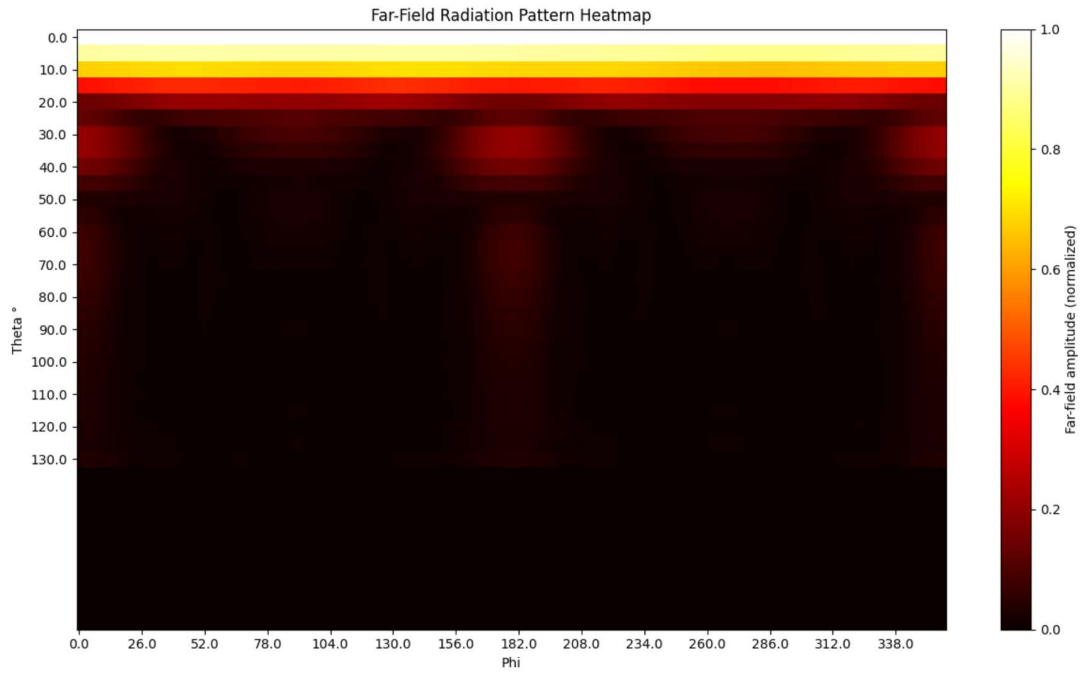
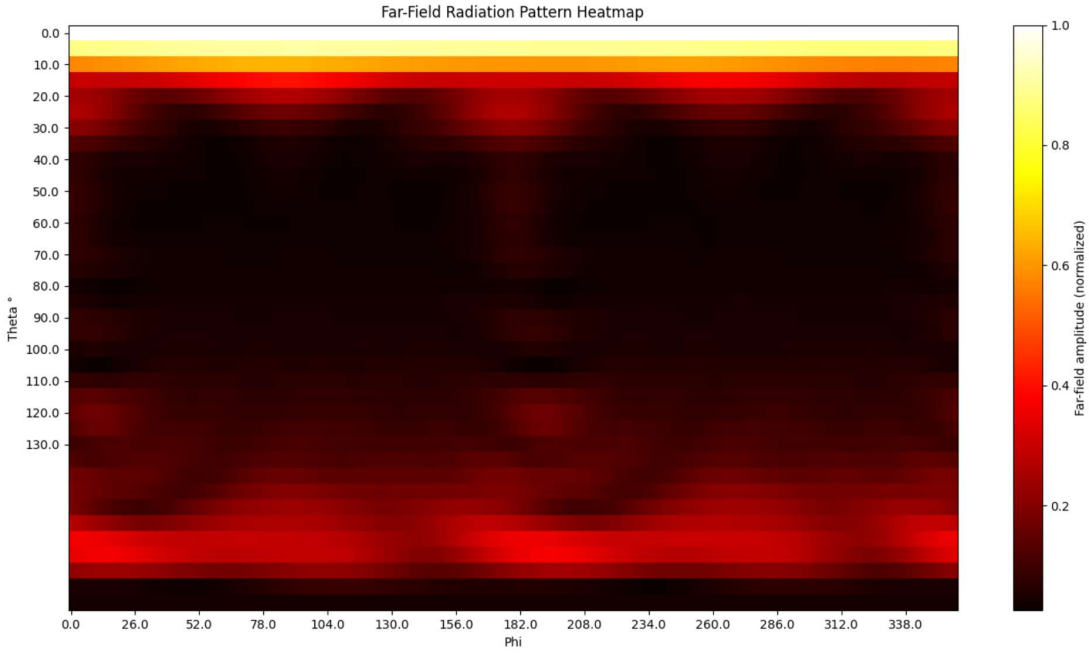
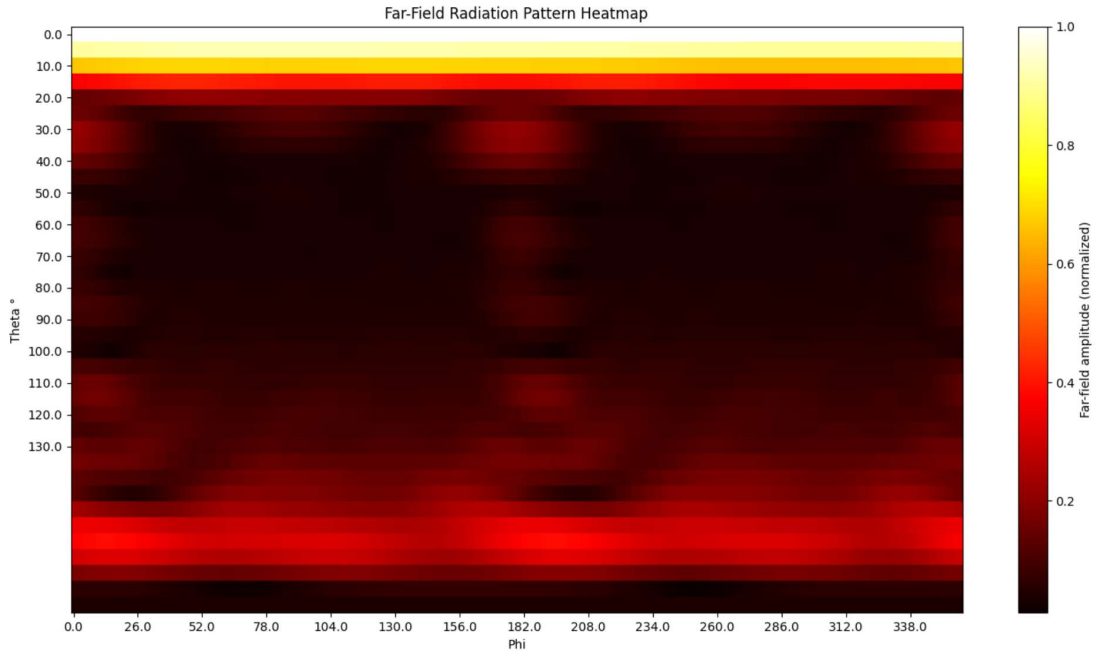


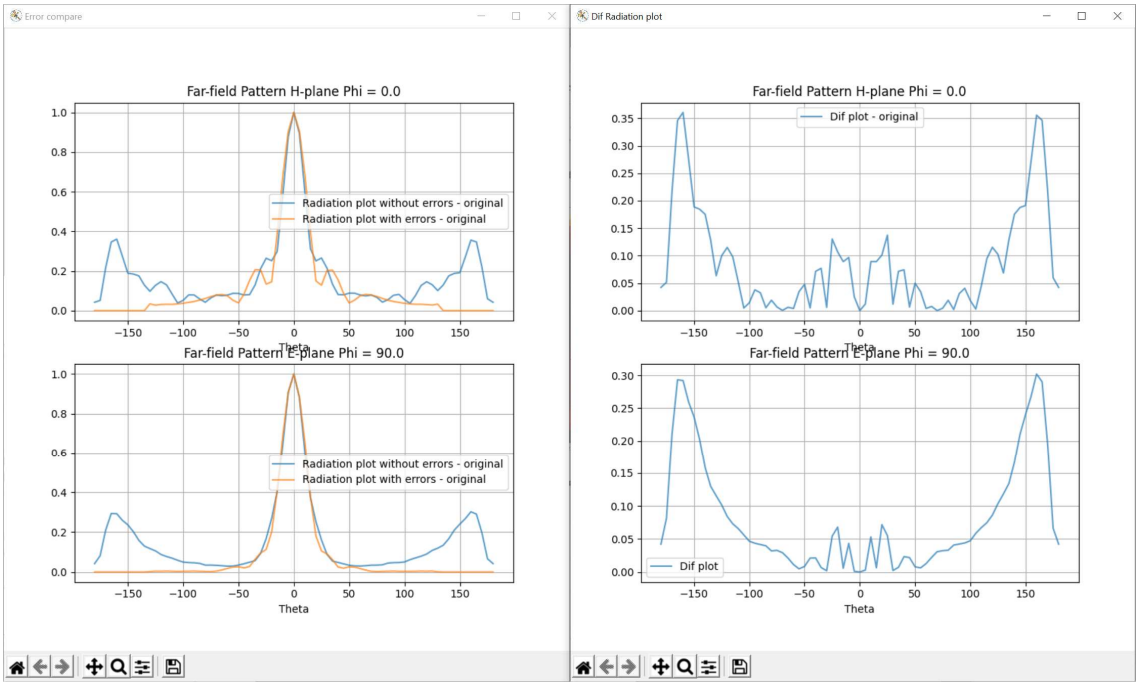
Loaded heatmap (Kim's transformed to far field)



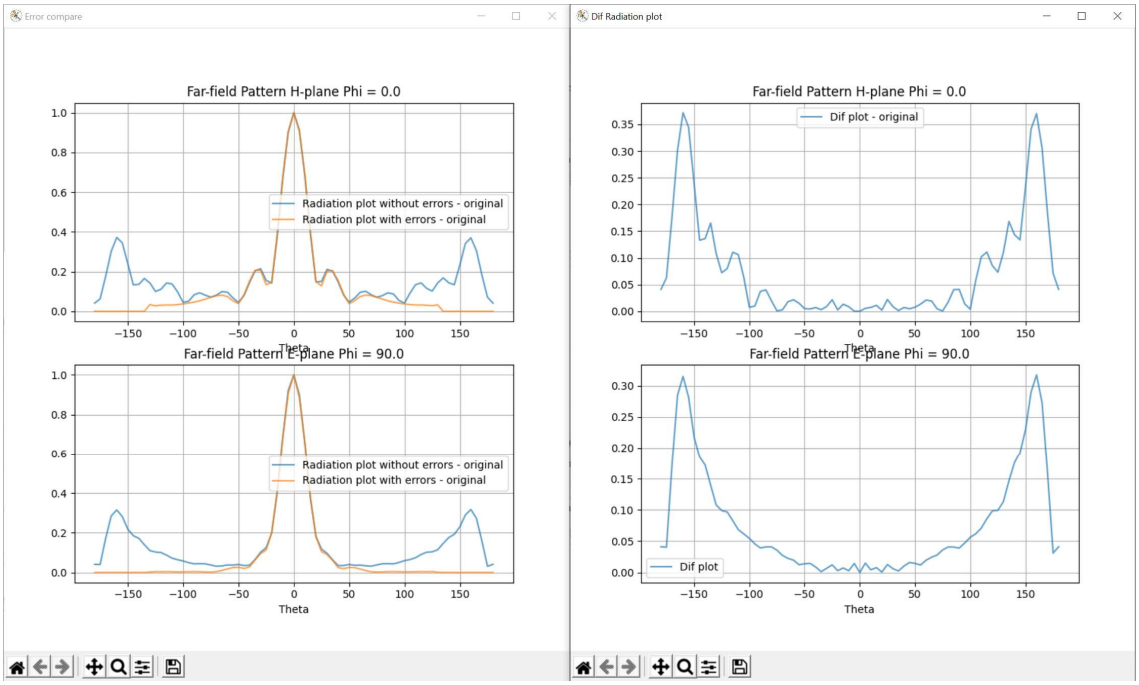
Our transform heatmap from Kim's transformed to 30 cm data set.



Our transformed with the same data set as Kim used.



Orange plot is Kim's, Blue is ours. Right plot is a plot of the difference between the two plots. (It is made with the 30 cm transformed dataset)
(Notice that in both the heatmap and the radiation diagram, there are large deviations at the end, however most of this is in the region that we did not have any data for.



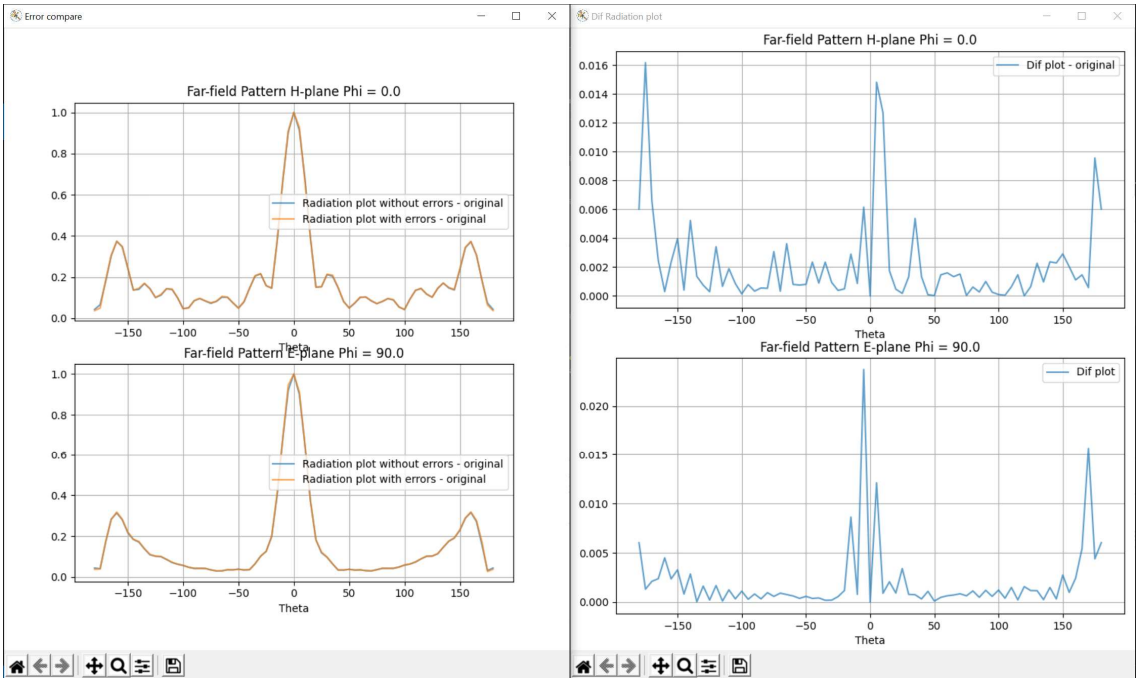
Orange and blue is the same as before, this time it is made with the same dataset Kim used.

From this we see that within ±100 degrees, the transformation at max dif is 5%. In our entire data range of ±130 degrees the max dif is 10%

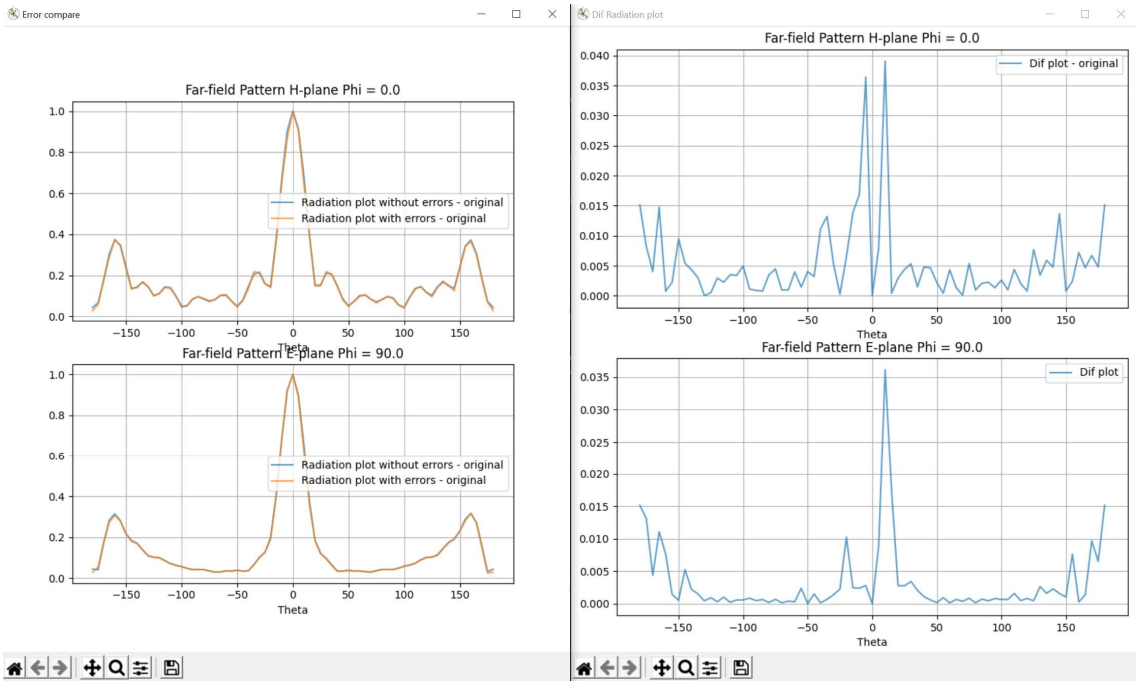
With Errors:

```python
# Function to introduce amplitude errors (modifies data in place)
def amplitude_errors(data, standard_deviation):
    for i in range(data.shape[0]):
        for j in range(data.shape[1]):
            # Apply amplitude error to both components (E_theta and E_phi)
            amplitude_error_theta = 1 + np.random.normal(0, standard_deviation)
            amplitude_error_phi = 1 + np.random.normal(0, standard_deviation)
            data[i, j, 0] *= amplitude_error_theta
            data[i, j, 1] *= amplitude_error_phi
```

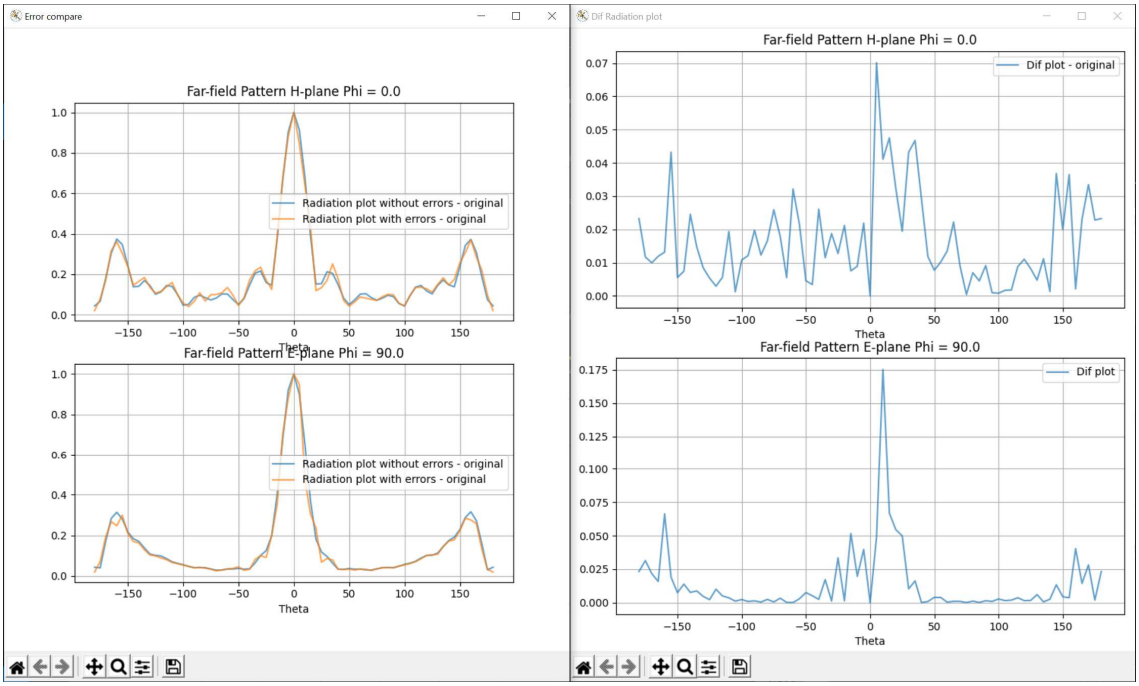
Python function for amplitude errors



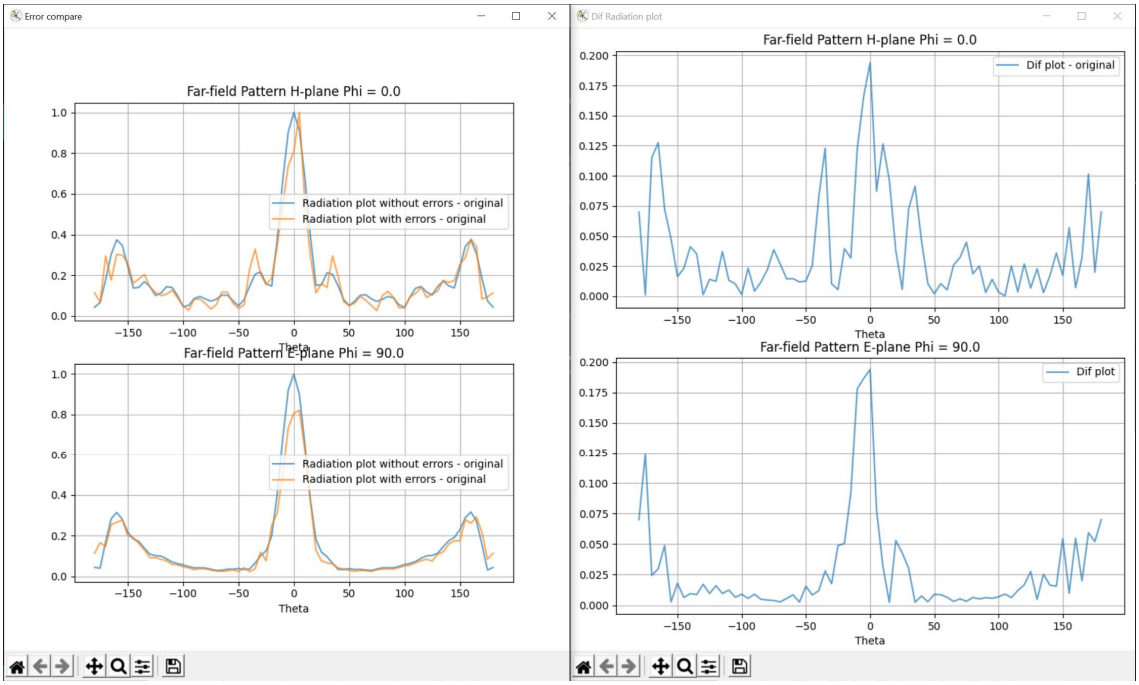
This is with amplitude errors with standard deviation = 0.1



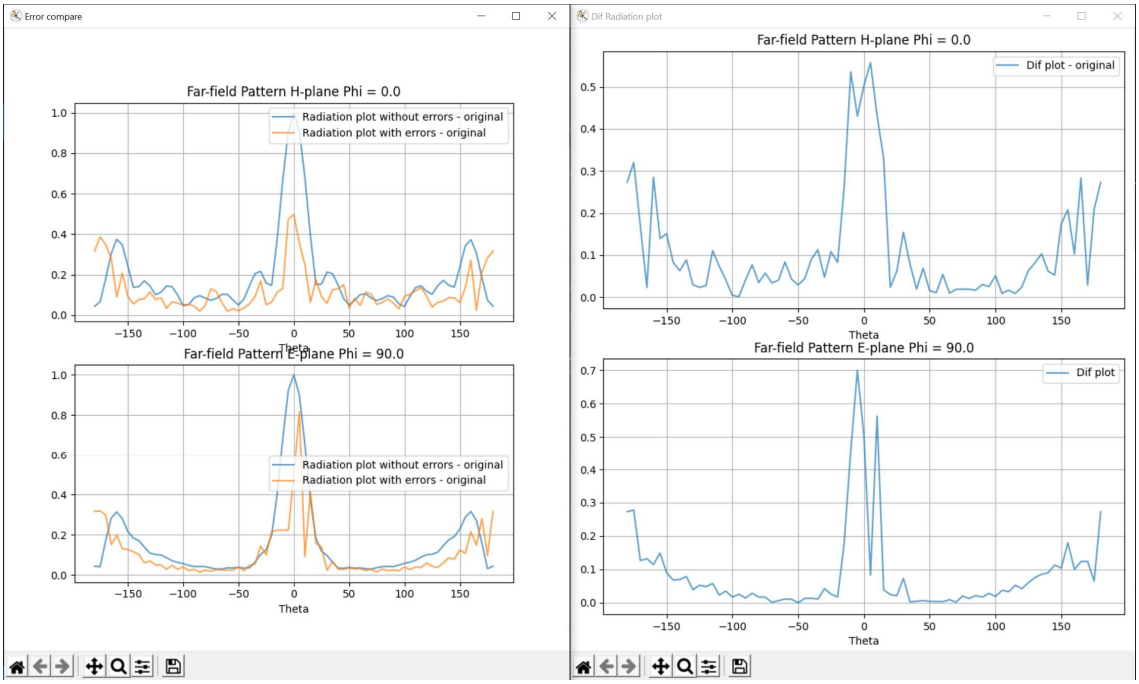
This is with amplitude errors with standard deviation = 0.1



This is with amplitude errors with standard deviation = 0.5



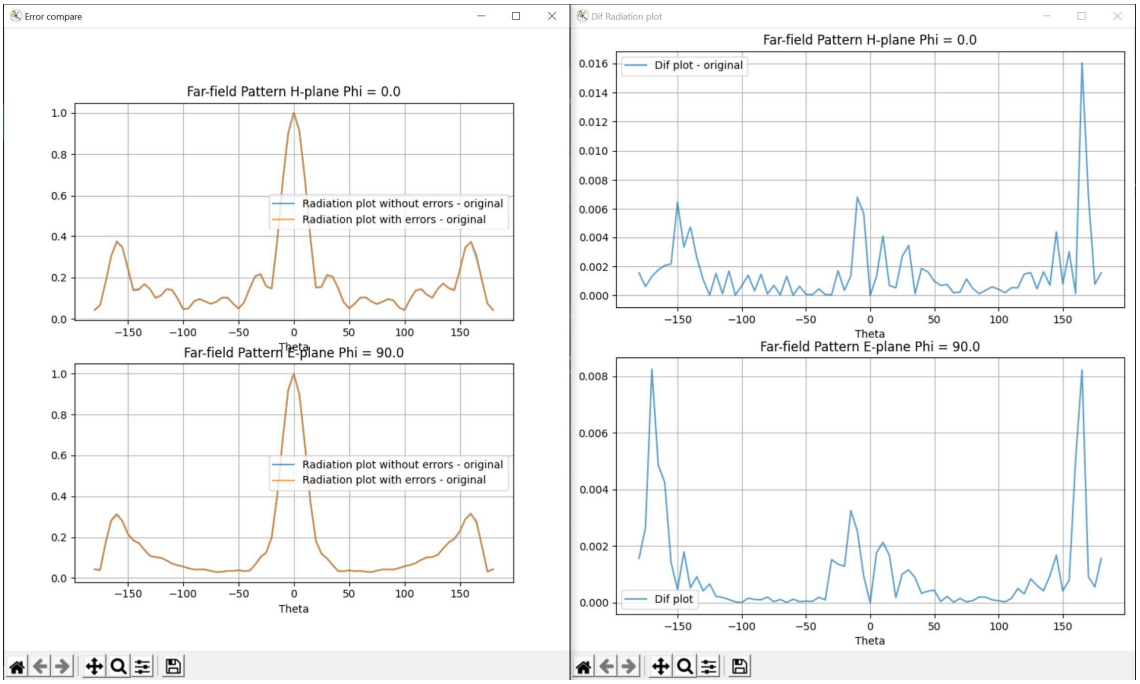
This is with amplitude errors with standard deviation = 1.



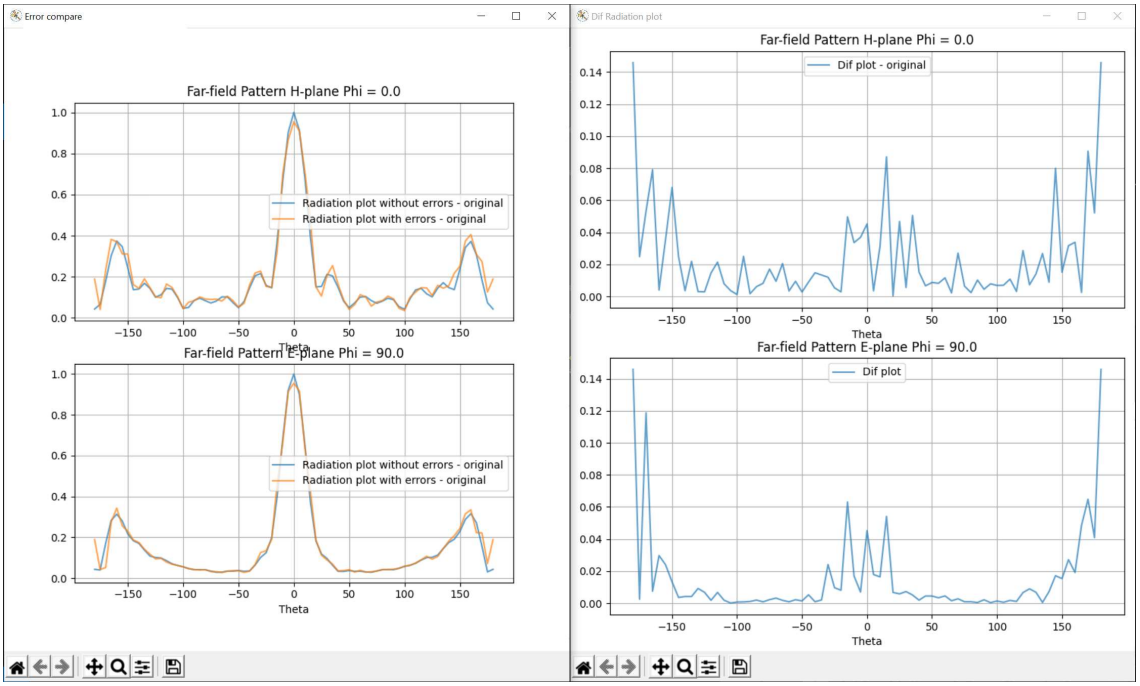
This is with amplitude errors with standard deviation = 5.

```python
# Function to introduce phase errors (modifies data in place)
def phase_errors(data, standard_deviation):
    for i in range(data.shape[0]):
        for j in range(data.shape[1]):
            # Apply phase error to both components (E_theta and E_phi)
            phase_error_theta = 2*np.pi * np.random.normal(0, standard_deviation)
            phase_error_phi = 2*np.pi * np.random.normal(0, standard_deviation)
            data[i, j, 0] *= np.exp(1j * phase_error_theta)
            data[i, j, 1] *= np.exp(1j * phase_error_phi)
```

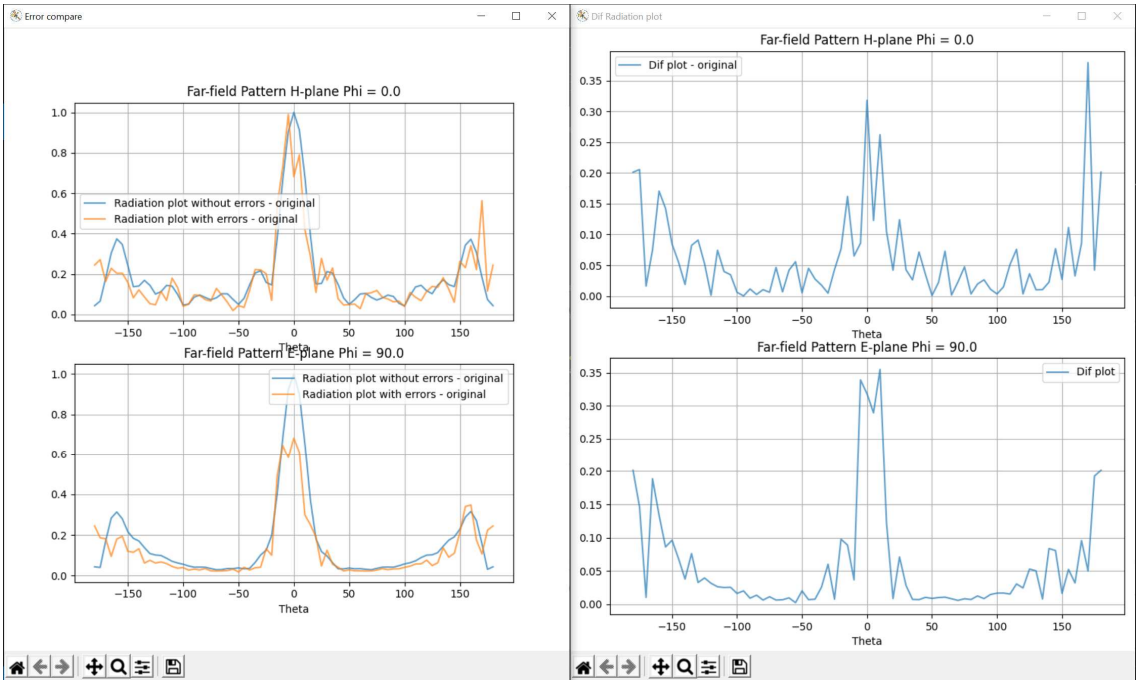
Function for phase errors



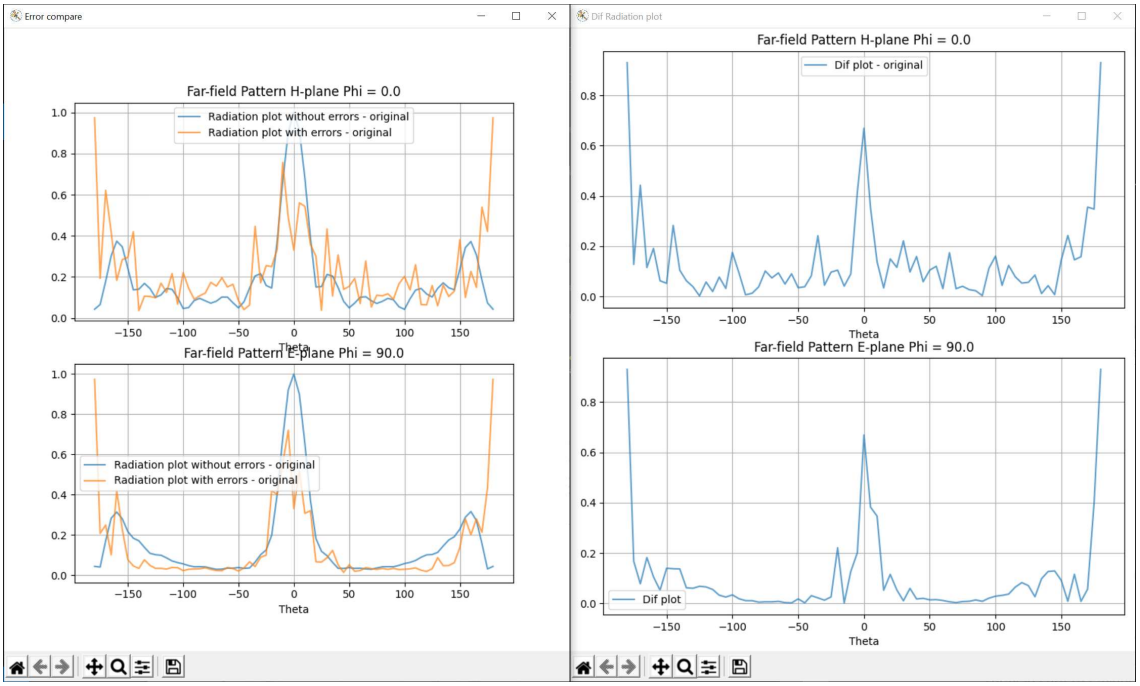
This is with phase errors with standard deviation = 0.01



This is with phase errors with standard deviation = 0.10

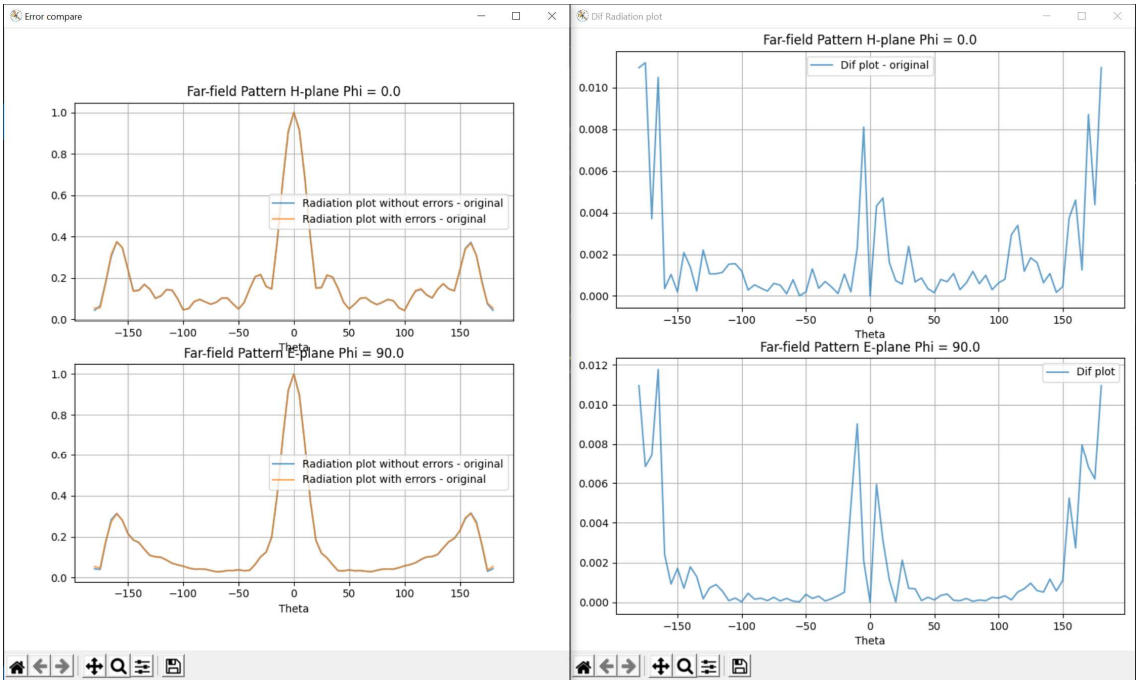


This is with phase errors with standard deviation = 0.20

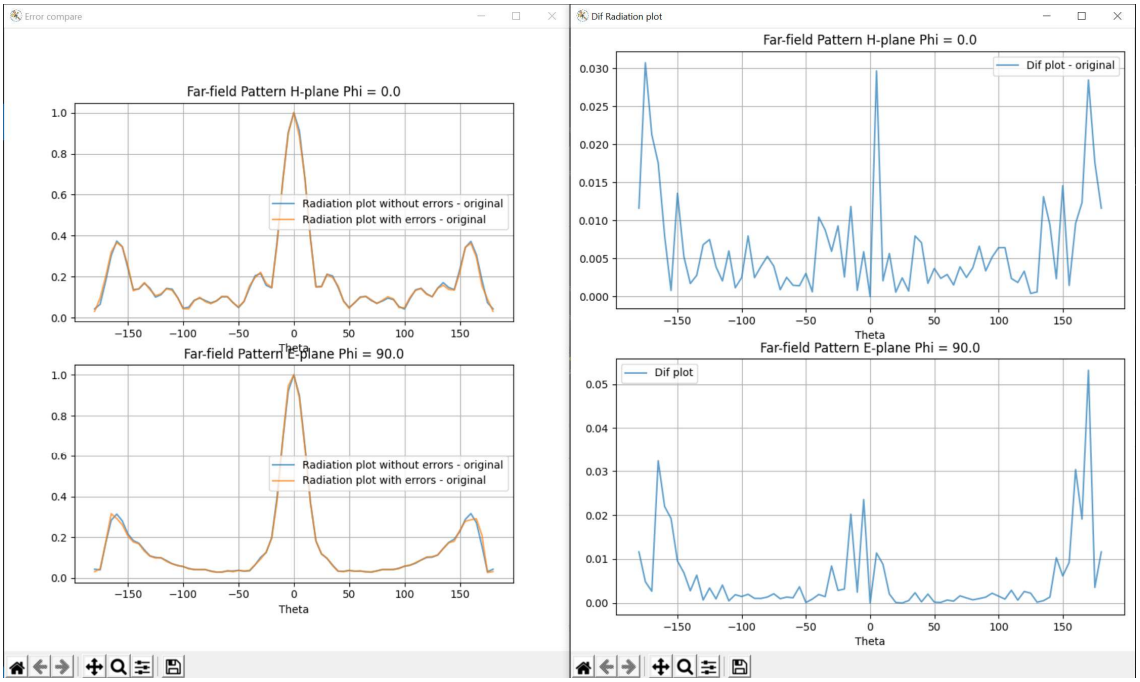


This is with phase errors with standard deviation = 0.50

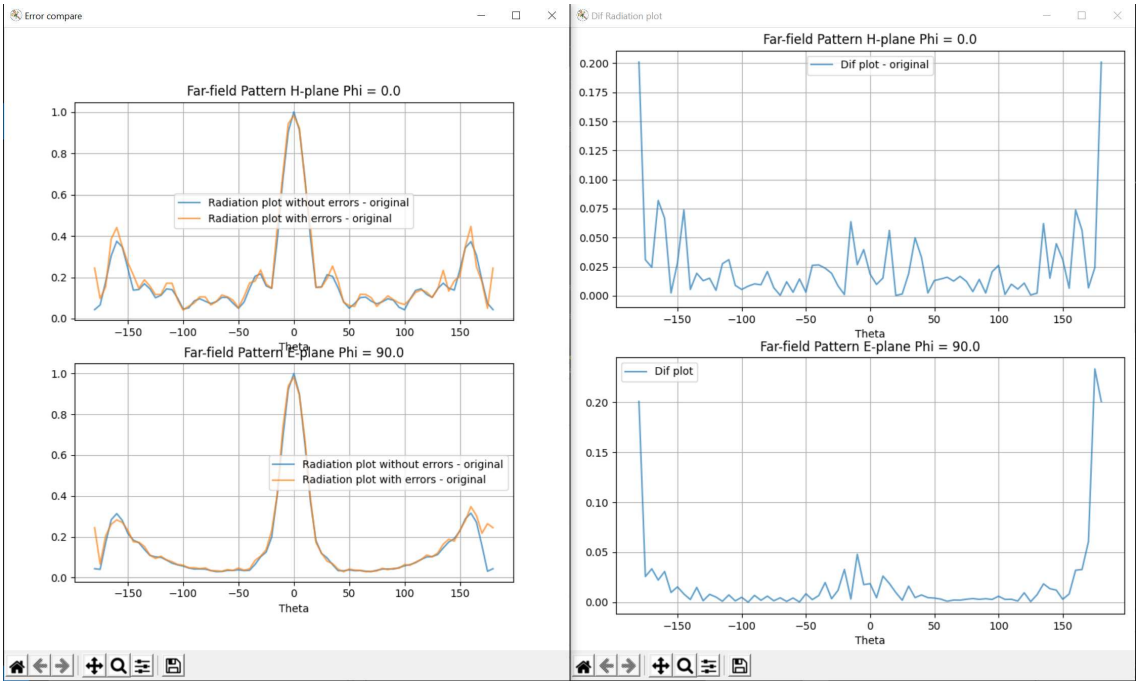
Now with both phase and amplitude errors!



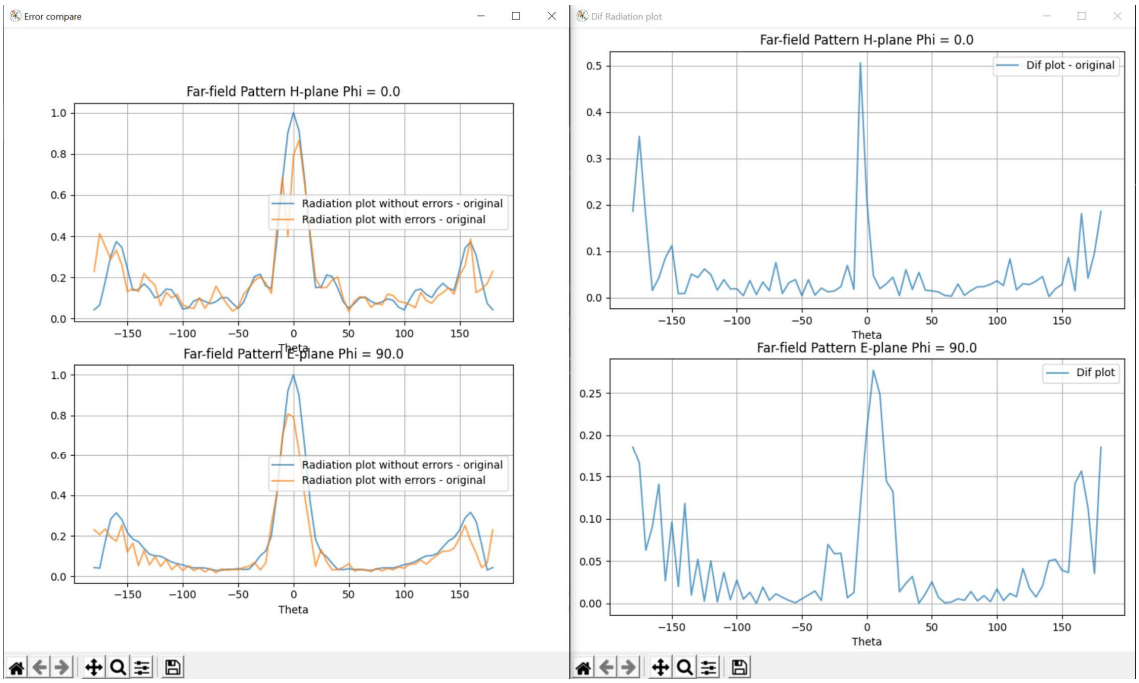
Both errors with standard deviation = 0.01



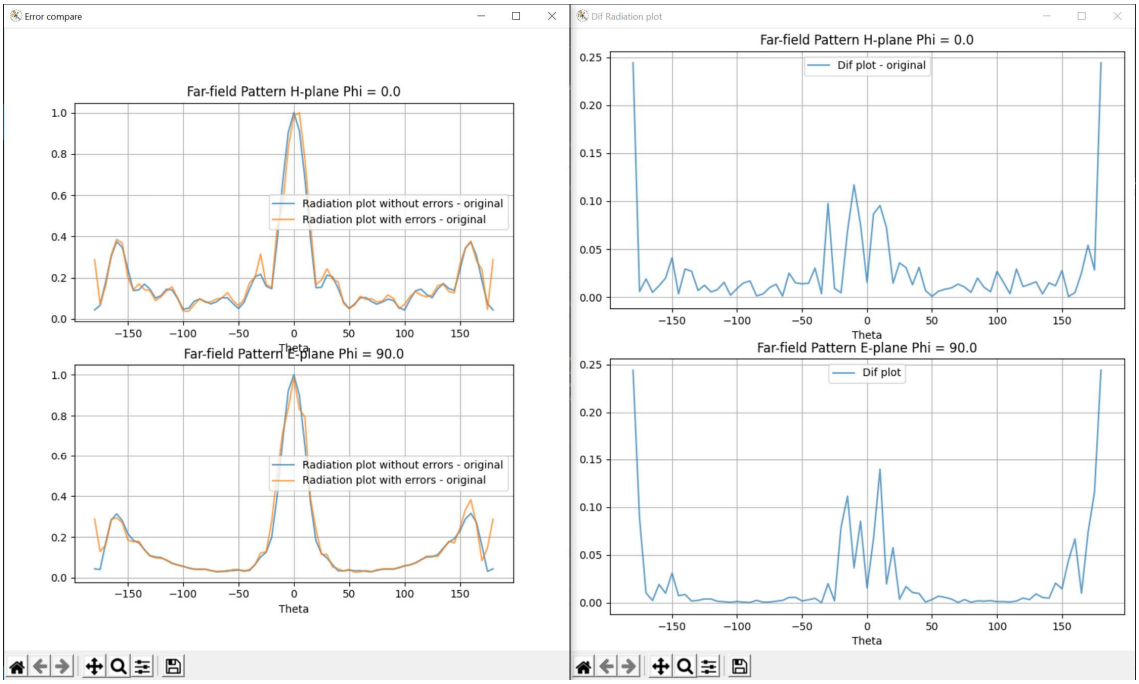
Both errors with standard deviation = 0.05



Both errors with standard deviation = 0.10



Both errors with standard deviation = 0.10



Phase error with standard deviation = 0.08 and amplitude errors with standard deviation = 0.4