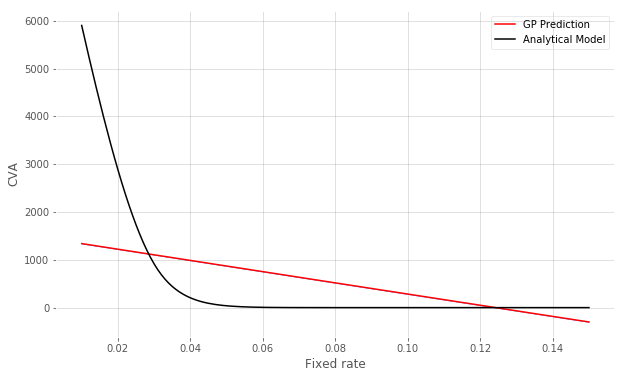
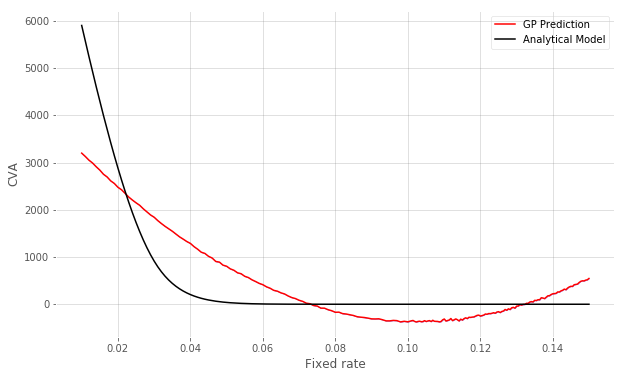
Keep other parameter the same, when the noise term is 0.1:

*K\_noise = Kernel + np.eye(training\_number) \* (1e-1)*



as noise approaches to 0, the predictive line curves, but still far from the test CVA:

*K\_noise = Kernel + np.eye(training\_number) \* (1e-5)*

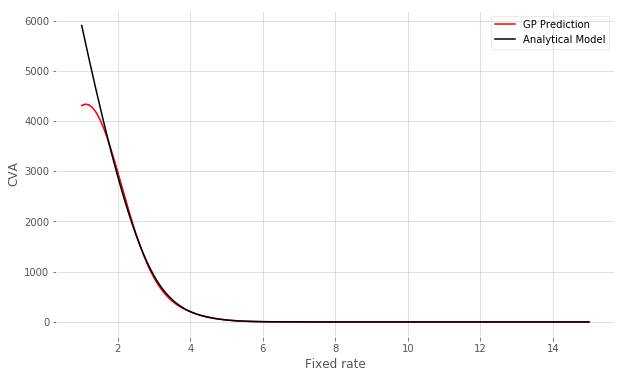


I checked the value of Kernel function, they are all quite close to 1, so I’m guessing the tiny change in fixed rate (training data) is too small for kernel function to reflect their relationship, so I tried to increase the change in fixed rate by multiplying 100, and make sigma\_n = 1:

*x\_train = np.array(x[idx], dtype='float32').reshape(training\_number, 1) \* 100*

*x\_test = np.array(x, dtype='float32').reshape(len(x), 1) \* 100*

*K\_noise = Kernel + np.eye(training\_number)*

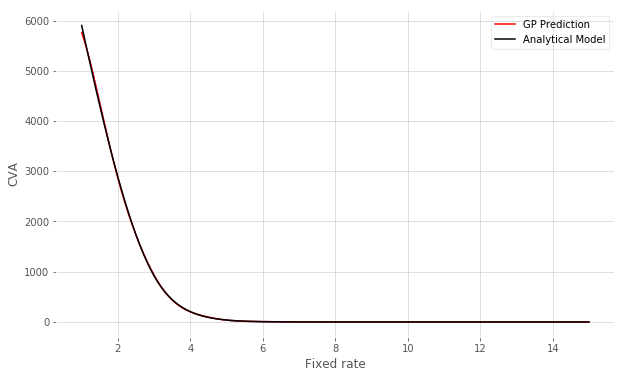


if I make the noise term smaller:

*x\_train = np.array(x[idx], dtype='float32').reshape(training\_number, 1) \* 100*

*x\_test = np.array(x, dtype='float32').reshape(len(x), 1) \* 100*

*K\_noise = Kernel + np.eye(training\_number) \* (1e-2)*



then the prediction matches the test data well.

I am not sure if this is a good idea to get the prediction match the test data, but it seems verified the problem of linear prediction from tiny change in the fixed rate.