PhotoNet

#Physics

Goal:

Given a spectrum, design a geometry to provide it.

Empirically test this as well.

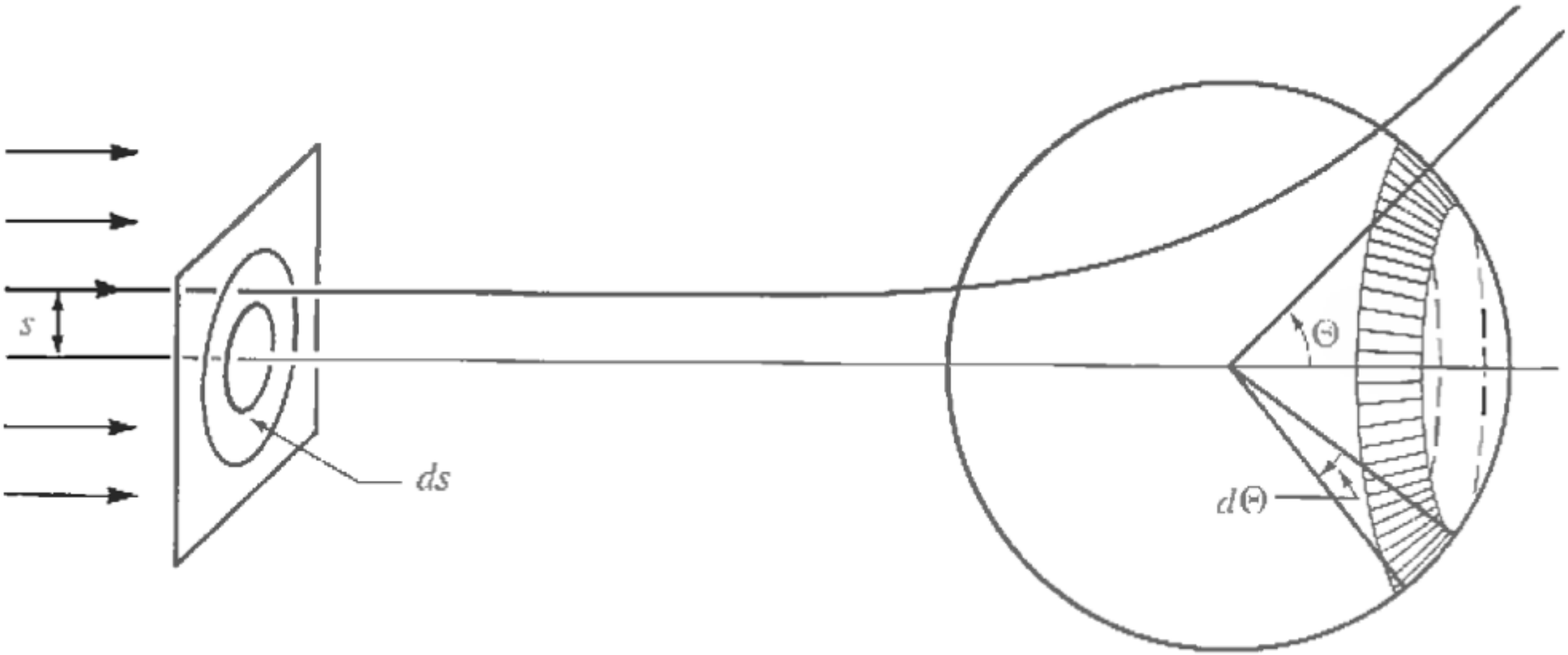
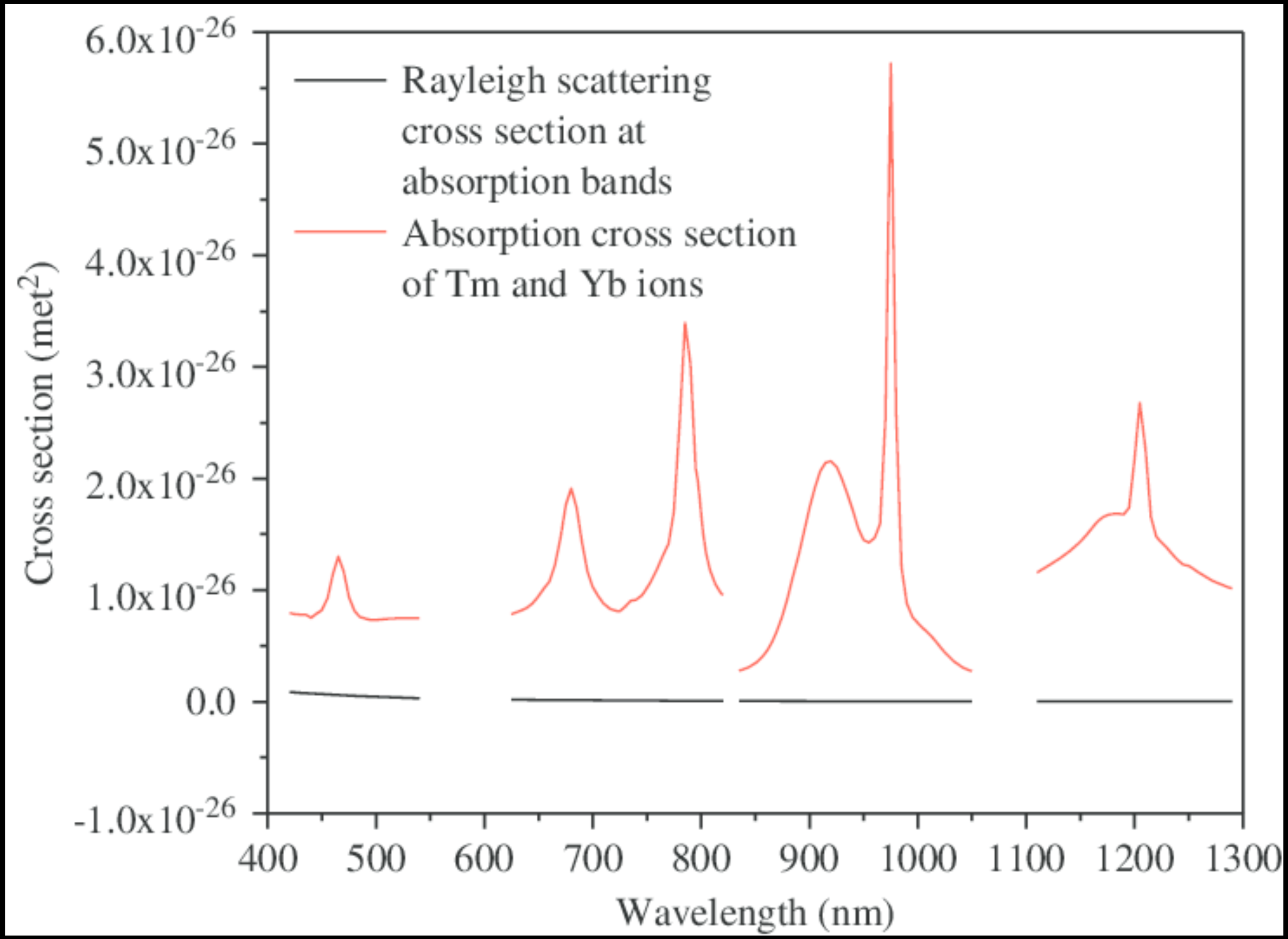


FIGURE 3.19 Scattering of an incident beam of particles by a center of force.



How do we design an object to match this spectrum?

We have a script in matlab that can generate the scattering/ absorption spectrum for a given geometric arrangment. The idea is to use this script to generate training data, then train a neural network with this data.

Thus the neural network will be able to produce the spectrum given a geometric arrangement, and also be able to further predict spectrums for objects it has not seen.

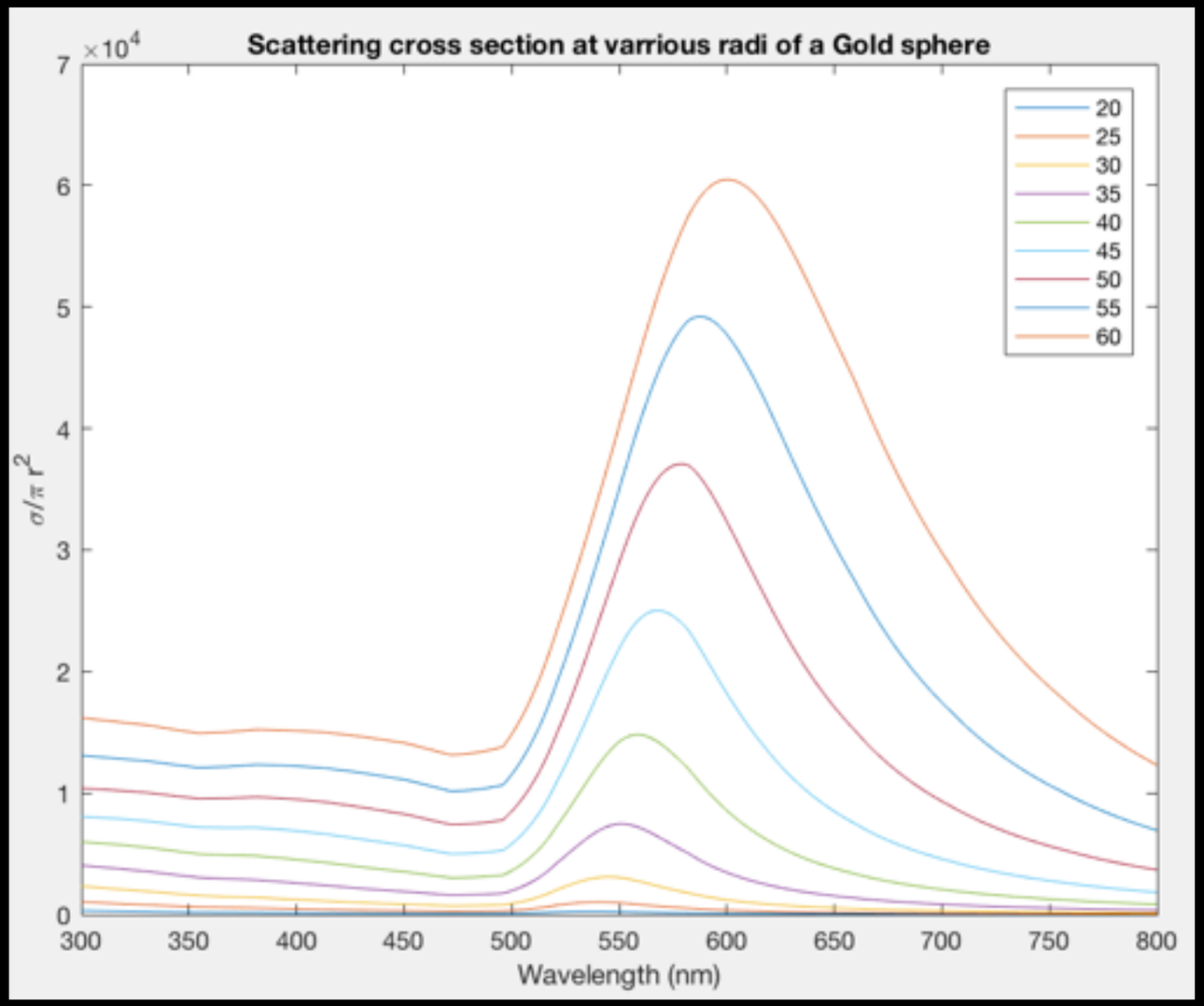
The major contribution of this work is that it allows us to find a way to match a spectrum. We can take a desired spectrum, feed it into the output of the neural network, then find out the geometric arrangement suggested to create such a spectrum.

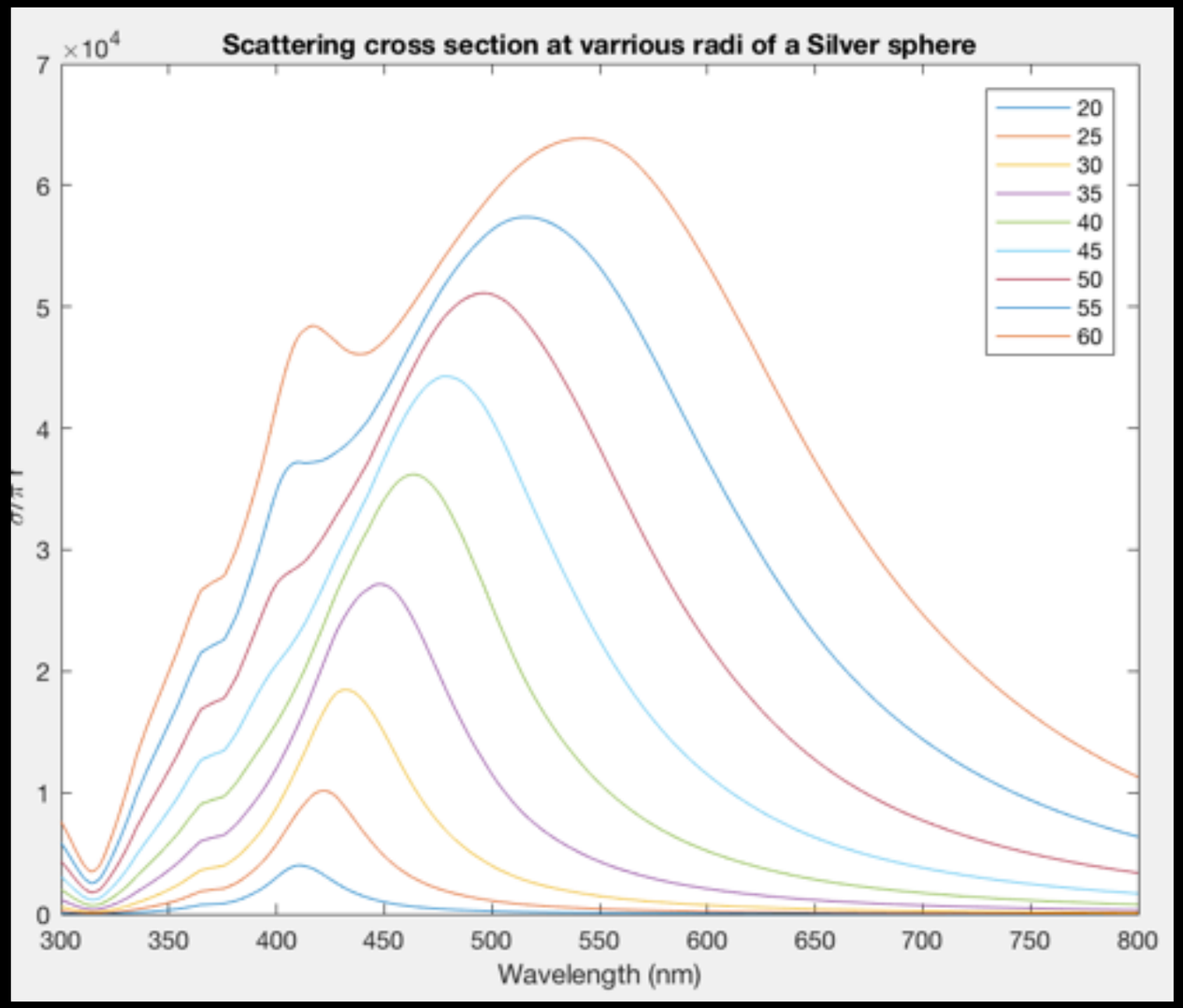
Experiment 1: Single Sphere of different radi

The first data was simply a differing radius gold sphere.

I sampled a sphere at each radius between 10nm and 40nm in .1nm increments.

Training Data: Spectrums from Matlab





Now we generate the data and train a neural network.

Step 1: Single Sphere with variable radius

- Architecture:
    - 1 in, 20x20x20x20x20x100 out
    - 3620 parameters
- Training Data:
    - 1800 training points, 100 spectrum points per
        - Spectrum is 300-800nm, sampled every 5.
        - Training is 10-> 100nm, .05 increments
    - ~180,000 data points
- Training results:
    - ~21 loss per epoch
    - .01 error per point of the spectrum.
- Reversing - feeding the spectrum, getting the geometry:
    - Had to do it manually
    - 23.5275 -> The program finds: 23.5396

Quite accurate!! (less than .04% error)

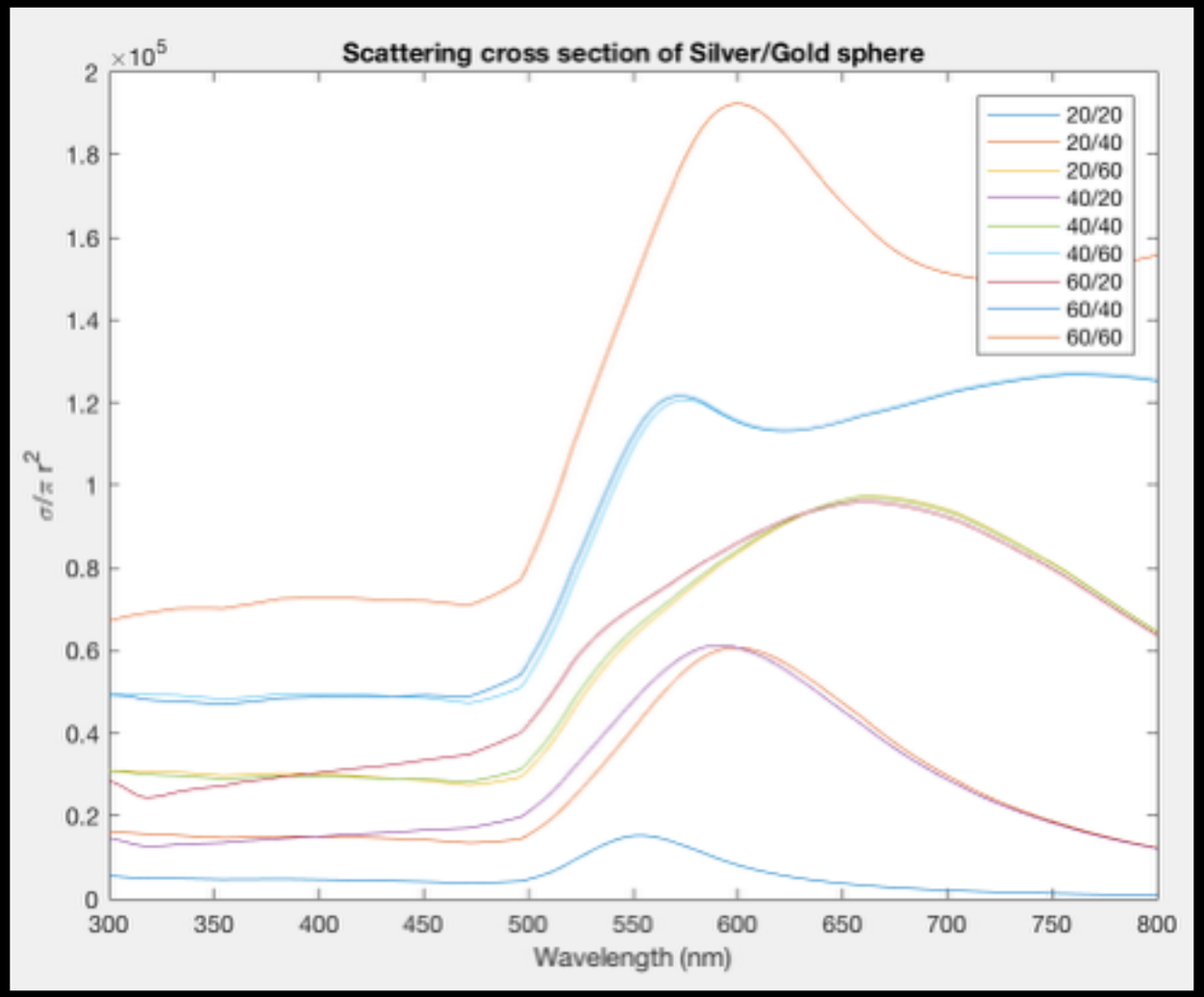
Experiment 2: Two Sphere of different radi and material

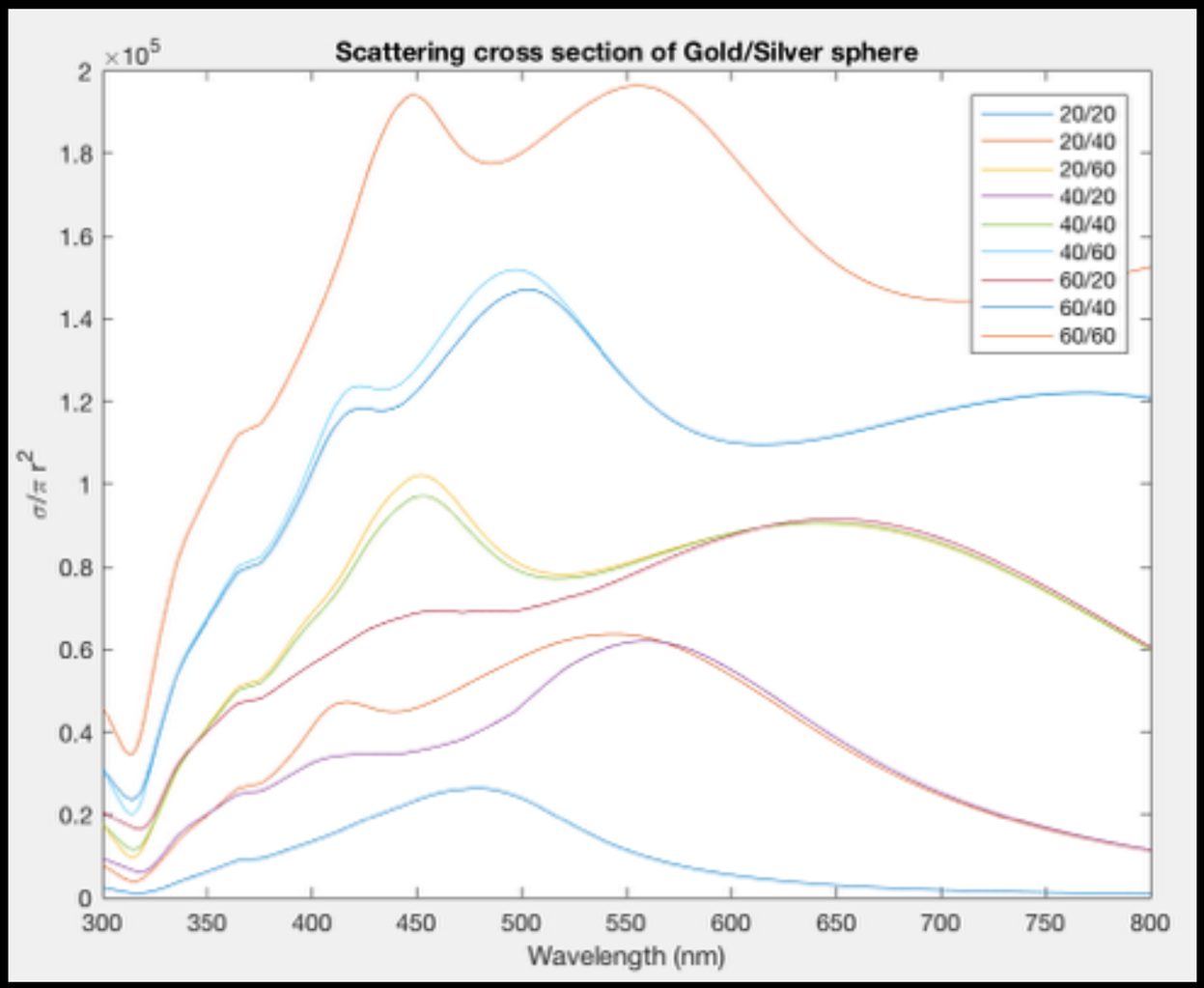
The next experiment was with two spheres of different materials (gold on the inside, silver on the outside) and different radi.

Step 2: Two-Layer Sphere with different materials

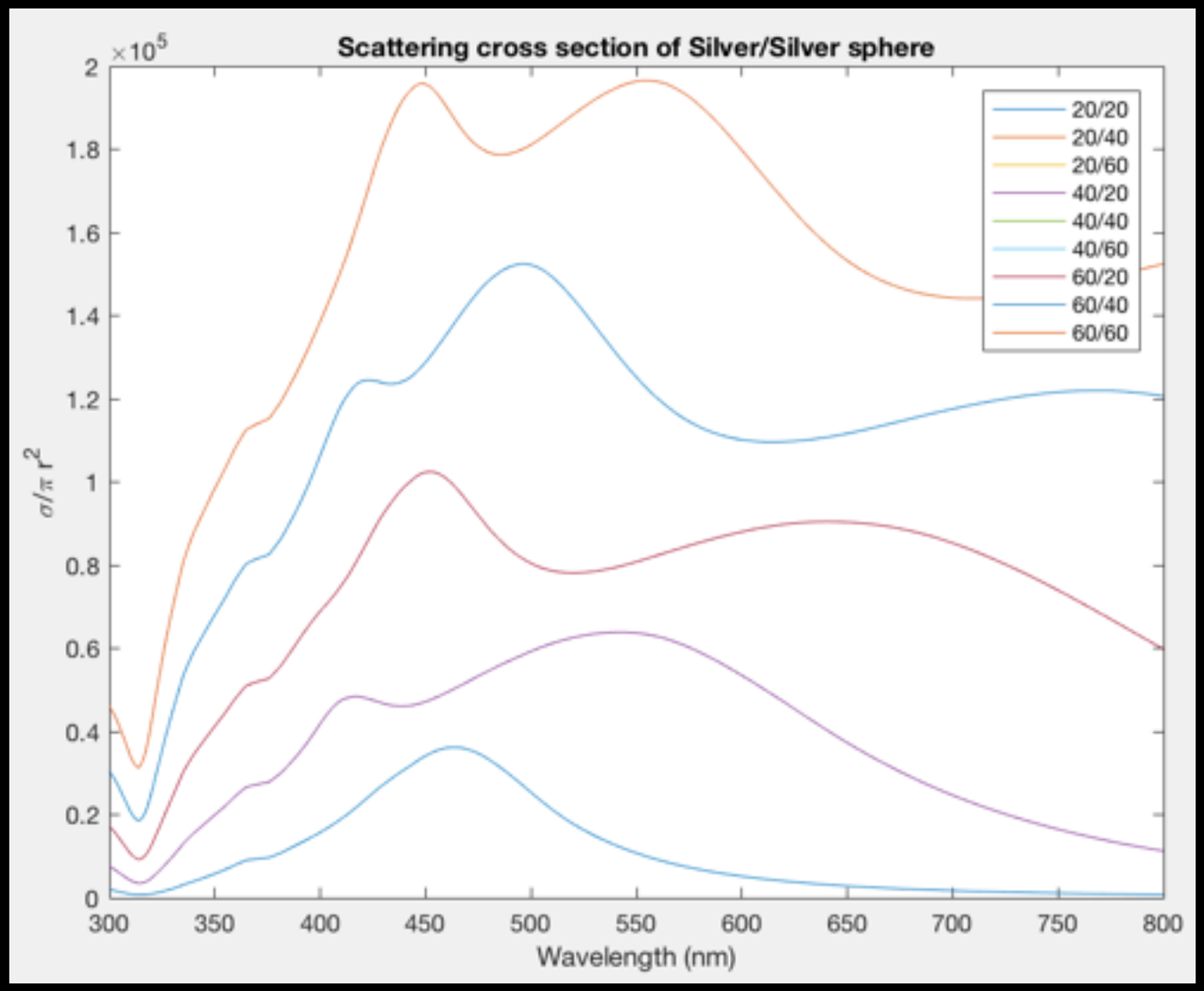
- Architecture:
  - 1 in, 20x20x20x20x20x100 out
  - 3620 parameters
- Training Data:
  - 3844 training points, 100 spectrum points per
    - Spectrum is 300-800nm, sampled every 5.
    - Training is 20:2:80, 20:2:80, gold/silver.
  - ~384,400

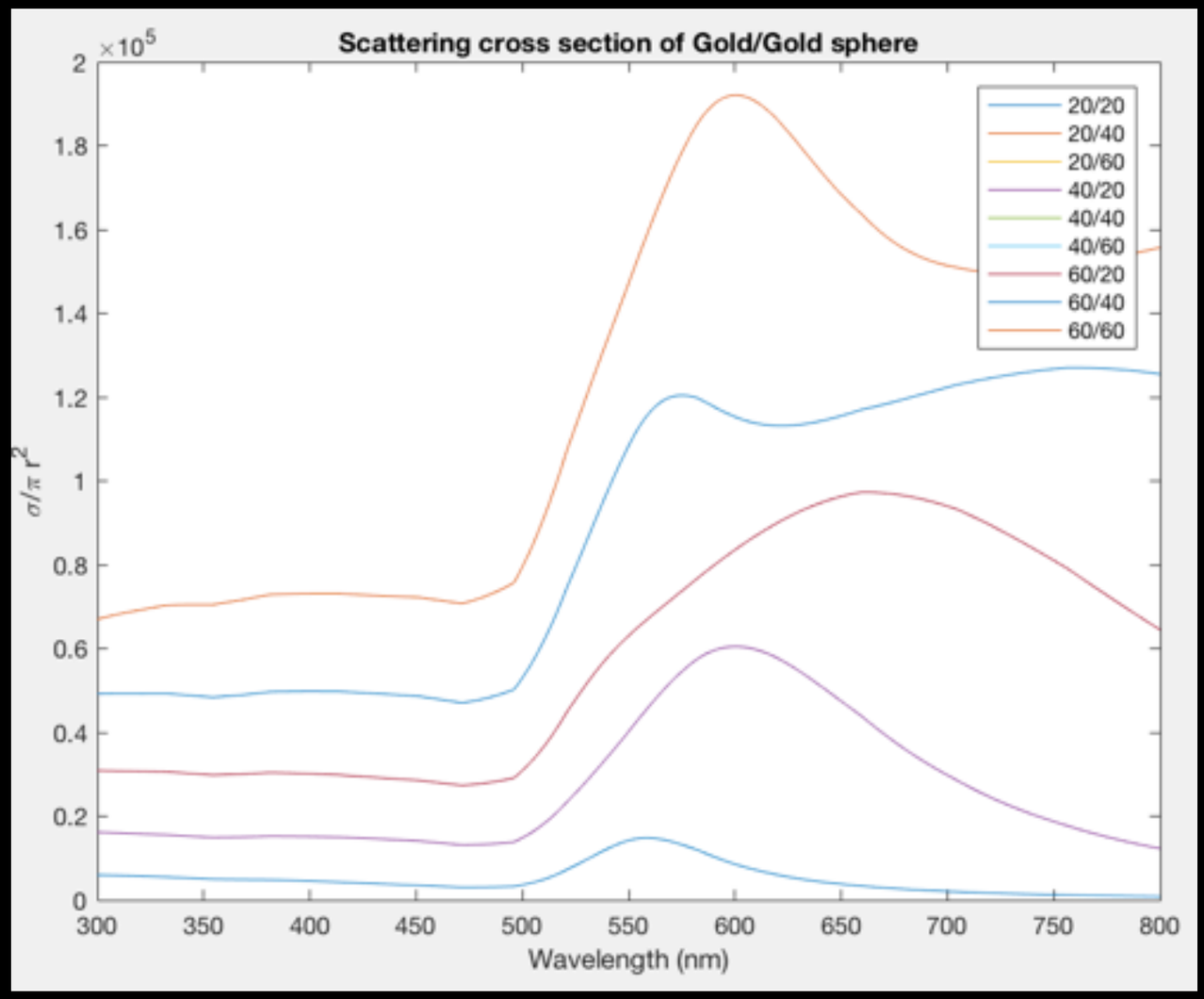
Training Data: Spectrums from Matlab





Training Data: Spectrums from Matlab

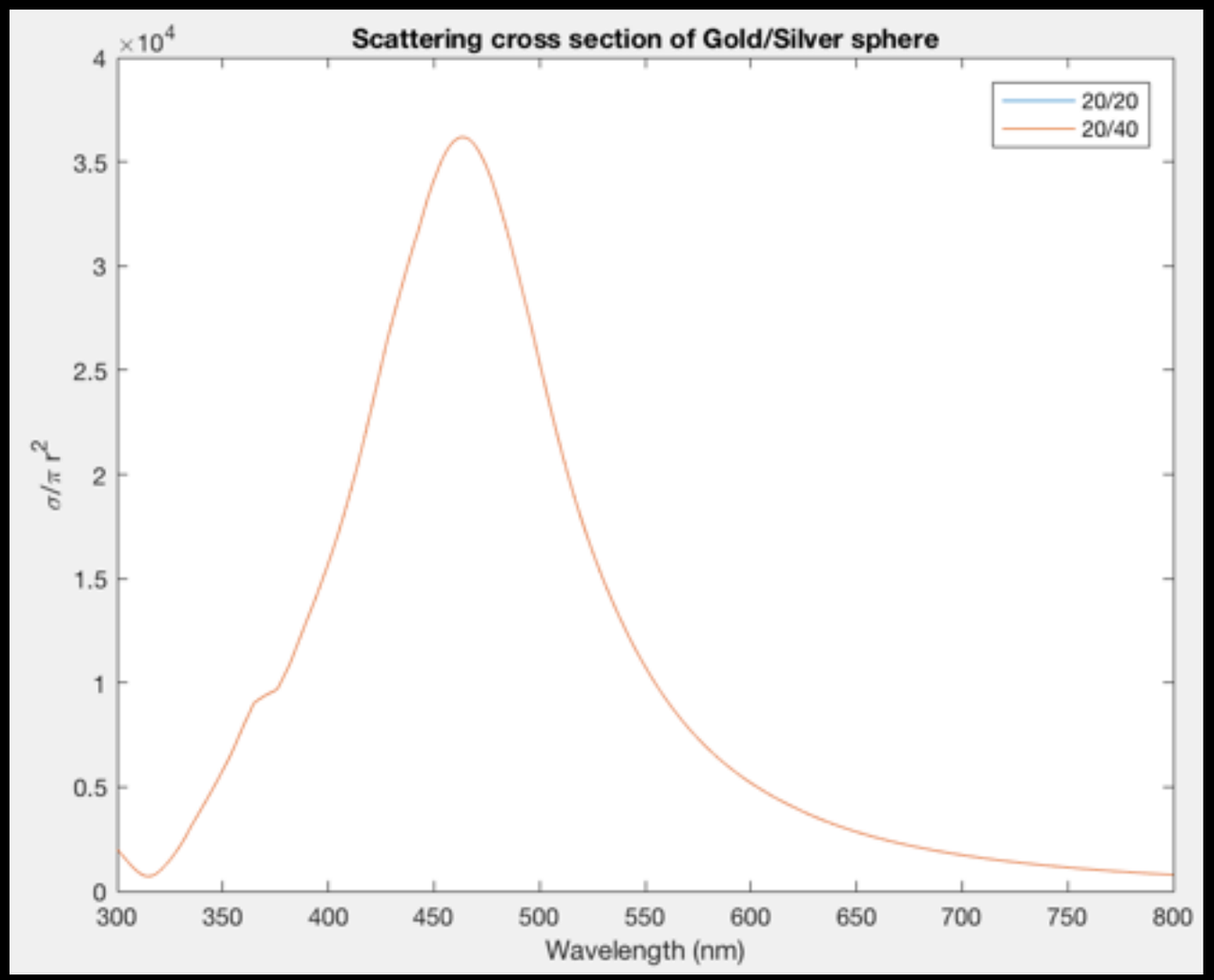


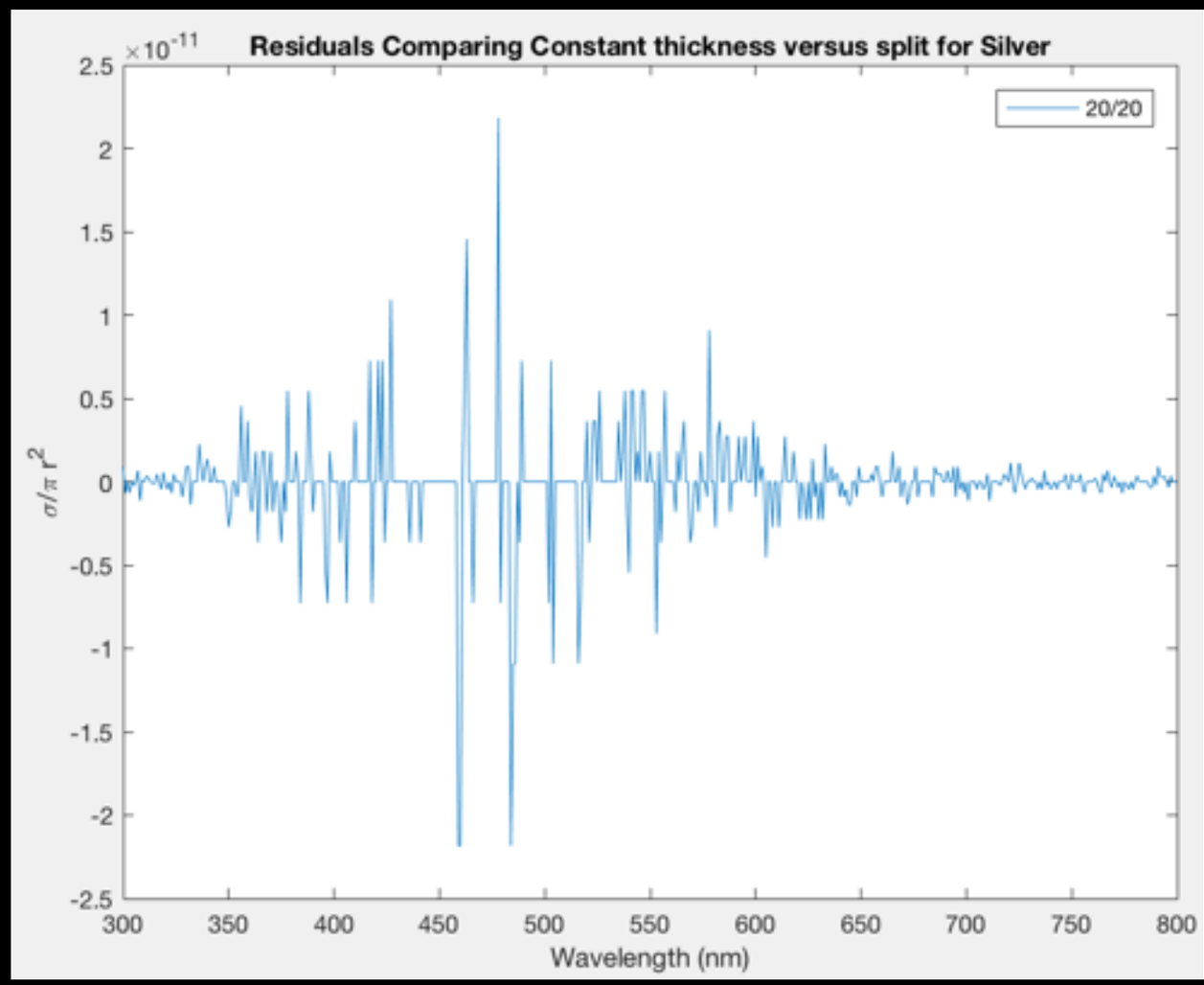


Note that these should match the cases for the pure silver/gold sphere.

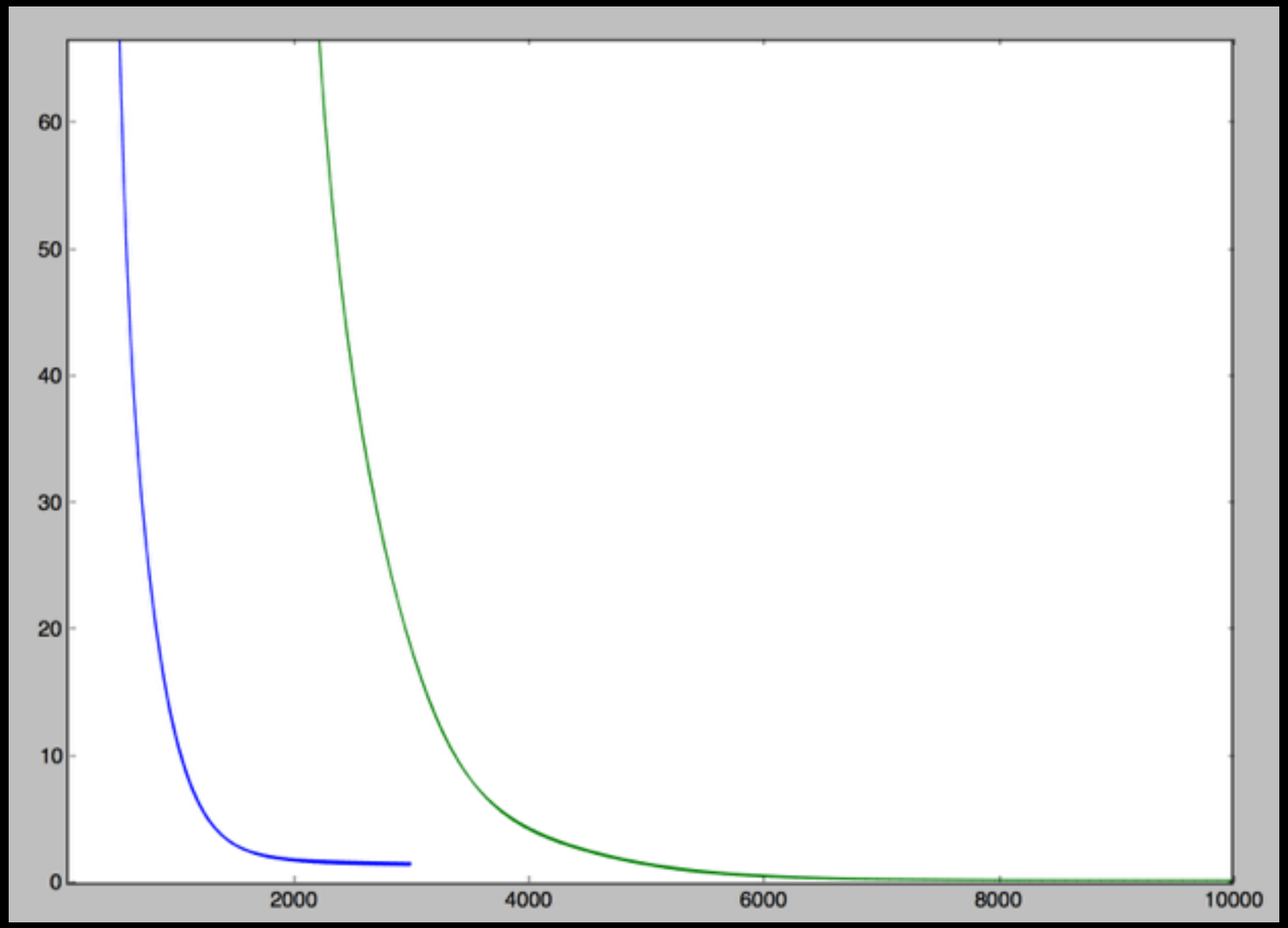
The differences are likely from the size differences.

Testing Training Data

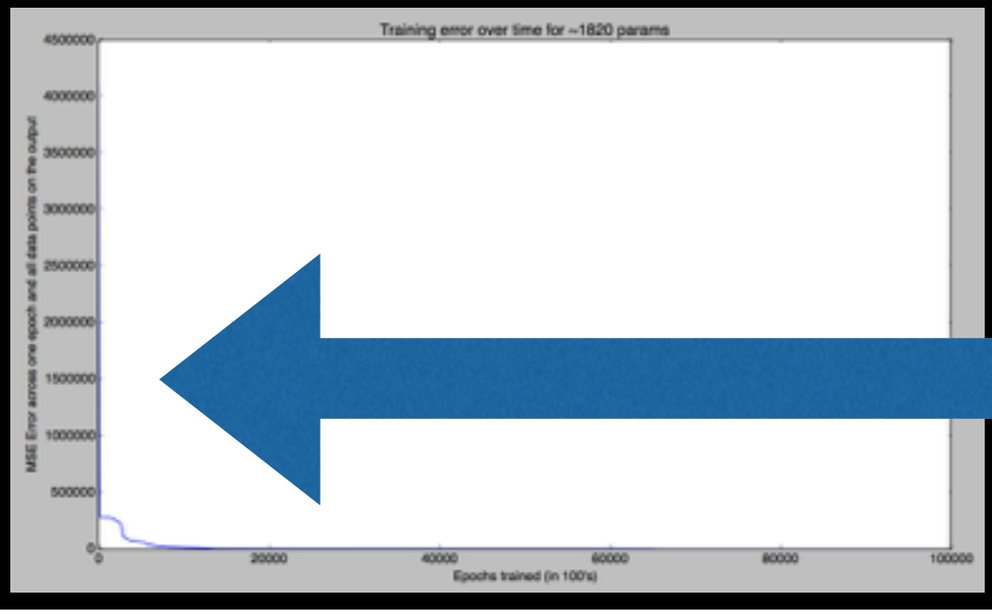




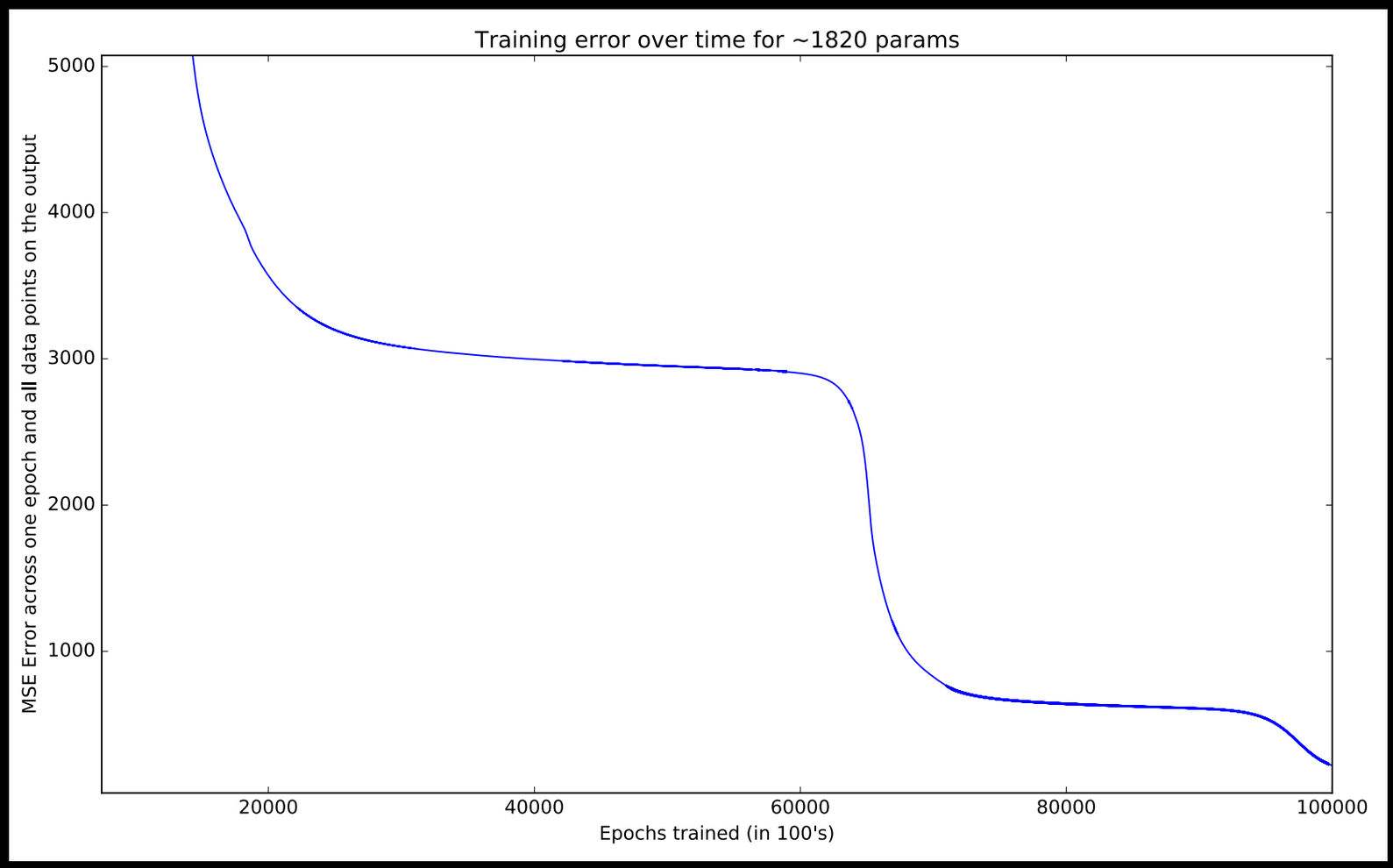
The residuals are very small (10^-11) compared to the spectrum (10^4).
The algorithm works well in matlab!



Training loss over iterations.
By around 6,00 iterations, it is approximately its final value

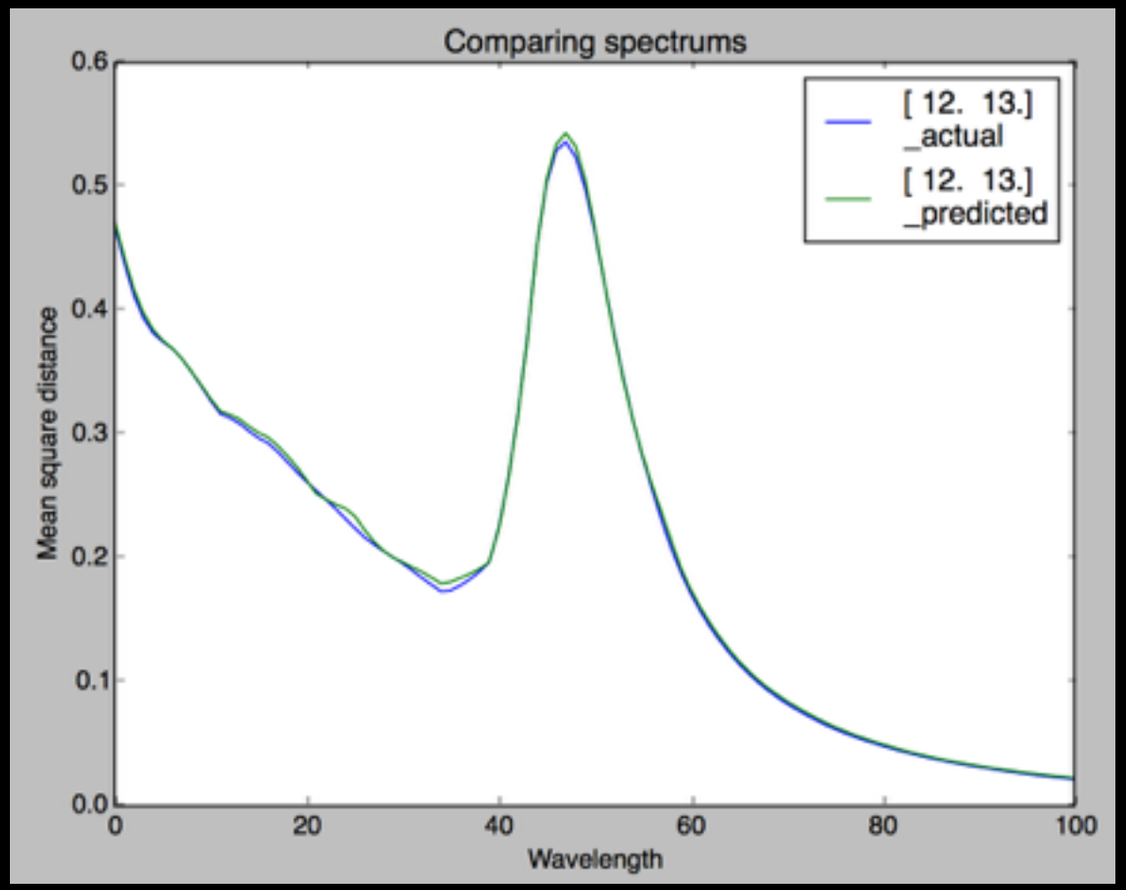


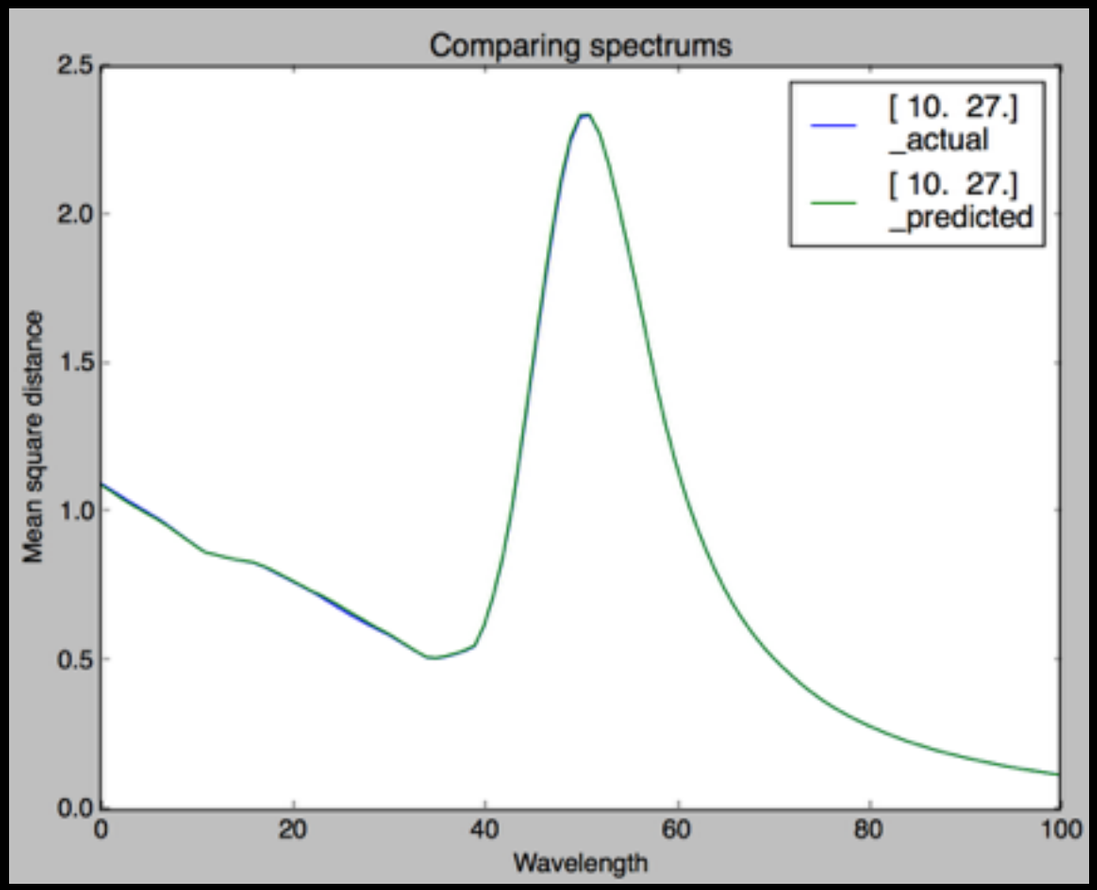
It trains that well



Training loss over iterations.
Still at 100,000 it has not converged.

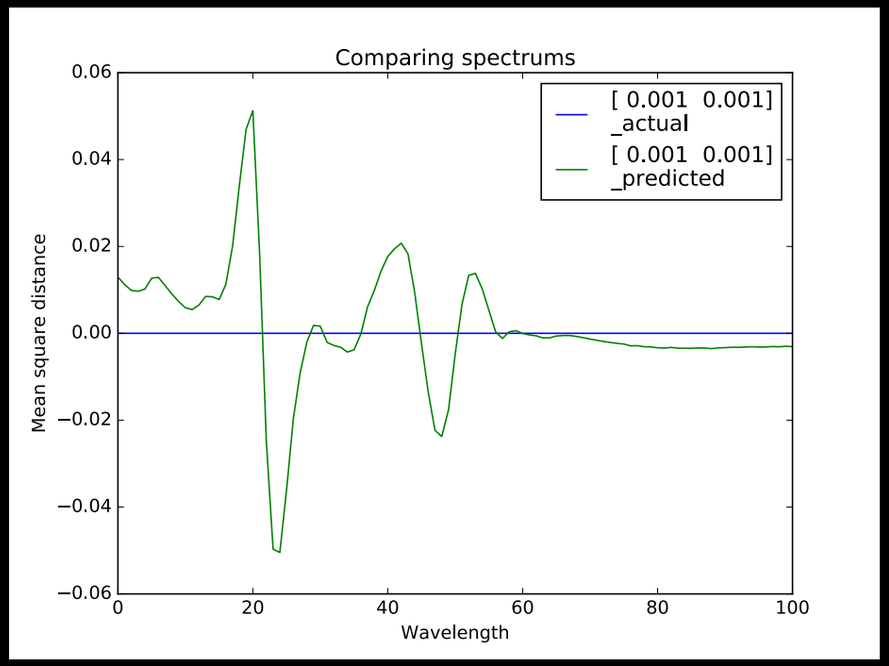
Experiment 2: Spectrum Results



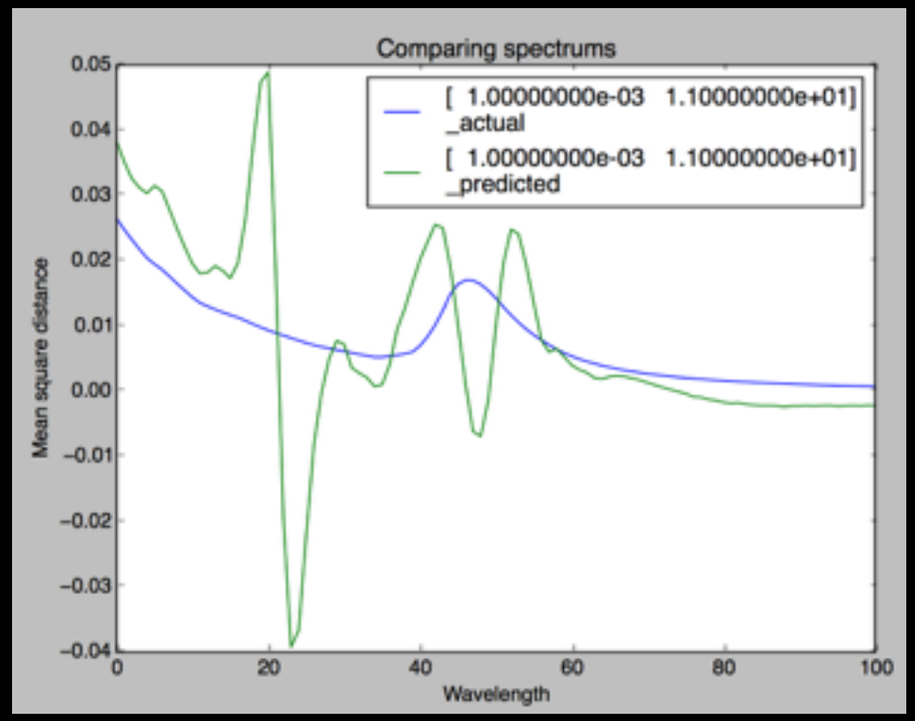


The first couple spectrums look bad (due to data problems)

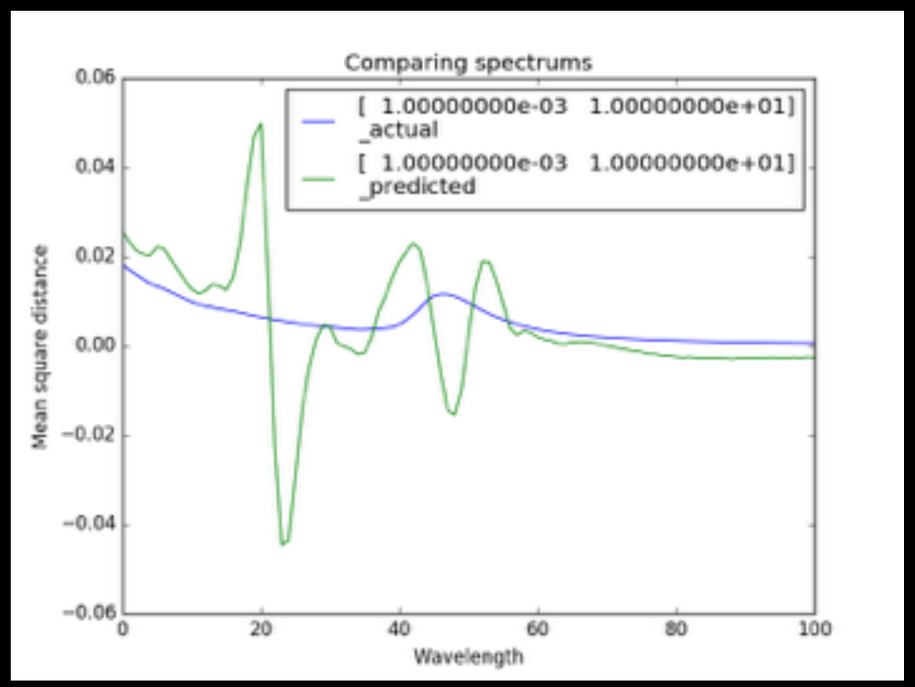
1st



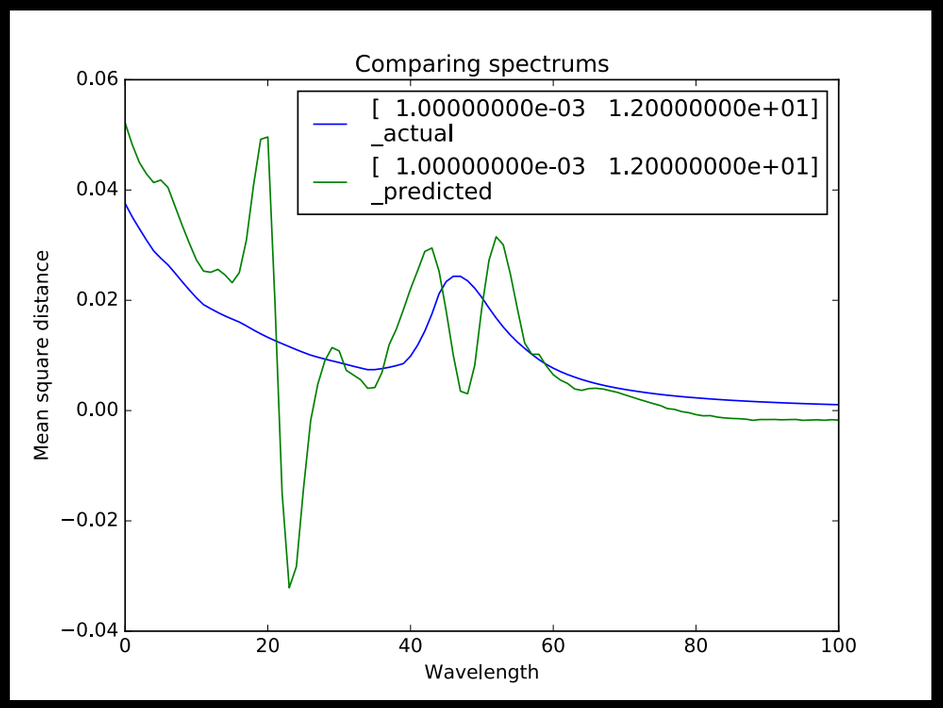
3



2

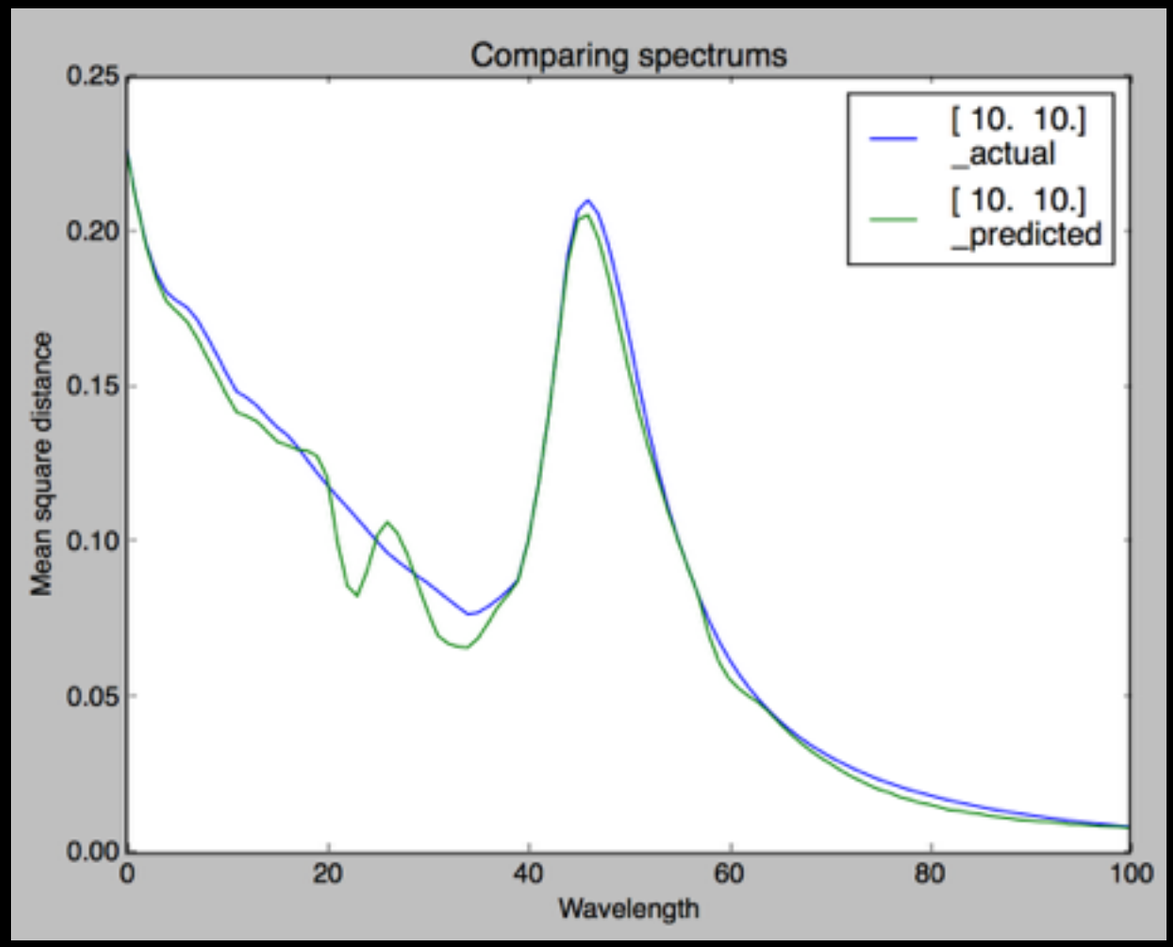


4

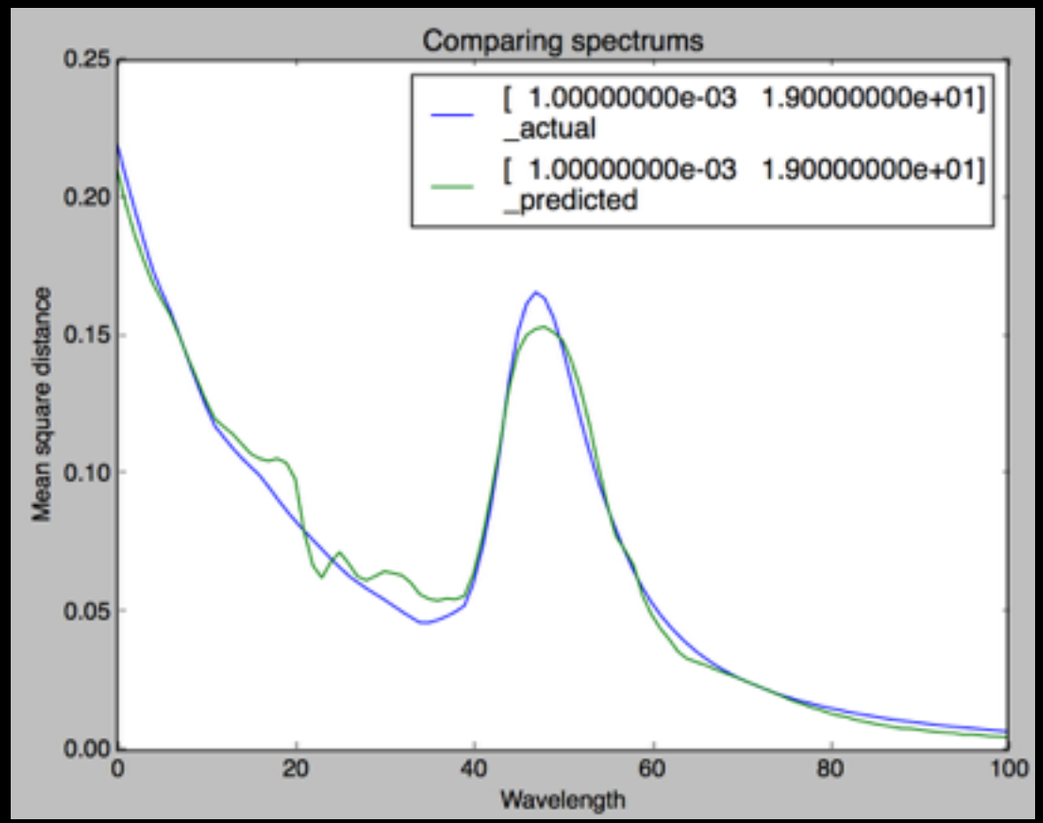


But once you sample from well inside the training data, the results are very clear

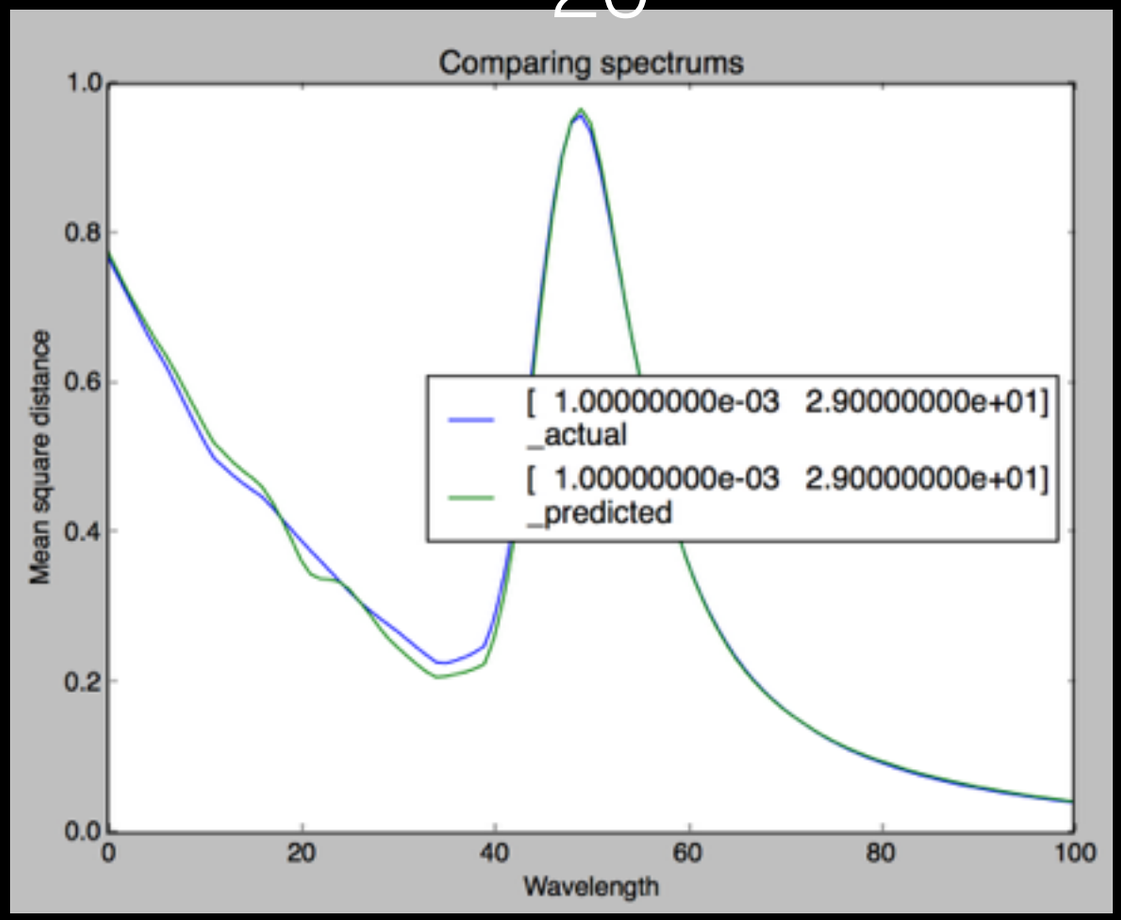
10 by 10



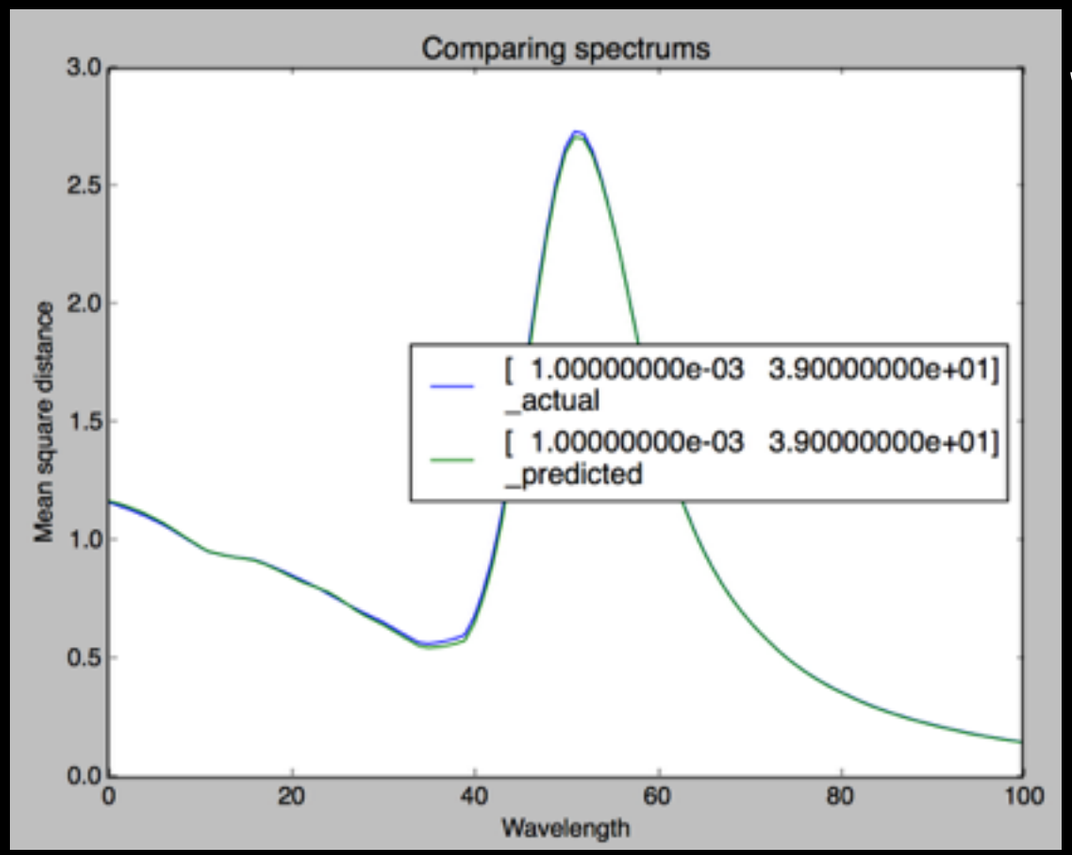
10


Comparing spectrums
[1.00000000e-03 1.90000000e+01]
_actual
[1.00000000e-03 1.90000000e+01]
_predicted
Mean square distance
Wavelength
0.00
0.05
0.10
0.15
0.20
0.25
0
20
40
60
80
100

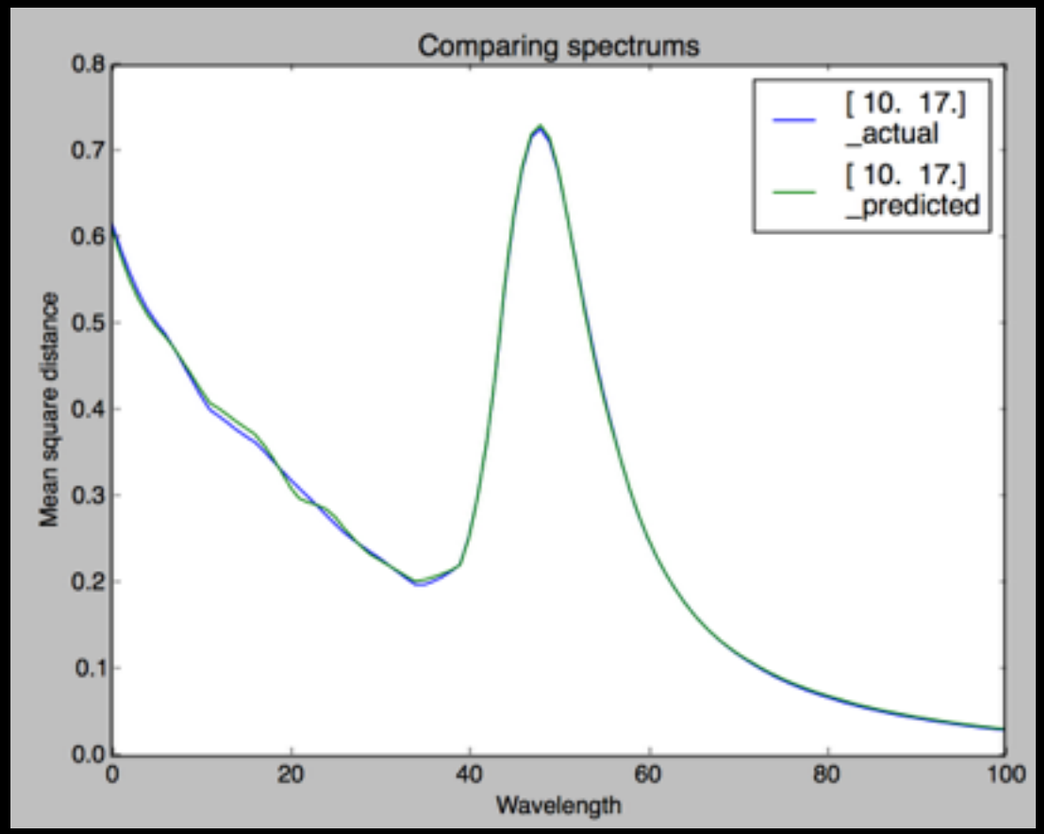
20

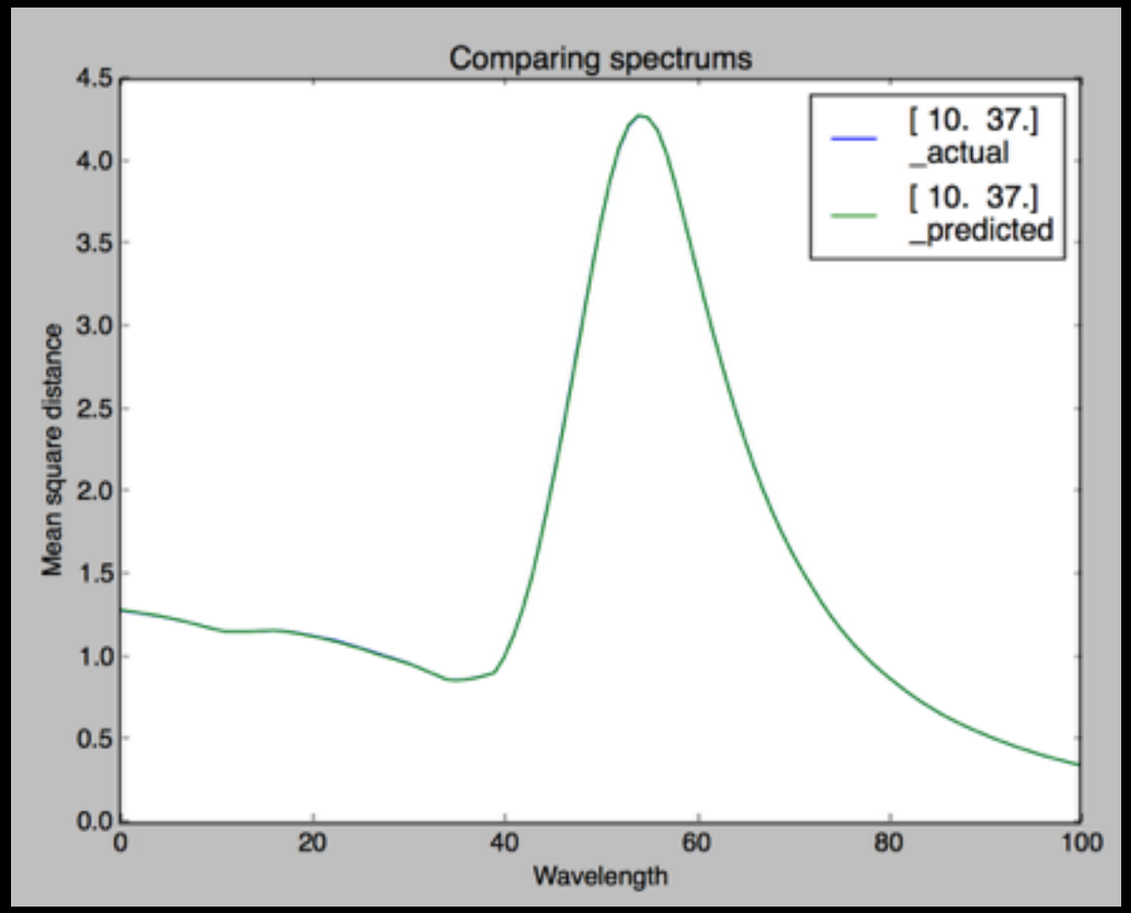

Comparing spectrums
[1.00000000e-03 2.90000000e+01]
_actual
[1.00000000e-03 2.90000000e+01]
_predicted
Mean square distance
Wavelength
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
100

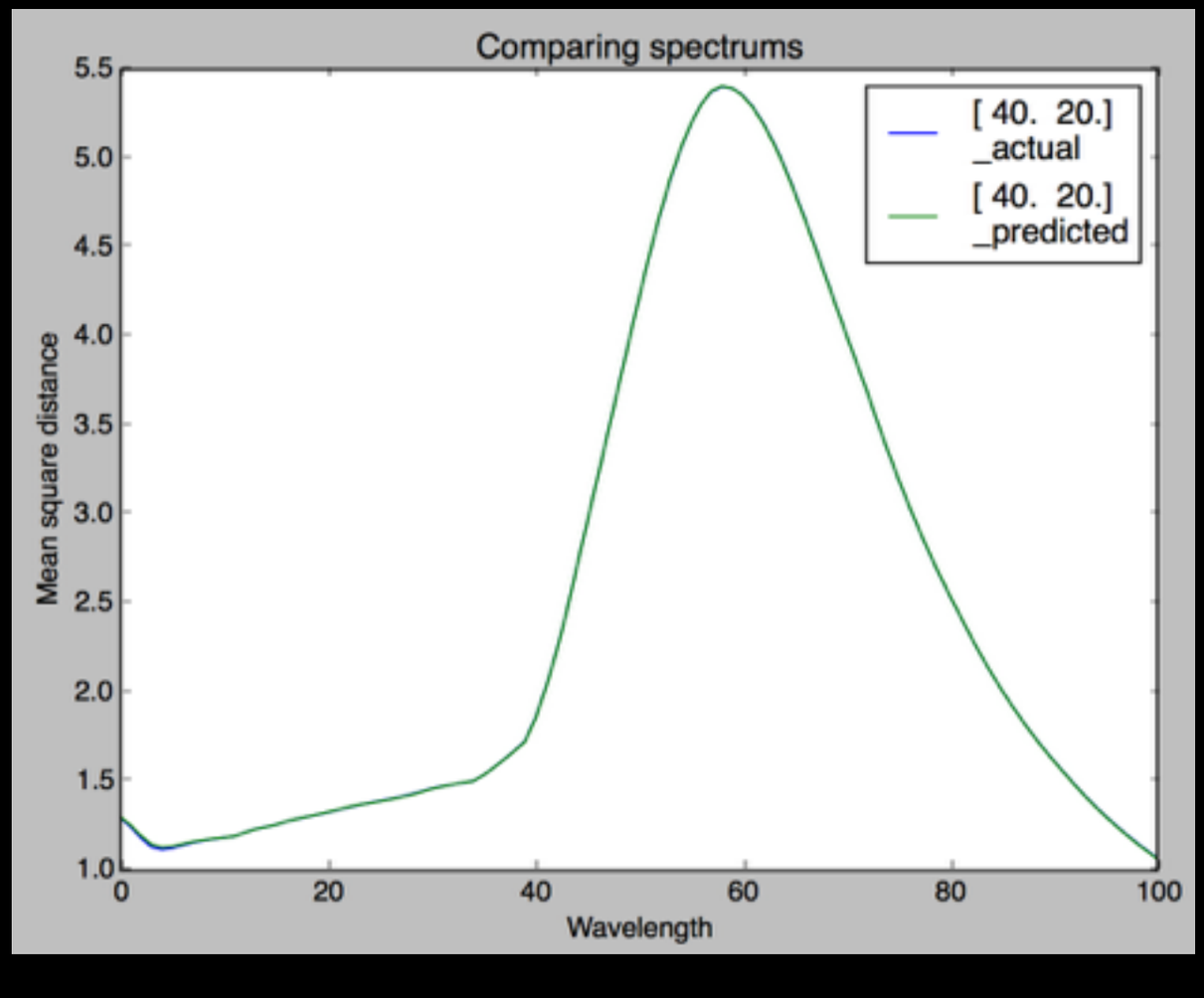
30

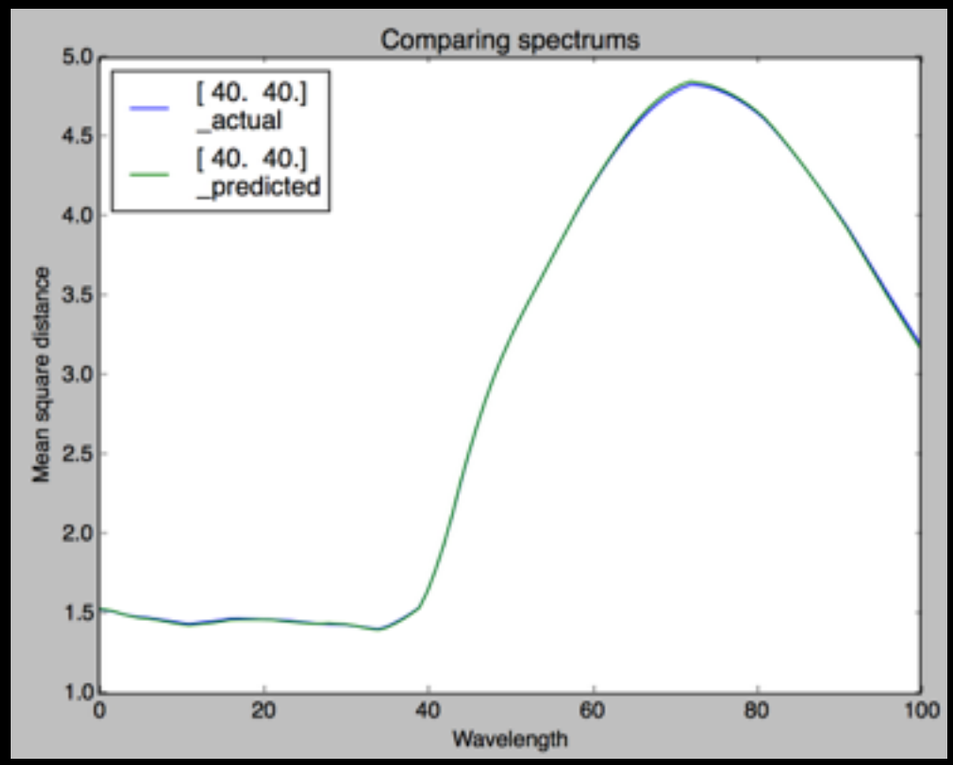

Comparing spectrums
[1.00000000e-03 3.90000000e+01]
_actual
[1.00000000e-03 3.90000000e+01]
_predicted
Mean square distance
Wavelength
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0
20
40
60
80
100

40


Comparing spectrums
[10. 17.]
_actual
[10. 17.]
_predicted
Mean square distance
Wavelength
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
20
40
60
80
100


Comparing spectrums
[10. 37.]
_actual
[10. 37.]
_predicted
Mean square distance
Wavelength
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
0
20
40
60
80
100

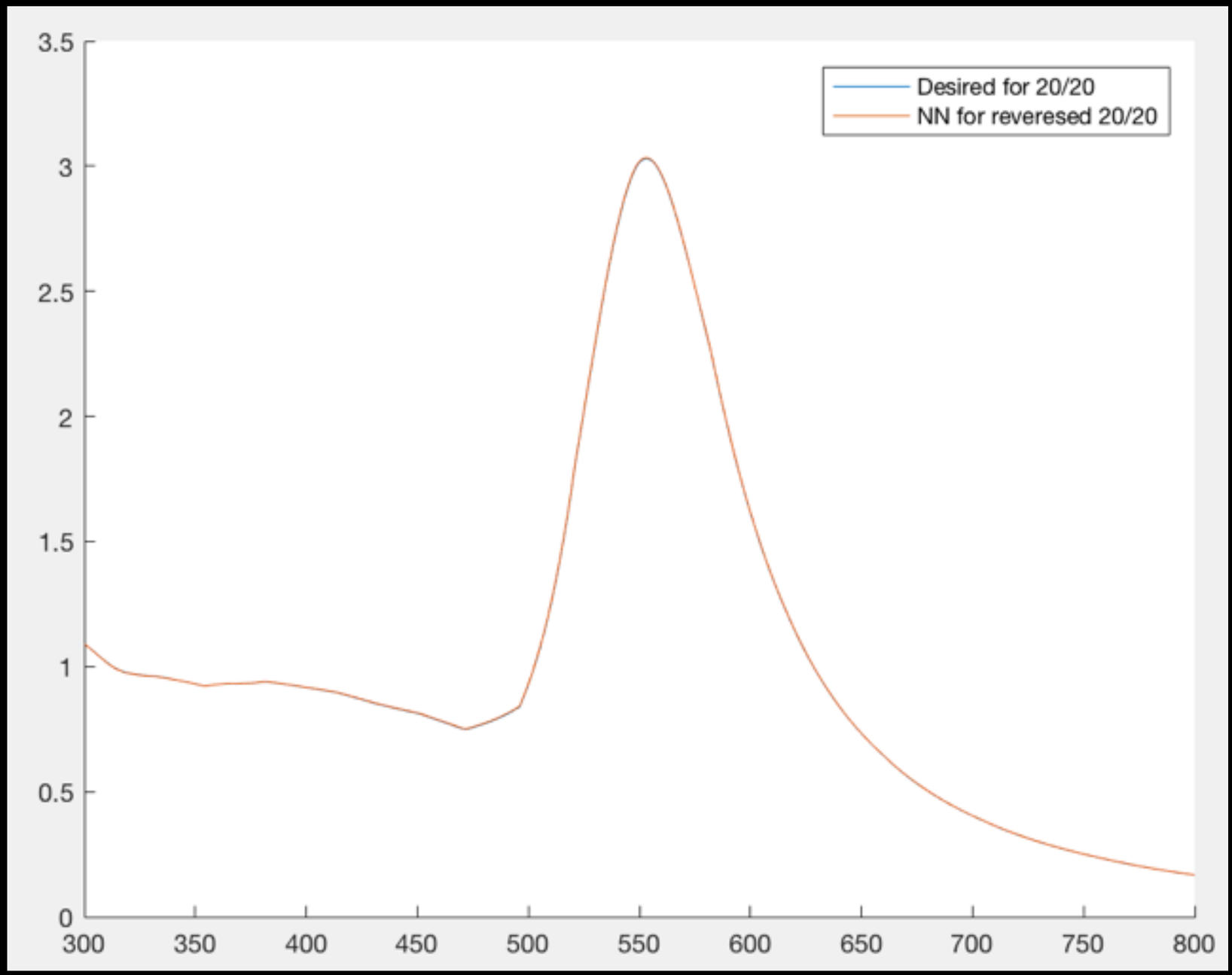

Comparing spectrums
[40. 20.]
_actual
[40. 20.]
_predicted
Mean square distance
Wavelength
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
0
20
40
60
80
100


Comparing spectrums
[40. 40.]
_actual
[40. 40.]
_predicted
Mean square distance
Wavelength
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
0
20
40
60
80
100

Now we reverse it for the two variables

We give it the spectrum, and it designs the geometry

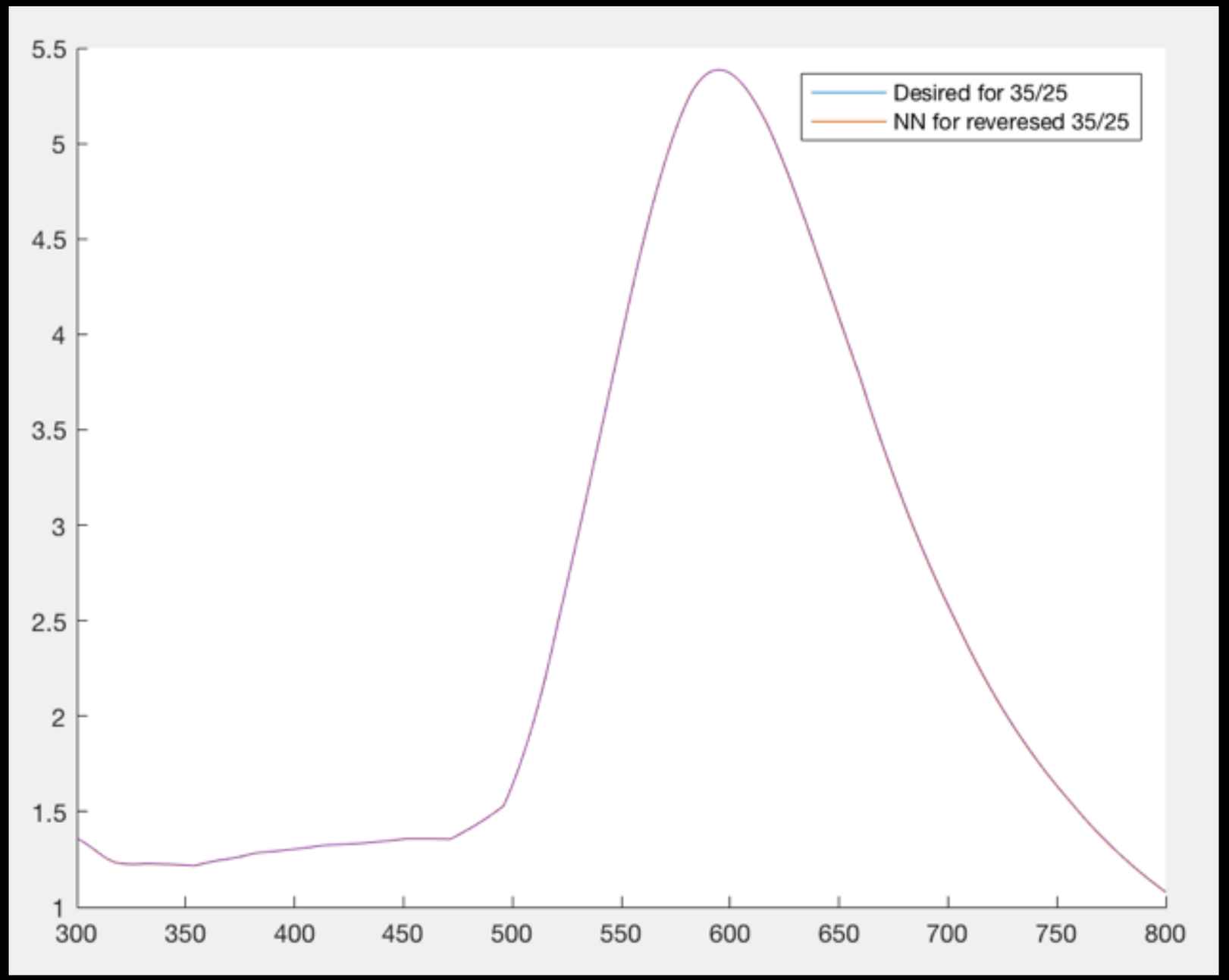
1 - Feeding in this spectrum:



Loss: .0272

It finds radi of 20.046 and 19.963nm. This spectrum was produced by 20 and 20nm. The spectrum for the radi that it found is graphed in orange.

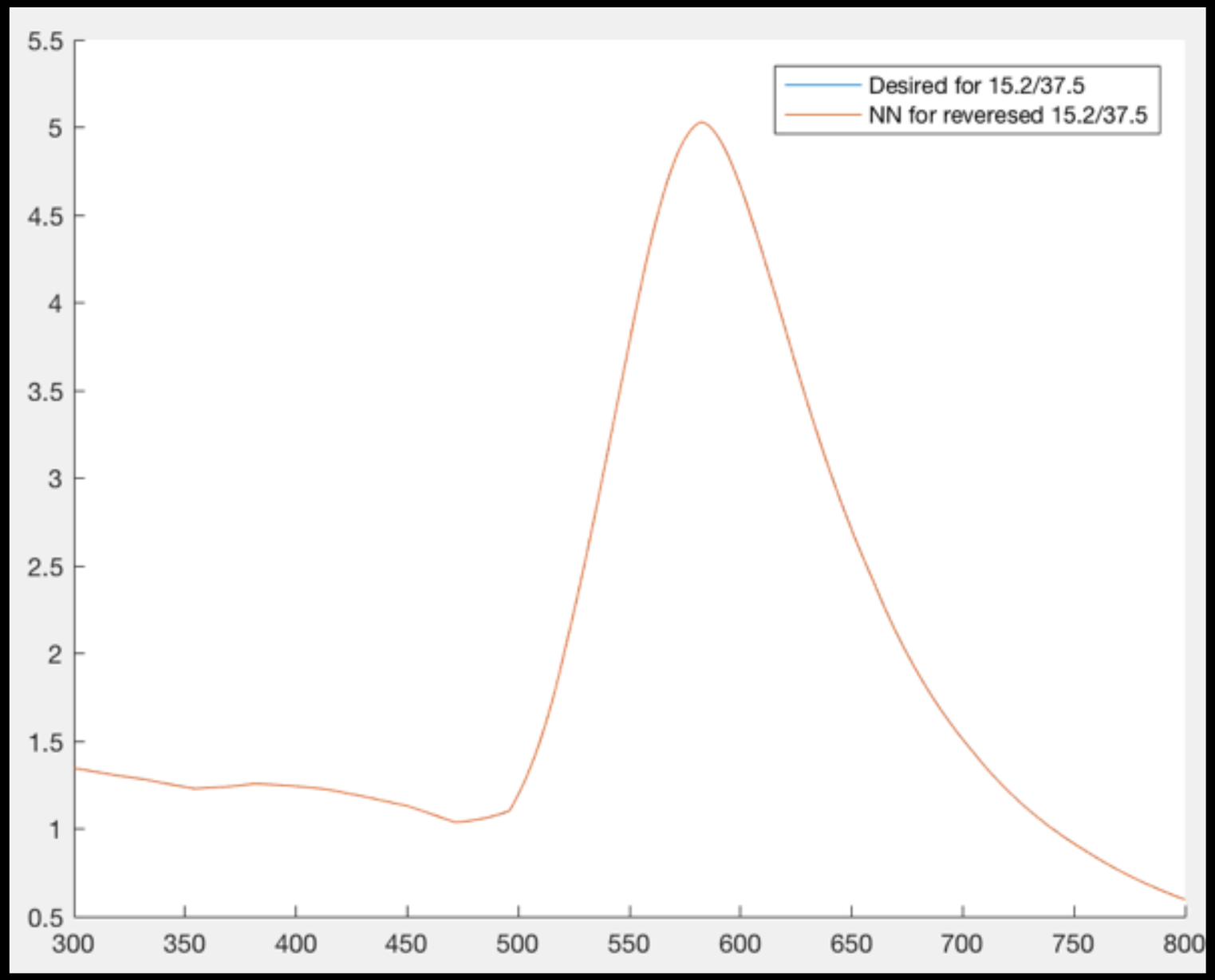
2 - Feeding in this spectrum:



Loss: .05207

It finds radi of 34.97225 and 25.05748 nm. This spectrum was produced by 35 and 25nm. The spectrum for the radi that it found is graphed in orange.

3 - Feeding in this spectrum:

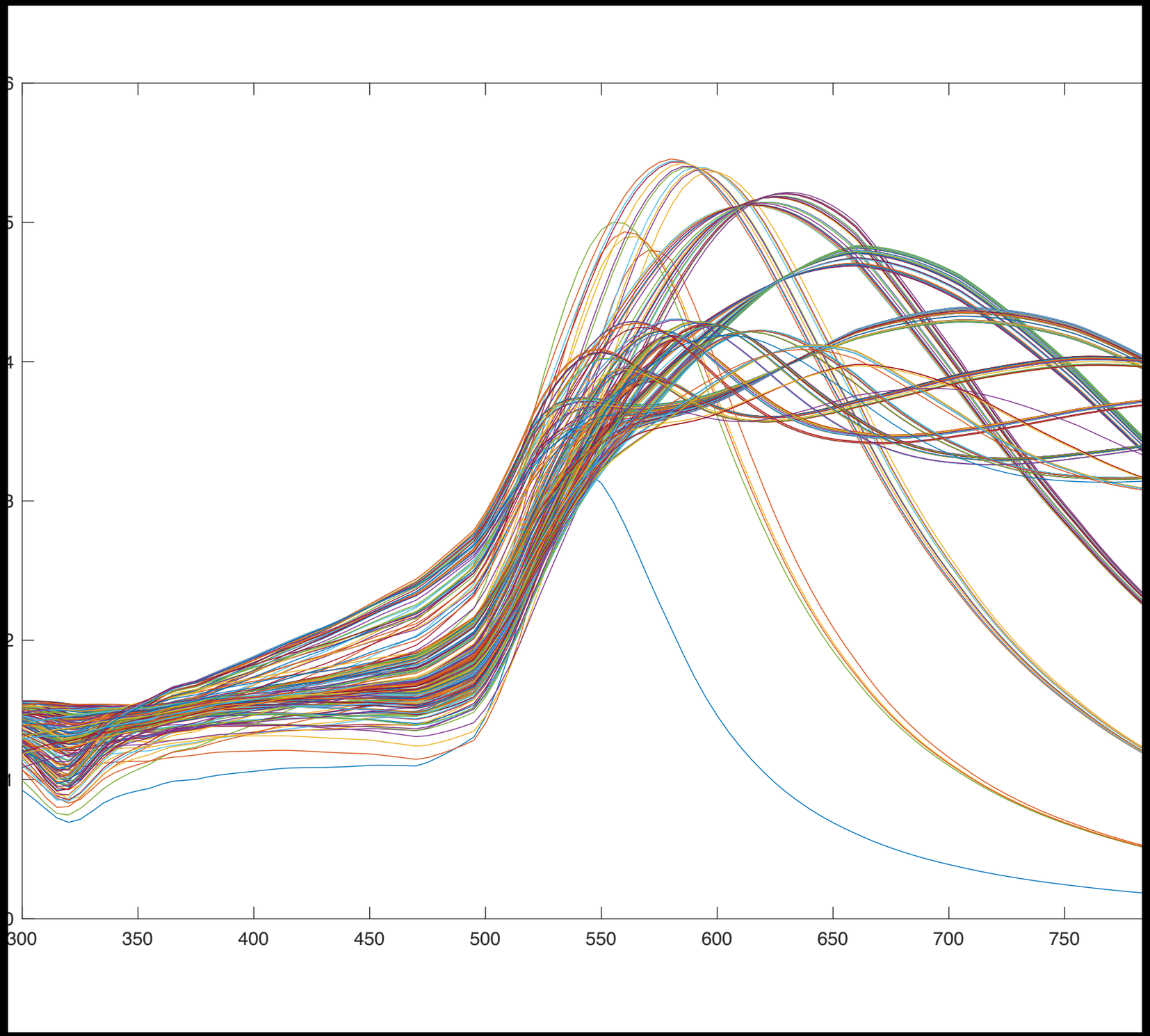


Loss: .07833

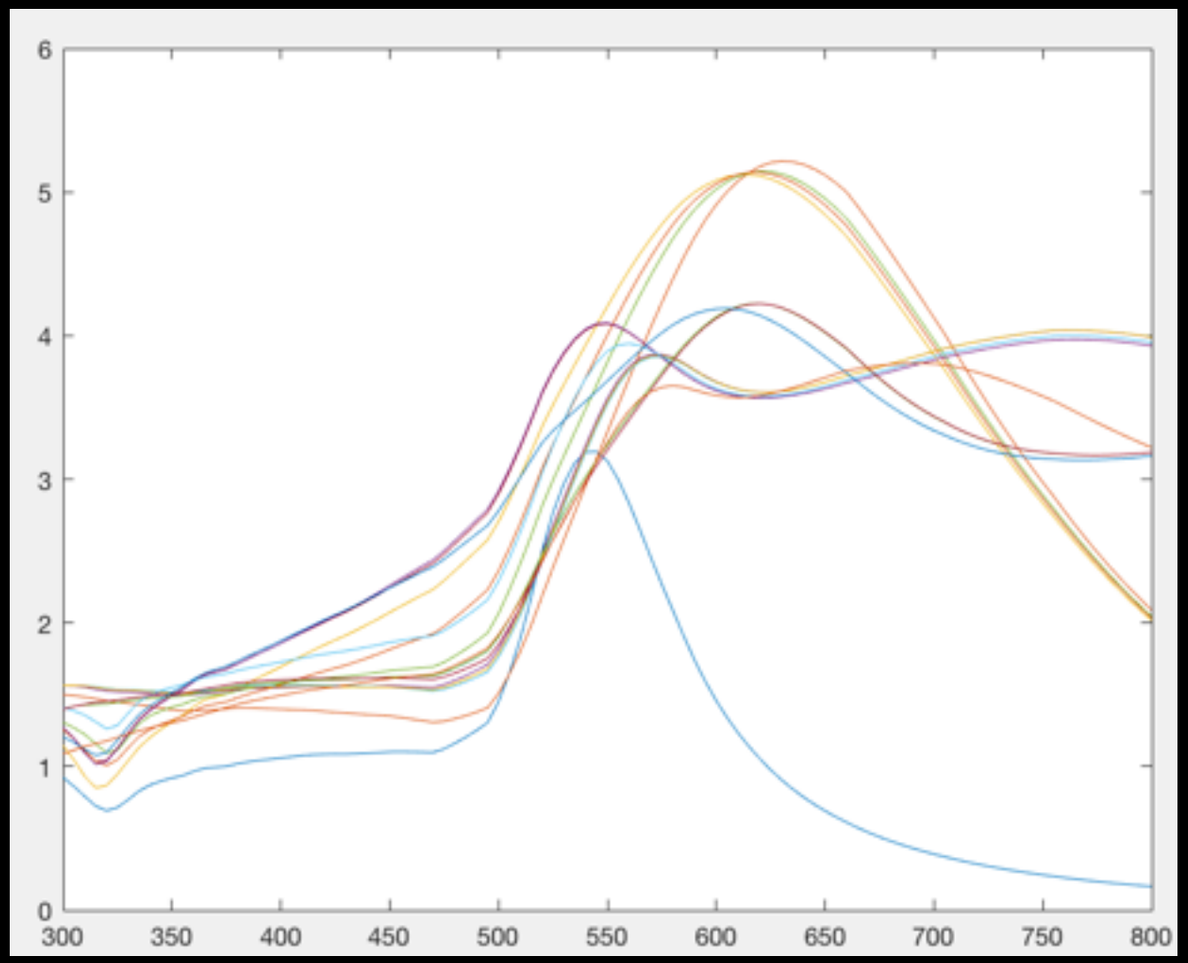
It finds radi of 15.299 and 37.3933 nm. This spectrum was produced by 15.2 and 37.5 nm. The spectrum for the radi that it found is graphed in orange.

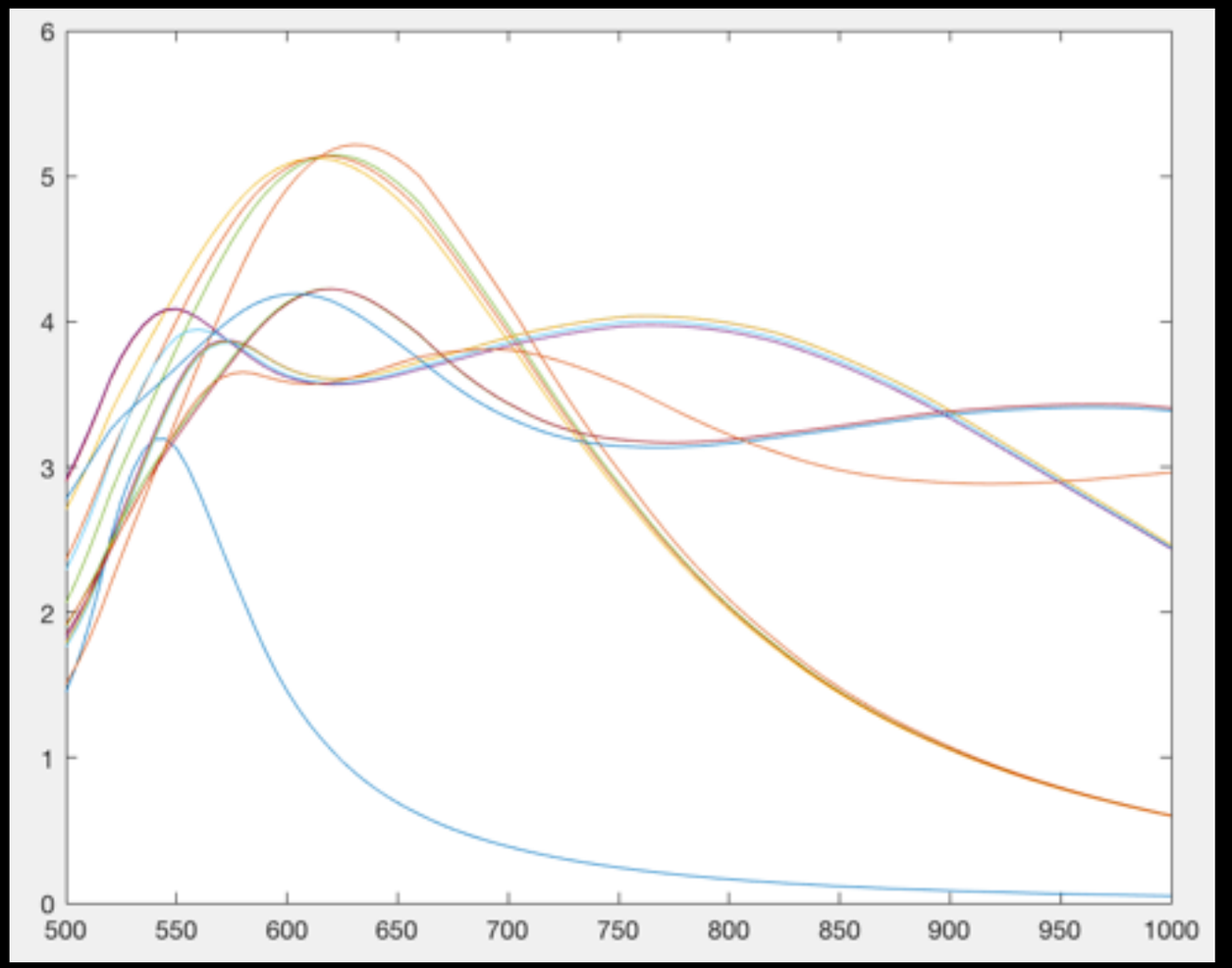
Now we train it for 4 radius, and we remove the 0 case (it just messes with things, and was causing the broken spectrums earlier)

Training Data
4 radi, gold core. 10:40, stepping 2.



Idea! (not currently done)





Shift the spectrum we are considering?

Step 2: Four-Layer Sphere with different materials (in progress)

- Architecture:
  - 4 in, 20x20x20x20x20x100 out
  - 3680 parameters
- Training Data:
  - 65,536 training points, 100 spectrum points per
    - Spectrum is 300-800nm, sampled every 5.
    - Training is 20:2:40, 20:2:40, gold/silver/gold/silver.
  - ~6,553,600
- Training results:
- Reversing:

Step 2: Four-Layer Sphere with different materials (in progress)