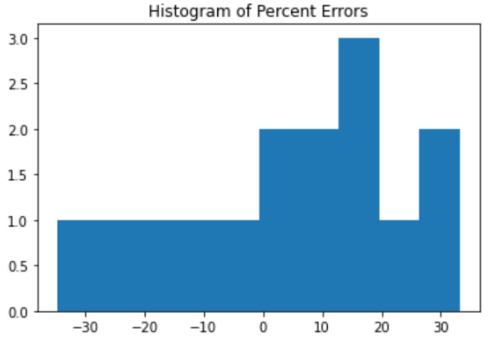


ReH

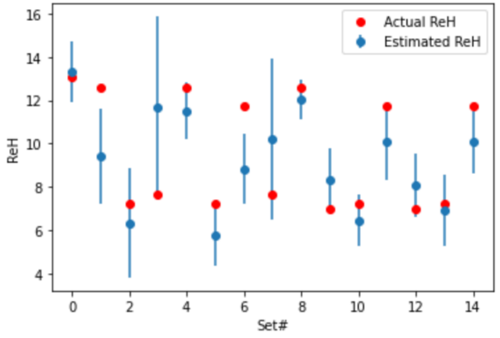
```
[36] y_yhat, err = uts.y_yhat_errCFFs(data, results, 0)
```

```
[37] uts.evaluate(y_yhat)
```

```
Mean percent error:  17.284864788557567
RMSE:  1.910664625564968
RMSE w yhat=mean:  2.5254628436780107
R-squared:  0.42761669285887705
```



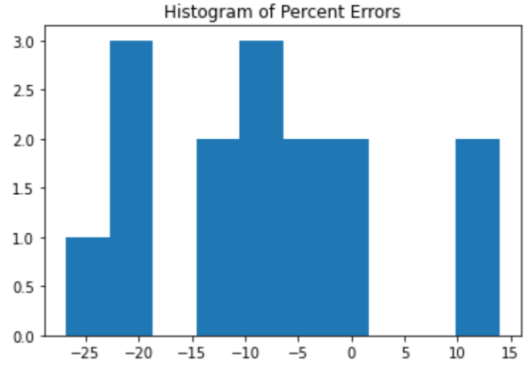

```
[38] uts.plotError(y_yhat, err, "ReH")
```



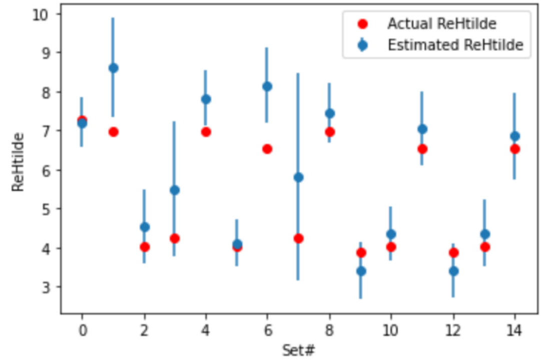
ReHtilde

```
[42] y_yhat, err = uts.y_yhat_errCFFs(data, results, 2)
     uts.evaluate(y_yhat)
```

```
Mean percent error:  11.761022042351389
RMSE:  0.8809442282390986
RMSE w yhat=mean:  1.4030345621243816
R-squared:  0.6057605411612466
```




```
[43] uts.plotError(y_yhat, err, "ReHtilde")
```

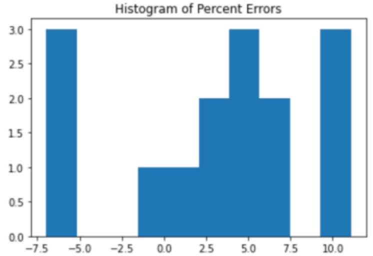


With no weight sharing between sets

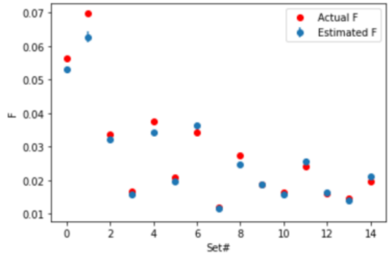
Propagated Fs at 180

```
[ ] y_yhat, err = uts.y_yhat_errFs(results, data)
    uts.evaluate(y_yhat)
```

```
Mean percent error:  5.83080037363791
RMSE:  0.0025002300670507937
RMSE w yhat=mean:  0.015855649879790133
R-squared:  0.9751348059642975
```




```
[ ] uts.plotError(y_yhat, err, "F")
```

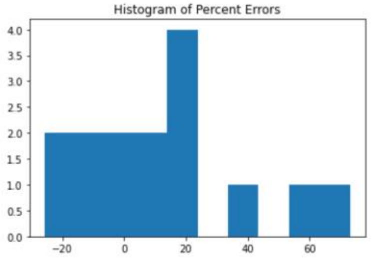


ReE

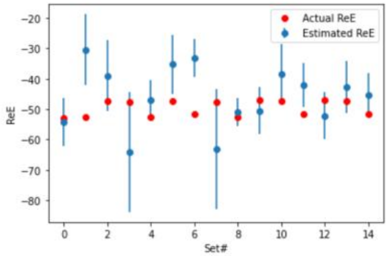
```
[ ] y_yhat, err = uts.y_yhat_errCFFs(data, results, 1)
```

```
[ ] uts.evaluate(y_yhat)
```

```
Mean percent error:  22.605824725626466
RMSE:  11.151278820278359
RMSE w yhat=mean:  2.525480507907797
R-squared:  -18.496708918080845
```



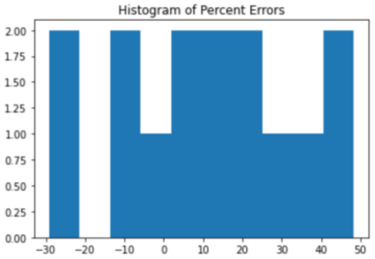

```
[ ] uts.plotError(y_yhat, err, "ReE")
```



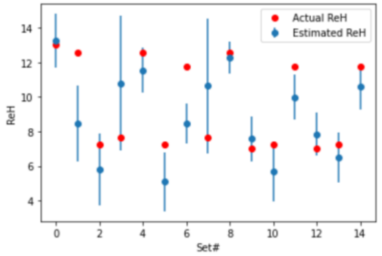

```
[ ] y_yhat, err = uts.y_yhat_errCFFs(data, results, 0)
```

```
[ ] uts.evaluate(y_yhat)
```

```
Mean percent error:  20.687183134096795
RMSE:  2.04410108357398
RMSE w yhat=mean:  2.5254628436780107
R-squared:  0.3448771154006801
```



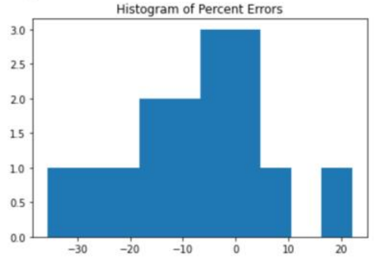

```
[ ] uts.plotError(y_yhat, err, "ReH")
```



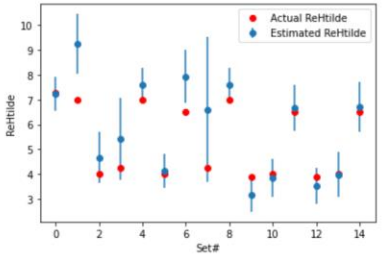
ReHtilde

```
[ ] y_yhat, err = uts.y_yhat_errCFFs(data, results, 2)
    uts.evaluate(y_yhat)
```

```
Mean percent error:  11.625297853458502
RMSE:  1.0279815810145139
RMSE w yhat=mean:  1.4030345621243816
R-squared:  0.46317352332621164
```




```
[ ] uts.plotError(y_yhat, err, "ReHtilde")
```



Sharing weights between equivalent sample #'s within different kinematic sets was not improving the local fit

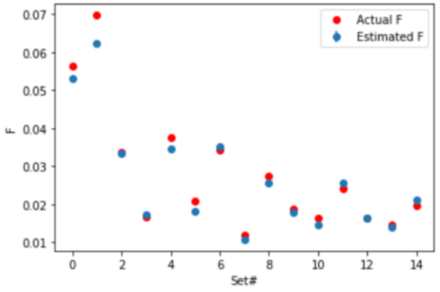
Using 10 replicas for training

Propagated Fs at 180

```
y_yhat, err = uts.y_yhat_errFs(results, data)
uts.evaluate(y_yhat)
```

```
Mean percent error:  6.972740356527691
RMSE:  0.002576464788301455
RMSE w yhat=mean:  0.015855649879790133
R-squared:  0.9735953552758353
```


```
[29] uts.plotError(y_yhat, err, "F")
```

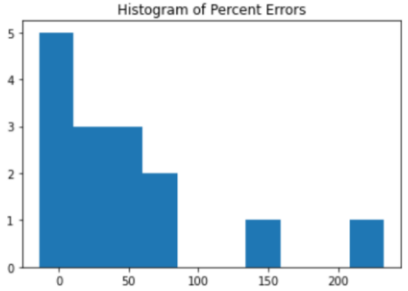


ReE

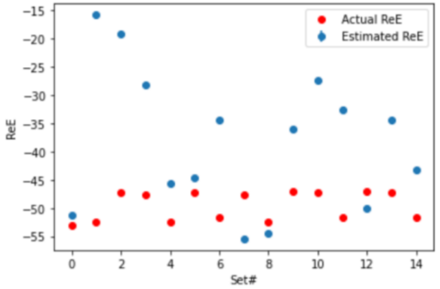
```
[23] y_yhat, err = uts.y_yhat_errCFFs(data, results, 1)
```

```
uts.evaluate(y_yhat)
```

```
Mean percent error:  51.03621968166363
RMSE:  16.47360348889805
RMSE w yhat=mean:  2.525480507907797
R-squared:  -41.54898213388517
```




```
[25] uts.plotError(y_yhat, err, "ReE")
```

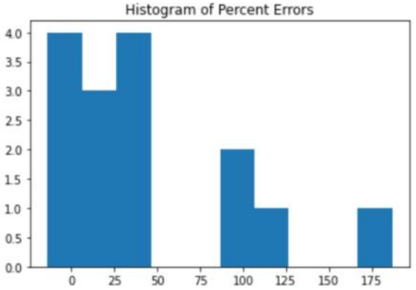


ReH

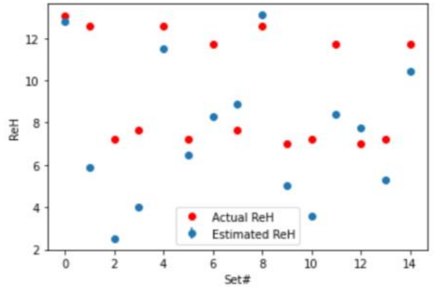
```
[20] y_yhat, err = uts.y_yhat_errCFFs(data, results, 0)
```

```
uts.evaluate(y_yhat)
```

```
Mean percent error:  47.55818410442468
RMSE:  2.94254374490872 73
RMSE w yhat=mean:  2.5254628436780107
R-squared:  -0.3575751800292435
```




```
[22] uts.plotError(y_yhat, err, "ReH")
```

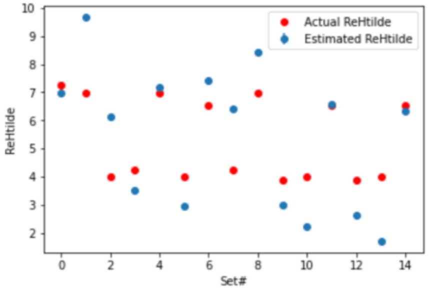


ReHtilde

```
y_yhat, err = uts.y_yhat_errCFFs(data, results, 2)
uts.evaluate(y_yhat)
```

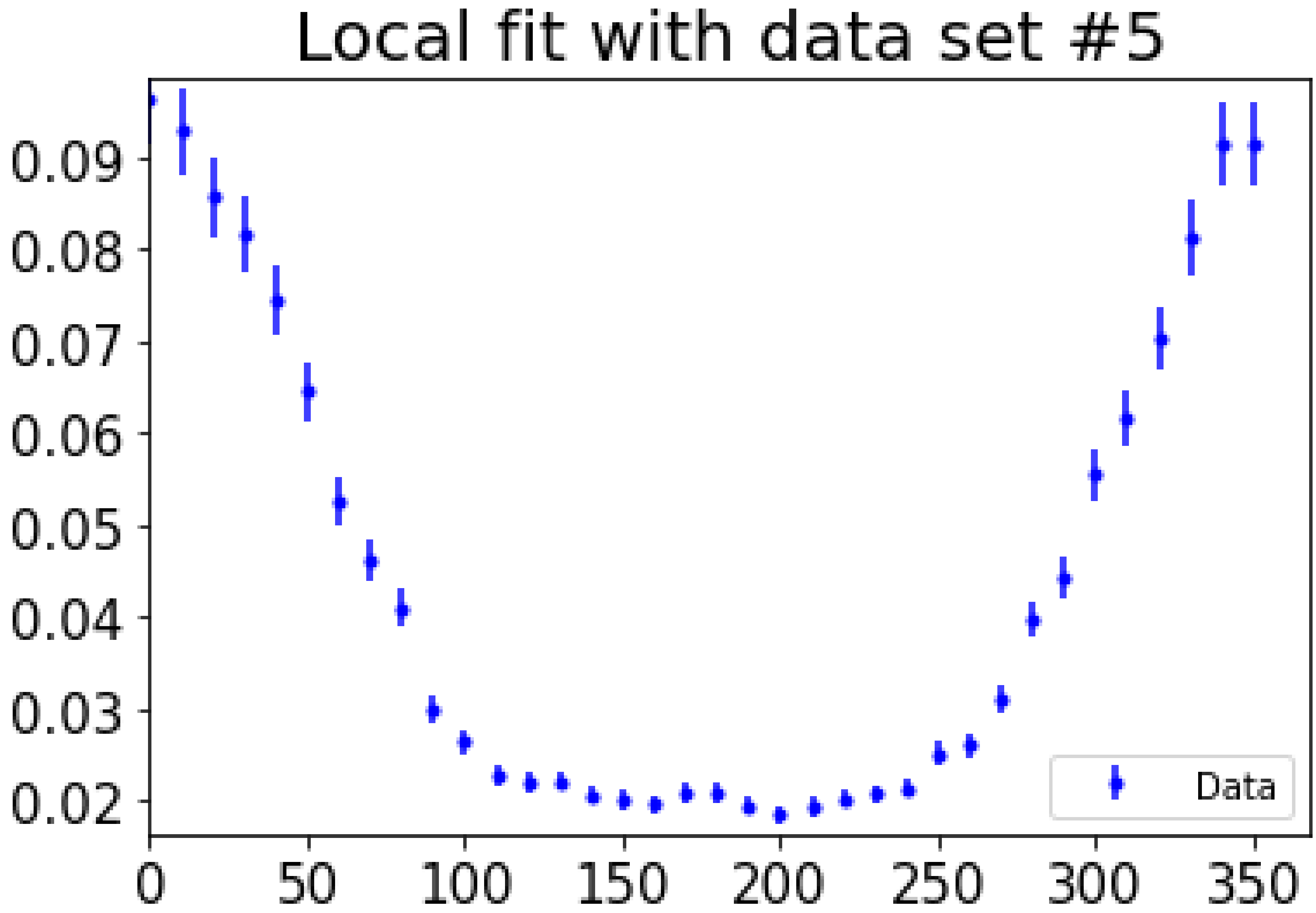
```
Mean percent error:  32.415910324464534
RMSE:  1.457502071397903
RMSE w yhat=mean:  1.4030345621243816
R-squared:  -0.07914952100603645
```

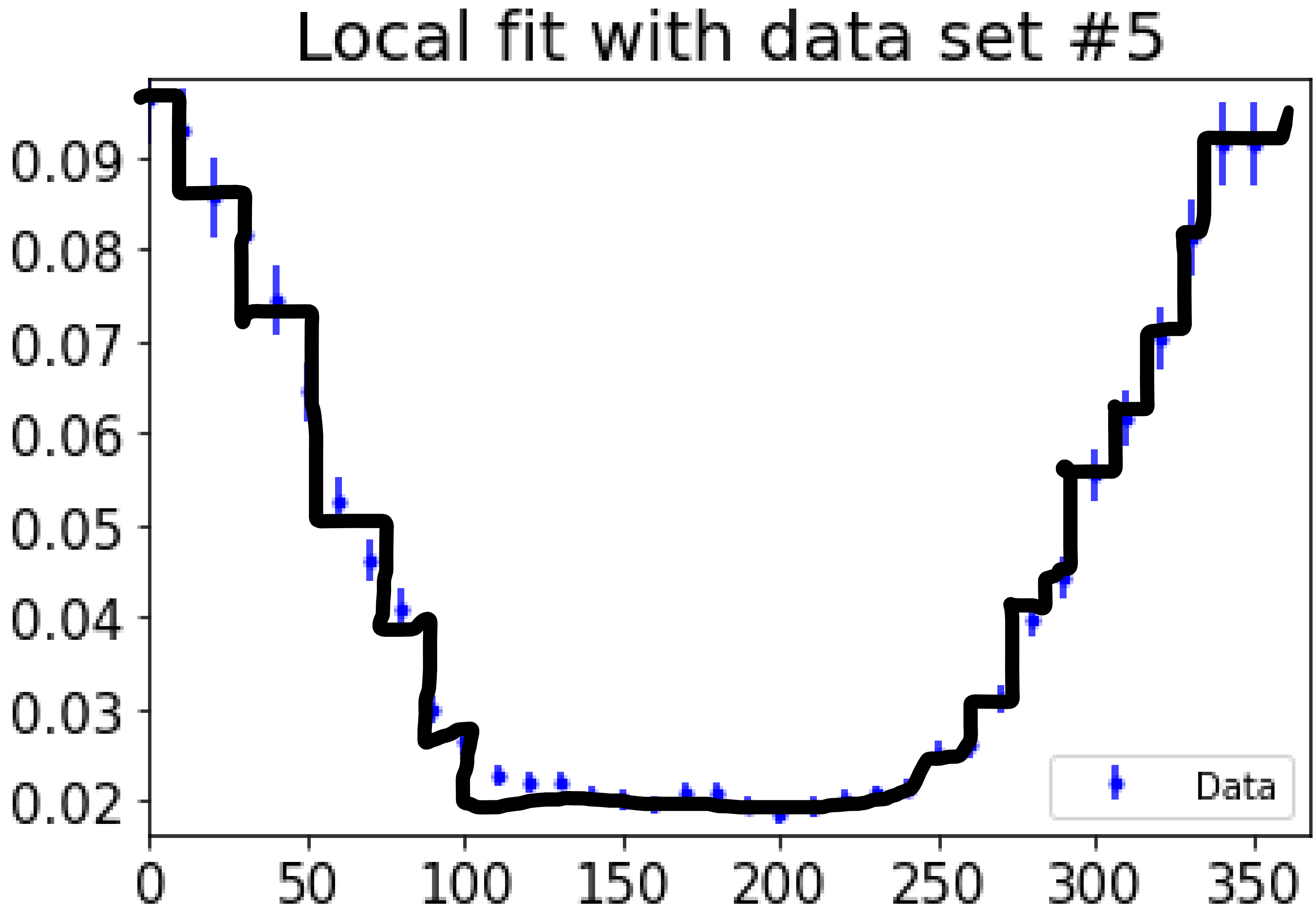

```
[27] uts.plotError(y_yhat, err, "ReHtilde")
```

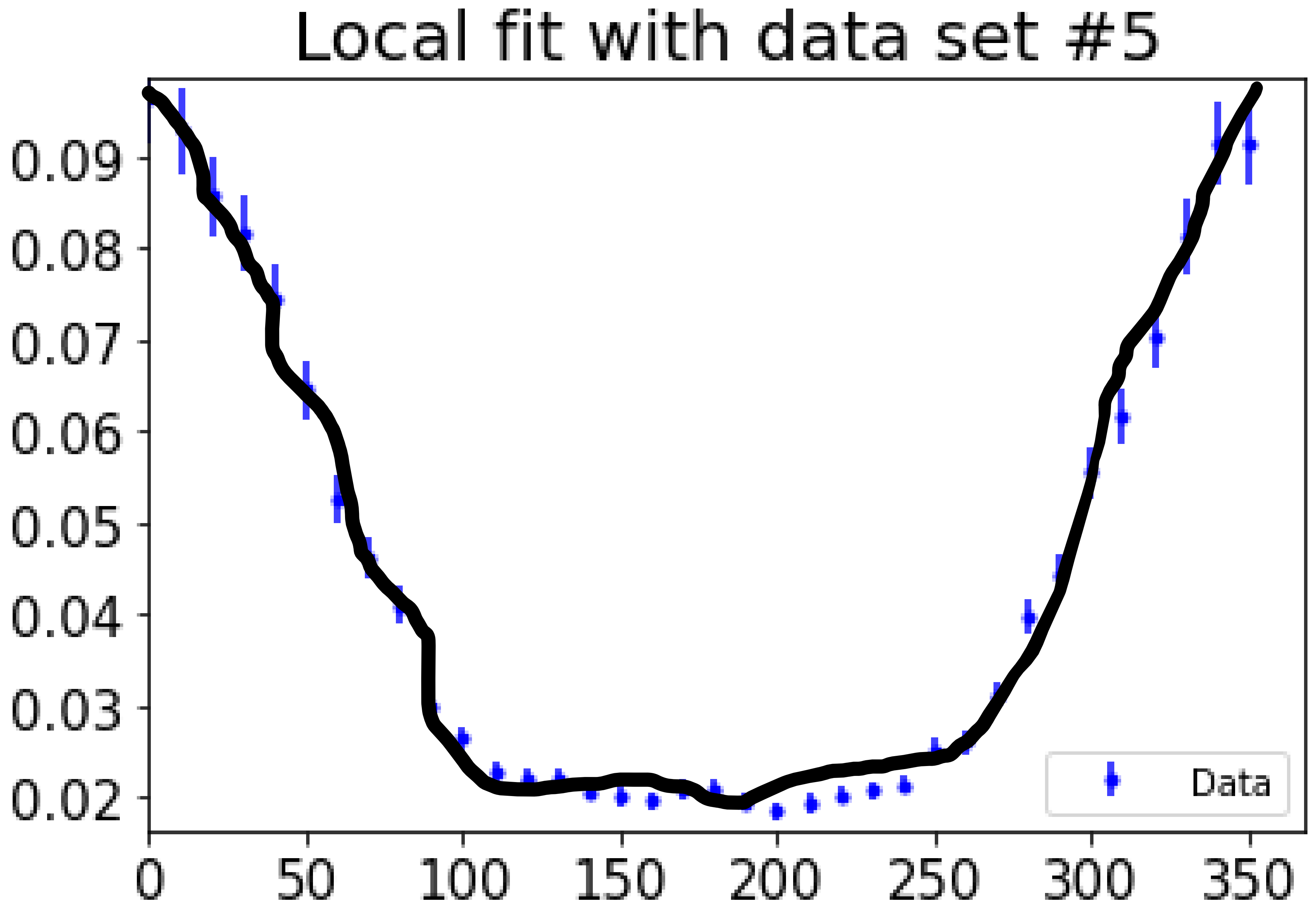


Replicas of input data (target F’s varied by respective errF’s) is helping in prediction of CFF’s, but not significant in fitting to F value itself

- Most likely overfitting is being reduced
- When viewing a new sample of F, the network will try to update the CFFs to look for CFF values that better fit the new F, but due to the nature of the optimization algorithm will find new CFF values close to the previous prediction.
- In other words, linear approximation of F is smoothed out rather than overfitting heavily to the existing data points.


Local fit with data set #5
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0
50
100
150
200
250
300
350
Data


Local fit with data set #5
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0
50
100
150
200
250
300
350
Data


Local fit with data set #5
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0
50
100
150
200
250
300
350
Data

Testing this hypothesis

- Does decreasing batch size improve CFF bound results under the same settings?
- Does adding dropout improve CFF bounds?
- What would validation accuracy look like if we split off a portion of data for testing from each sample of Phi's?

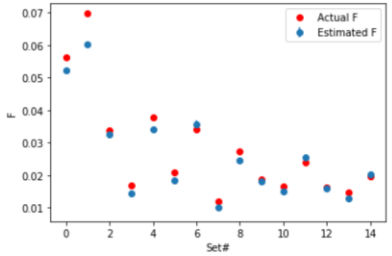
Decreasing batch size from default 32 -> 10

Propagated Fs at 180

```
y_yhat, err = uts.y_yhat_errFs(results, data)
uts.evaluate(y_yhat)
```

```
Mean percent error:  9.230884239704144
RMSE:  0.003204941508471753
RMSE w yhat=mean:  0.015855649879790133
R-squared:  0.9591424687833635
```


```
[205] uts.plotError(y_yhat, err, "F")
```

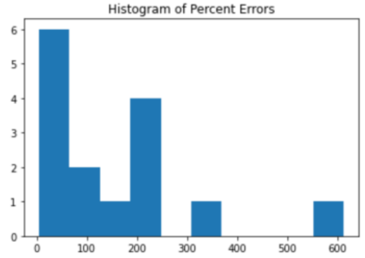


ReE

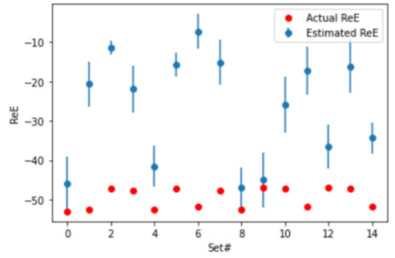
```
[199] y_yhat, err = uts.y_yhat_errCFFs(data, results, 1)
```

```
uts.evaluate(y_yhat)
```

```
Mean percent error:  148.29241353490877
RMSE:  26.09130167115023
RMSE w yhat=mean:  2.525480507907797
R-squared:  -105.73416345425076
```




```
[201] uts.plotError(y_yhat, err, "ReE")
```

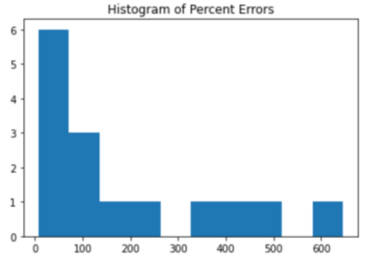


ReH

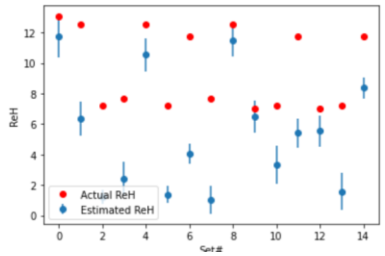
```
[196] y_yhat, err = uts.y_yhat_errCFFs(data, results, 0)
```

```
uts.evaluate(y_yhat)
```

```
Mean percent error:  185.28534959289684
RMSE:  4.81314354894882
RMSE w yhat=mean:  2.5254628436780107
R-squared:  -2.6322494136805963
```



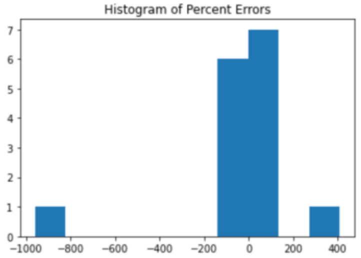

```
[198] uts.plotError(y_yhat, err, "ReH")
```



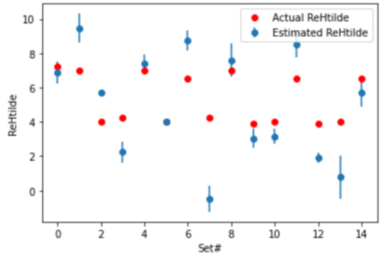
ReHtilde

```
y_yhat, err = uts.y_yhat_errCFFs(data, results, 2)
uts.evaluate(y_yhat)
```

```
Mean percent error:  117.03398740209146
RMSE:  2.0298322809025007
RMSE w yhat=mean:  1.4030345621243816
R-squared:  -1.0930691220600806
```



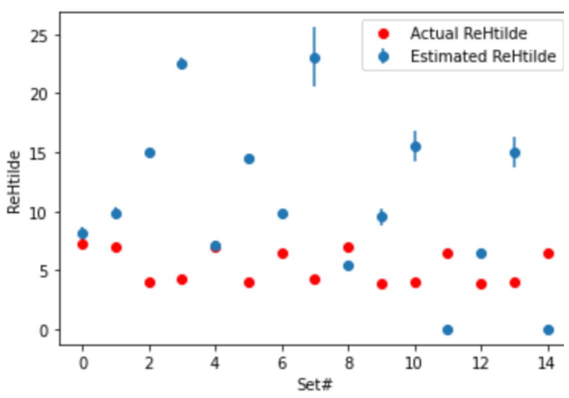

```
[203] uts.plotError(y_yhat, err, "ReHtilde")
```



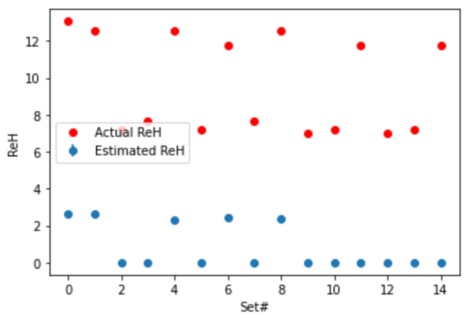
Decreasing batch size from default 32 -> 10 & w/ ReLU

Propagated Fs at 180

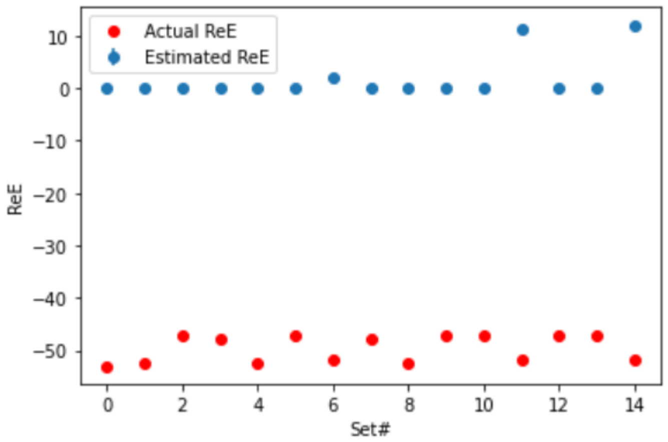
```
uts.plotError(y_hat, err, "ReHtilde")
```




```
uts.plotError(y_yhat, err, "ReH")
```

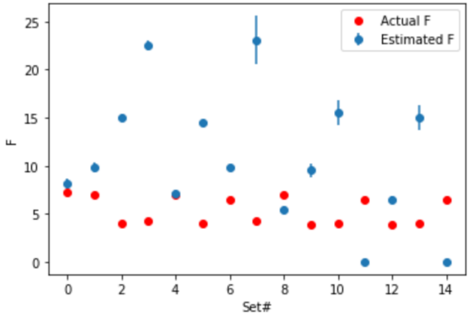



```
uts.plotError(y_yhat, err, "ReE")
```




```
[ ] y_yhat, err = uts.y_yhat_errFs(results, data)
    uts.evaluate(y_yhat)
```

```
uts.plotError(y_yhat, err, "F")
```



After reducing the batch size, F fitting was slightly worse, CFF fitting was relatively much worse.

This supports the hypothesis that F is being overfit.
Though we might have expected to not see a decrease in F’s fit, it was relatively small and I will have to try several different batch sizes to fully understand the effect.