


```
In [3]: ##matplotlib notebook
import os
import sys
sys.path.append("numpy_path")
import numpy as np
import struct
from matplotlib import pyplot as plt
import keras
from keras.models import Sequential, load_model
from keras.layers import Dense, Dropout, Activation
from keras.optimizers import RMSprop
import keras.callbacks as cb
from keras.callbacks import EarlyStopping, ModelCheckpoint
from math import cos, sin, pi
from statistics import mean
import os.path
import math
shape_size = 48

# define Loss history
class LossHistory(cb.Callback):
    def on_train_begin(self, logs={}):
        self.losses = []
    def on_batch_end(self, batch, logs={}):
        batch_loss = logs.get('loss')
        self.losses.append(batch_loss)

#plot Losses
def plot_losses(losses):
    plt.plot(losses)
    plt.title('Loss per batch')
    plt.show()

def feature_scaling(X):
    X = X.T
    for i in range(7):
        mean = X[i].mean()
        std = X[i].std()
        X[i] = [(x - mean)/std for x in X[i]]
    return X.T

# input dimension
in_dim = 6
out_dim = 200
def init_model():
    model = Sequential()
    model.add(Dense(20, input_dim=in_dim))
    model.add(Dropout(0.2))
    model.add(Activation('relu'))
    model.add(Dense(500))
    model.add(Dropout(0.2))
    model.add(Activation('relu'))
    model.add(Dense(200))
    model.add(Dropout(0.2))
    model.add(Activation('relu'))
    model.add(Dense(200))
```

```
model.add(Activation('relu'))
model.add(Dense(200))
model.add(Dropout(0.2))
model.add(Activation('relu'))
#     model.add(Dense(70))
#     model.add(Dropout(0.2))
#     model.add(Activation('relu'))
#     model.add(Dense(100))
#     model.add(Activation('relu'))
model.add(Dense(out_dim))
model.add(Activation('sigmoid'))
# use mean squared error to measure the losses
model.compile(loss=keras.losses.mean_squared_error,
              optimizer=keras.optimizers.Adam(lr = 0.001),
              metrics=['accuracy'])
return model
```

```

In [6]: data_size = 0
dummy1 = [0]*200
dummy2 = [0]*6
SP = np.array(np.reshape(dummy1, (1, 200)))
SH = np.array(np.reshape(dummy2, (1, 6)))
for i in range(2, 42):
    path = 'meep_code/data/DATA'+str(i)
    if not os.path.exists(path):
        #miss.append(i)
        print('Missing batch:' + str(i))
        continue

    files = next(os.walk(path))[2] #dir is your directory path as string]
    num_data = len(files)
    data_size += num_data
    skip = []

    coordinates = np.genfromtxt('meep_code/data/DATA'+str(i)+'_sh.txt')
    xc, yc = coordinates[:, 0], coordinates[:, 1]
    xc = np.reshape(xc, (num_data, shape_size))
    yc = np.reshape(yc, (num_data, shape_size))

    for j in range(num_data):
        tmp = np.genfromtxt(path+'/'+str(i)+'_sp'+str(j)+'.txt')
        valid = True
        for q in range(200):

            if math.isnan(float(tmp[q])):
                print('Batch '+str(i)+'\tsample '+str(j)+' has NAN value')
                valid = False
                break
            if tmp[q] > 3:
                print('Batch '+str(i)+'\tsample '+str(j)+' has extreme value')
                valid = False
                break
        if not valid:
            #skip.append(j)
            continue
        SP = np.concatenate((SP, np.reshape(tmp, (1, 200))))
        tmp = []
        for q in range(6):
            tmp.append(math.sqrt(xc[j][q]**2 + yc[j][q]**2))
        SH = np.concatenate((SH, np.reshape(np.array(tmp), (1, 6))))
        #print(SP.shape)
    #print(files)

    # coordinates = np.genfromtxt('meep_code/data/DATA'+str(i)+'_sh.txt')
    # xc, yc = coordinates[:, 0], coordinates[:, 1]
    # xc = np.reshape(xc, (num_data, shape_size))
    # yc = np.reshape(yc, (num_data, shape_size))
    # for j in range(num_data):
    #     if j in skip:
    #         continue
    #     tmp = []
    #     for q in range(6):
    #         tmp.append(math.sqrt(xc[j][q]**2 + yc[j][q]**2))
    #     SH = np.concatenate((SH, np.reshape(np.array(tmp), (1, 6))))

```

```

    print('Batch '+str(i)+' has \t'+str(num_data))
#     freq = np.genfromtxt('meep_code/data/freq.txt')
#     for j in range(num_data):
#         print(j)
#         plt.ylim(0, 1.1)
#         plt.plot(freq, SP[data_size - num_data + j + 1])
#         plt.show()
#         plt.axis('equal')
#         plt.ylim(-0.5, 0.5)
#         plt.xlim(-0.5, 0.5)
#         plt.plot(xc[j], yc[j])
#         plt.fill(xc[j], yc[j])
#         plt.show()

```

```

Batch 2 has      7
Batch 3 has      6
Batch 4 has     17
Batch 5 has      5
Batch 6 has     14
Batch 7 has     14
Batch 8 has     40
Batch 9 has      5
Batch 10 has    14
Batch 11 has    14
Batch 12 has    40
Batch 13 has    13
Batch 14 has    37
Batch 15 has    37
Missing batch:16
Batch 17 has     5
Batch 18 has    14
Batch 19 has    14
Batch 20 has    40
Batch 21 has    12

```

```

In [7]: ▶ DATA = np.append(SP, SH, axis = 1)
        np.random.shuffle(DATA)

        Y = DATA[:, :200]
        X = DATA[:, 200:]

        train_size = int(len(DATA) * 0.8)

        train_X = X[0:train_size, :]
        train_Y = Y[0:train_size, :]
        test_X = X[train_size:, :]
        test_Y = Y[train_size:, :]

```

```
In [4]: ▶ model = init_model()
history = LossHistory()
# when training, using minibatch seems to be pretty good
model.fit(train_X, train_Y, epochs=1000, batch_size=10,
          callbacks=[history],
          validation_data=(test_X, test_Y), verbose=2)
score = model.evaluate(test_X, test_Y, batch_size=10)
print(score)
plot_losses(history.losses)
```

WARNING: Logging before flag parsing goes to stderr.

W0726 15:39:43.332967 139638701143872 deprecation_wrapper.py:119] From /home/shengnan/anaconda3/envs/mp/lib/python3.6/site-packages/keras/backend/tensorflow_backend.py:74: The name tf.get_default_graph is deprecated. Please use tf.compat.v1.get_default_graph instead.

W0726 15:39:43.394692 139638701143872 deprecation_wrapper.py:119] From /home/shengnan/anaconda3/envs/mp/lib/python3.6/site-packages/keras/backend/tensorflow_backend.py:517: The name tf.placeholder is deprecated. Please use tf.compat.v1.placeholder instead.

W0726 15:39:43.406083 139638701143872 deprecation_wrapper.py:119] From /home/shengnan/anaconda3/envs/mp/lib/python3.6/site-packages/keras/backend/tensorflow_backend.py:4138: The name tf.random_uniform is deprecated. Please use tf.random.uniform instead.

W0726 15:39:43.460803 139638701143872 deprecation_wrapper.py:119] From /home/shengnan/anaconda3/envs/mp/lib/python3.6/site-packages/keras/backend/tensorflow_backend.py:133: The name tf.placeholder_with_default is deprecated. Please use tf.compat.v1.placeholder_with_default instead.

```
In [11]: ▶ model.save('Simulator')
```

```
In [4]: model = load_model('Simulator')
```

WARNING: Logging before flag parsing goes to stderr.

W0726 16:27:41.996426 139737506166592 deprecation_wrapper.py:119] From /home/shengnan/anaconda3/envs/mp/lib/python3.6/site-packages/keras/backend/tensorflow_backend.py:517: The name tf.placeholder is deprecated. Please use tf.compat.v1.placeholder instead.

W0726 16:27:42.049476 139737506166592 deprecation_wrapper.py:119] From /home/shengnan/anaconda3/envs/mp/lib/python3.6/site-packages/keras/backend/tensorflow_backend.py:4138: The name tf.random_uniform is deprecated. Please use tf.random.uniform instead.

W0726 16:27:42.072472 139737506166592 deprecation_wrapper.py:119] From /home/shengnan/anaconda3/envs/mp/lib/python3.6/site-packages/keras/backend/tensorflow_backend.py:131: The name tf.get_default_graph is deprecated. Please use tf.compat.v1.get_default_graph instead.

W0726 16:27:42.084743 139737506166592 deprecation_wrapper.py:119] From /home/shengnan/anaconda3/envs/mp/lib/python3.6/site-packages/keras/backend/tensorflow_backend.py:133: The name tf.placeholder_with_default is deprecated. Please use tf.compat.v1.placeholder_with_default instead.

W0726 16:27:42.110102 139737506166592 deprecation.py:506] From /home/shengnan/anaconda3/envs/mp/lib/python3.6/site-packages/keras/backend/tensorflow_backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a future version.

Instructions for updating:

Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.

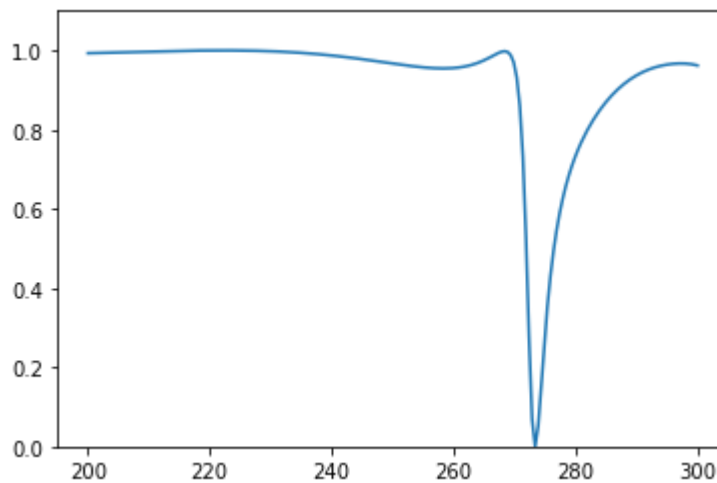
W0726 16:27:42.444418 139737506166592 deprecation_wrapper.py:119] From /home/shengnan/anaconda3/envs/mp/lib/python3.6/site-packages/keras/backend/tensorflow_backend.py:174: The name tf.get_default_session is deprecated. Please use tf.compat.v1.get_default_session instead.

W0726 16:27:42.753975 139737506166592 deprecation_wrapper.py:119] From /home/shengnan/anaconda3/envs/mp/lib/python3.6/site-packages/keras/optimizers.py:790: The name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

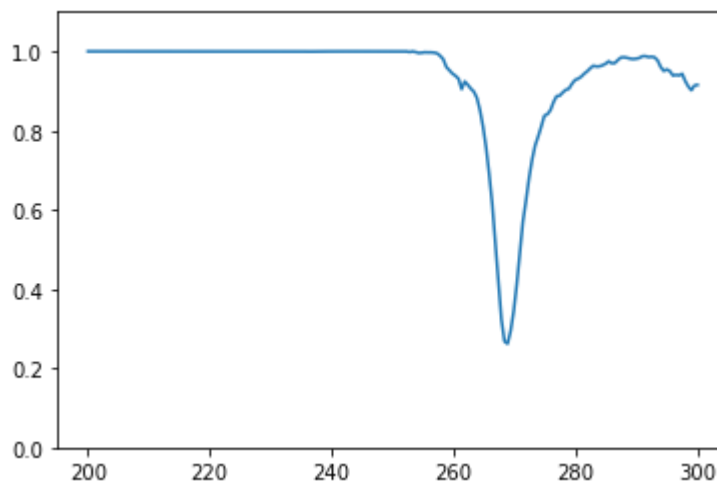
```
In [ ]: x = np.genfromtxt('meep_code/data/SP_xaxis.txt')
for i in range(len(test_X)):
    print('Test '+str(i))
    print('True spectrum: ')
    plt.ylim(0, 1.1)
    plt.plot(x, test_Y[i])
    plt.show()
    print('Predicted spectrum: ')
    plt.ylim(0, 1.1)
    plt.plot(x, np.reshape(model.predict(np.reshape(test_X[i], (1, 6))), (200, 1)))
    plt.show()
```

Test 0

True spectrum:

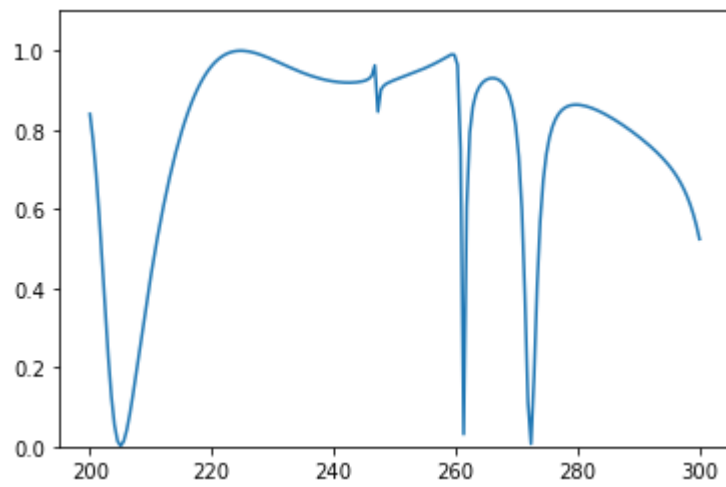


Predicted spectrum:

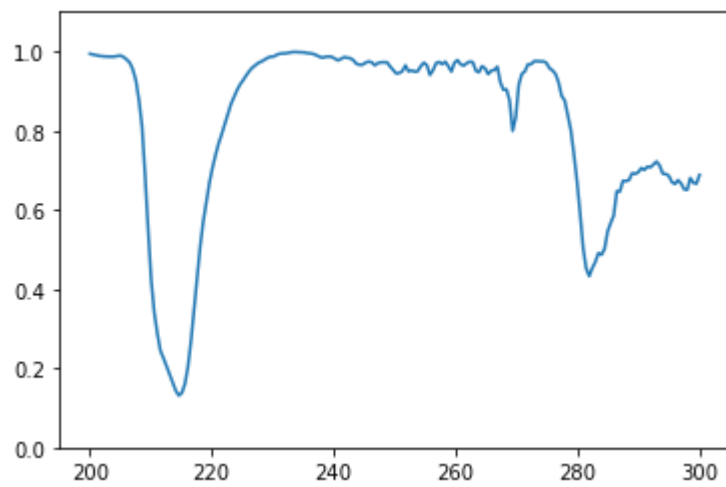


Test 1

True spectrum:

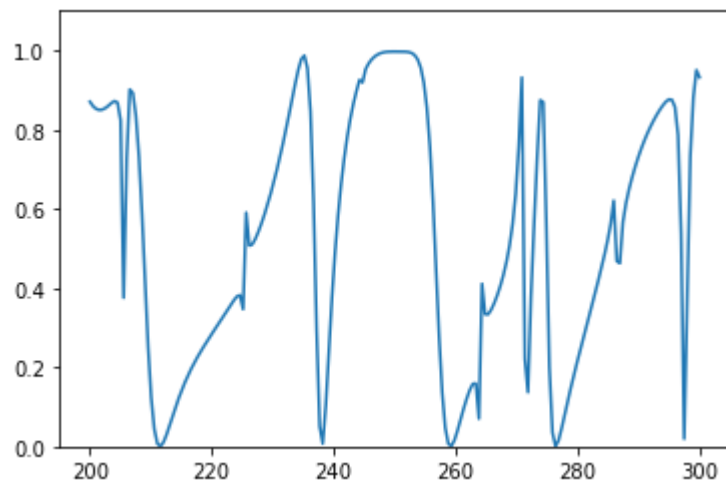


Predicted spectrum:

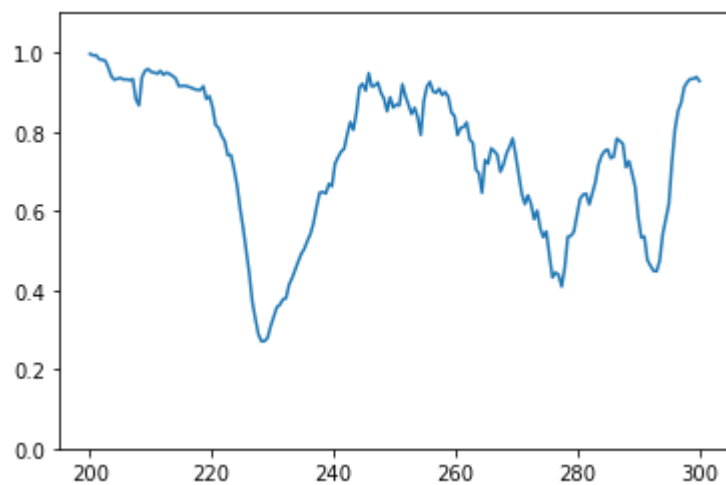


Test 2

True spectrum:

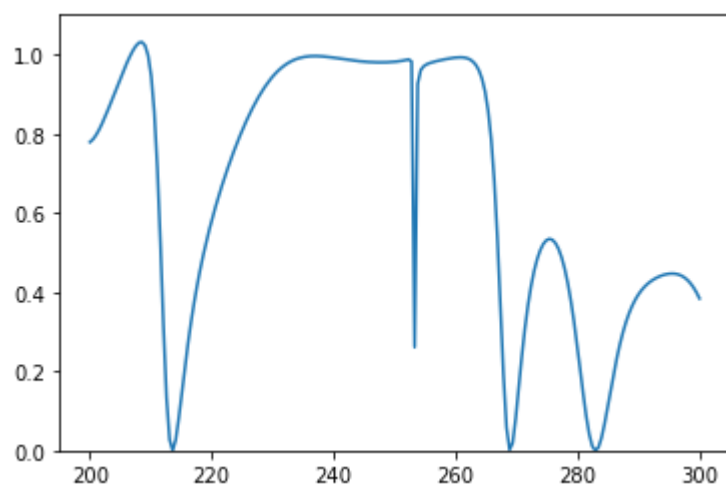


Predicted spectrum:

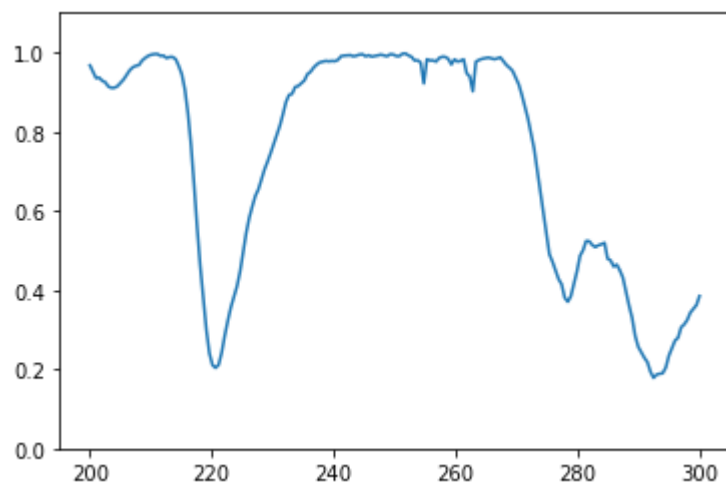


Test 3

True spectrum:

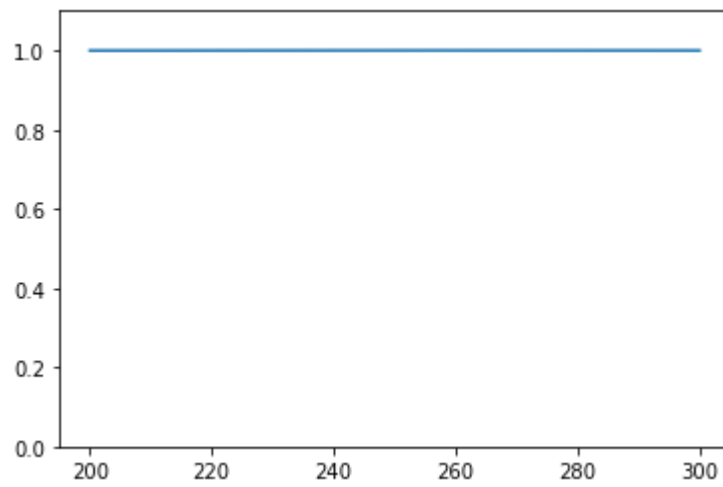


Predicted spectrum:

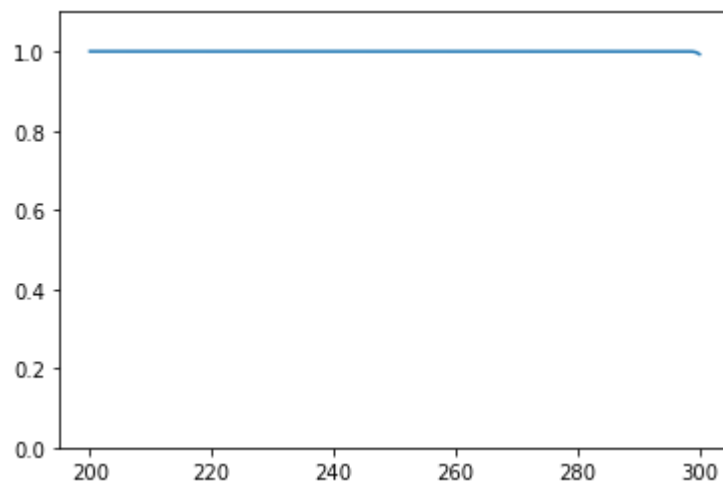


Test 4

True spectrum:

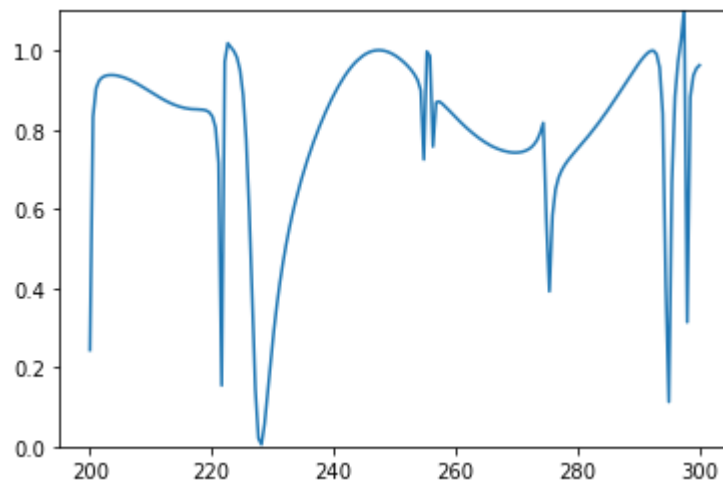


Predicted spectrum:

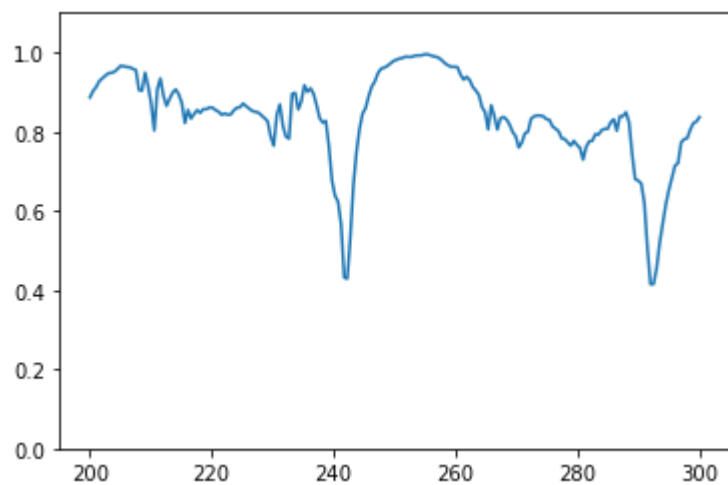


Test 5

True spectrum:

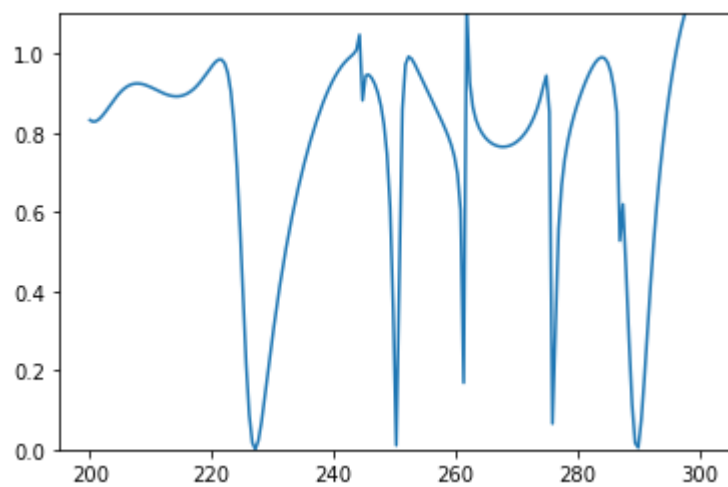


Predicted spectrum:

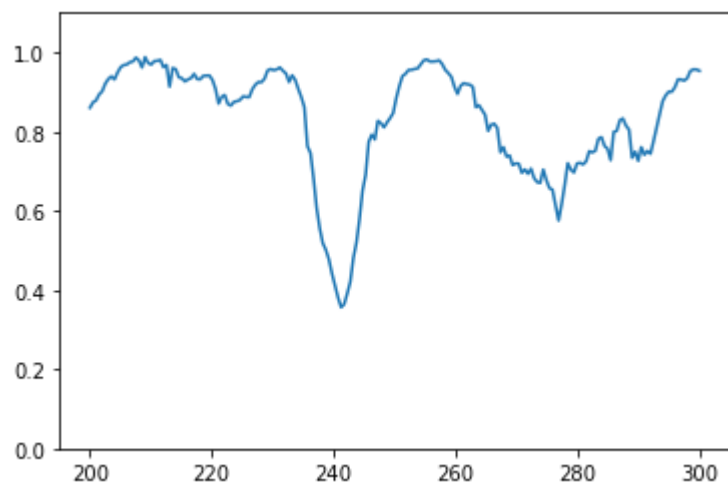


Test 6

True spectrum:

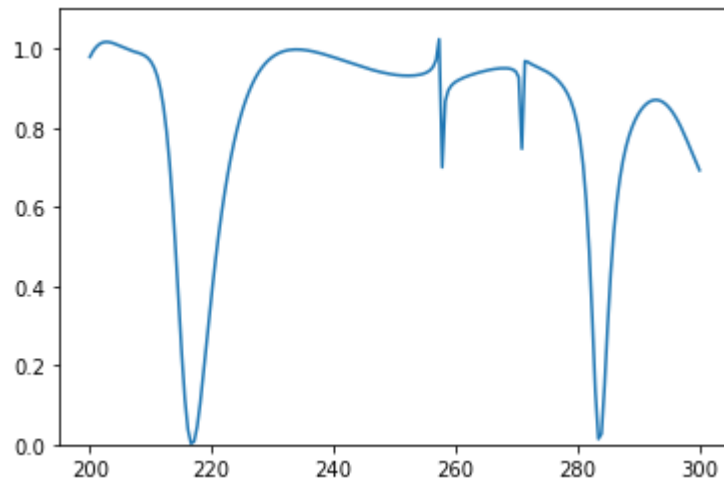


Predicted spectrum:

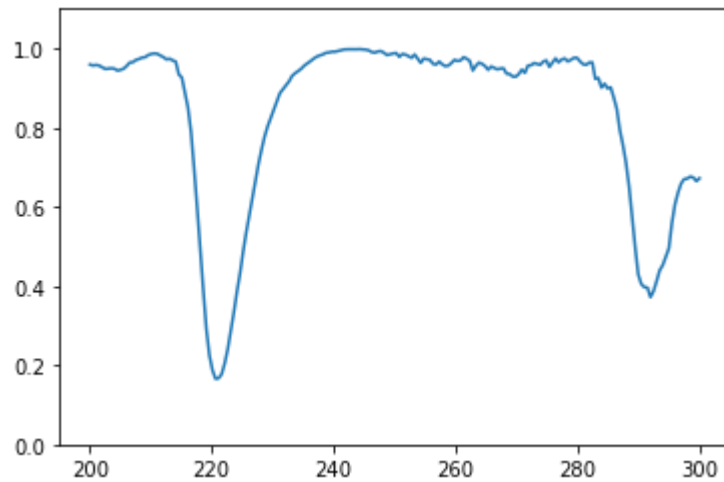


Test 7

True spectrum:

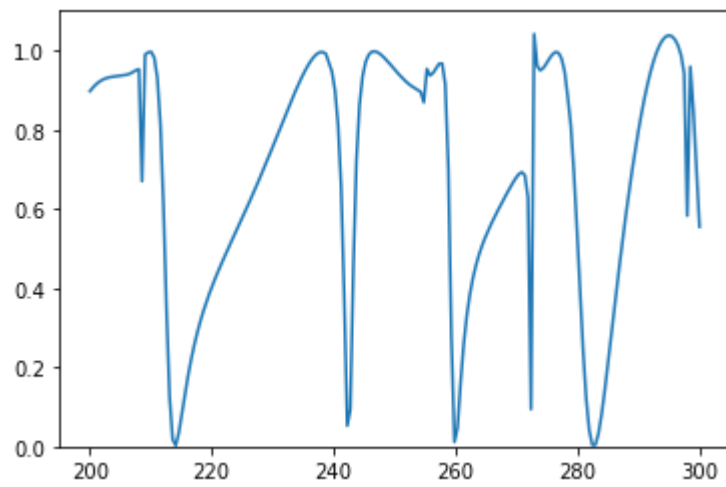


Predicted spectrum:

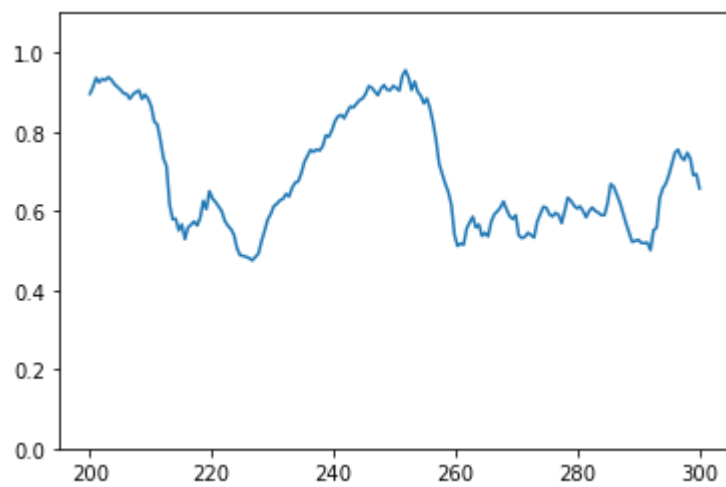


Test 8

True spectrum:

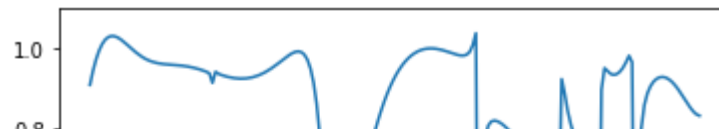


Predicted spectrum:

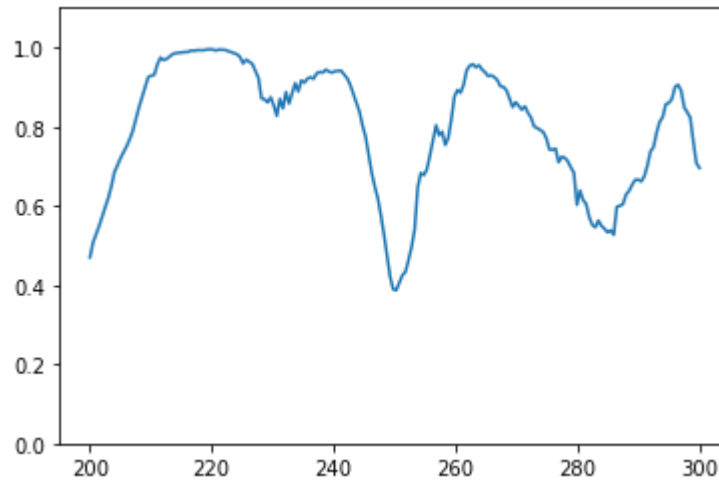


Test 9

True spectrum:

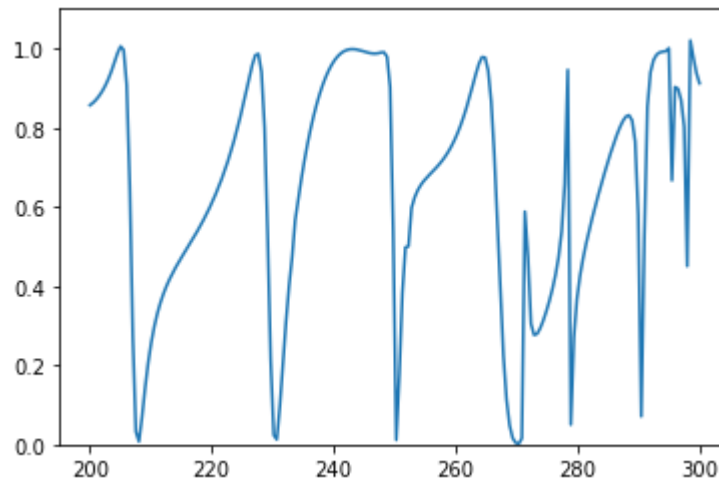


Predicted spectrum:

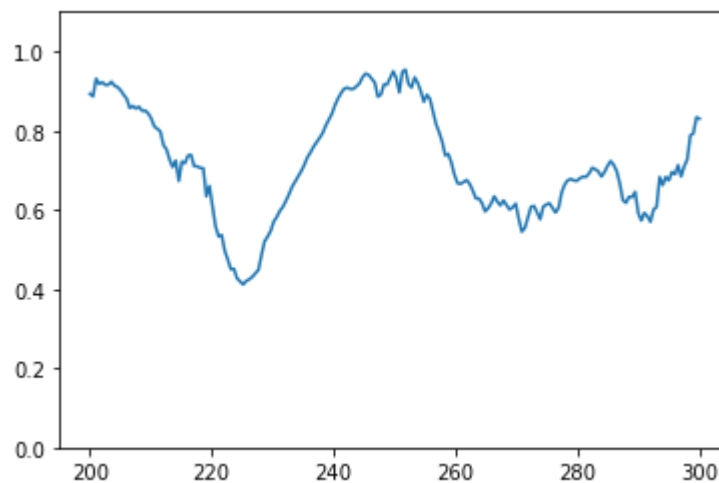


Test 10

True spectrum:

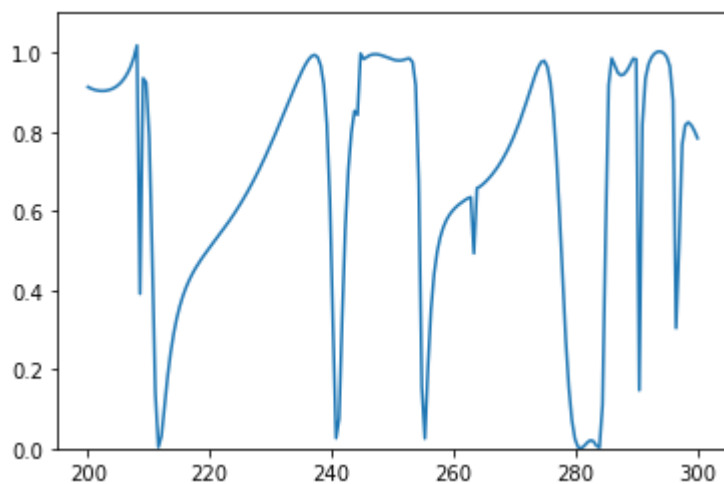


Predicted spectrum:

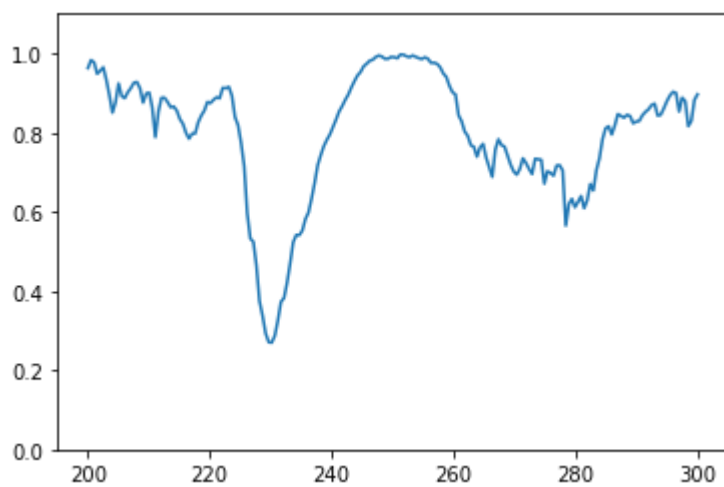


Test 11

True spectrum:

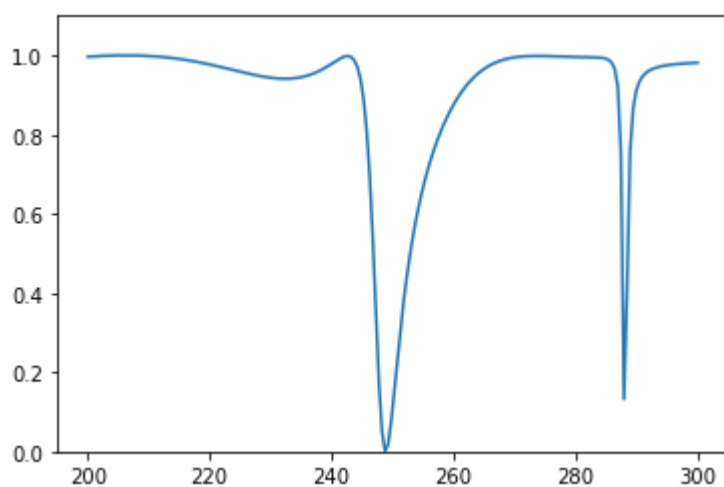


Predicted spectrum:

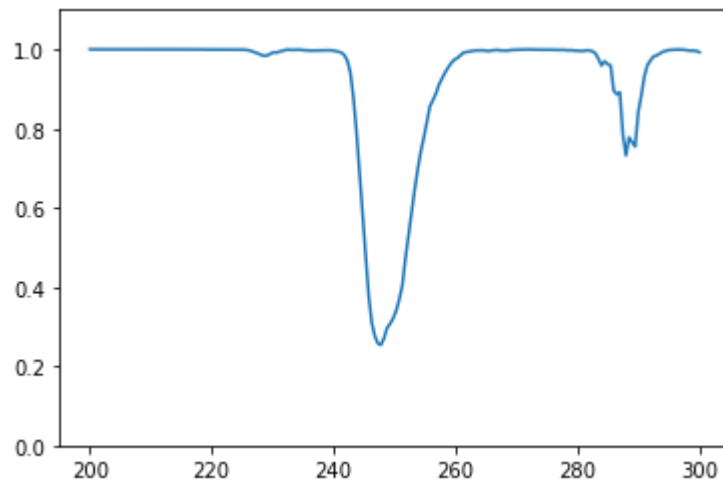


Test 12

True spectrum:

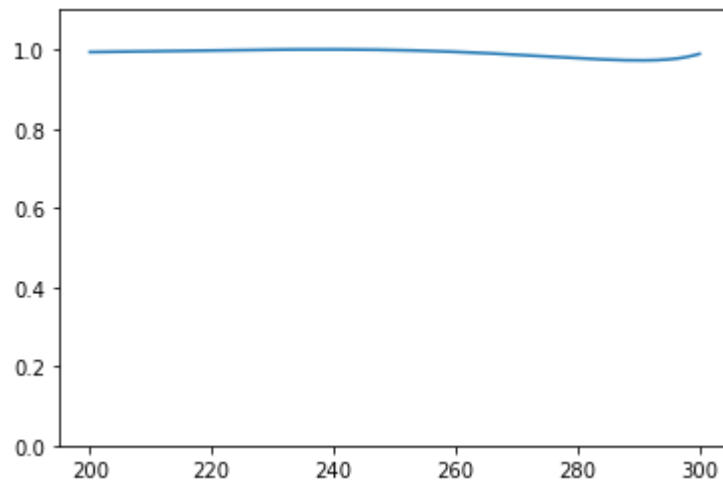


Predicted spectrum:

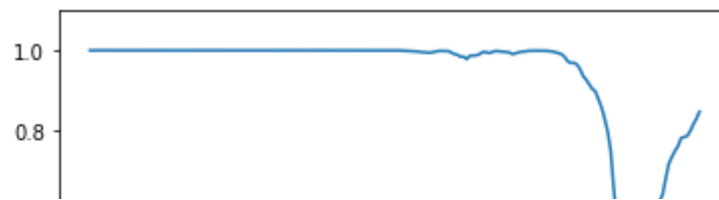


Test 13

True spectrum:

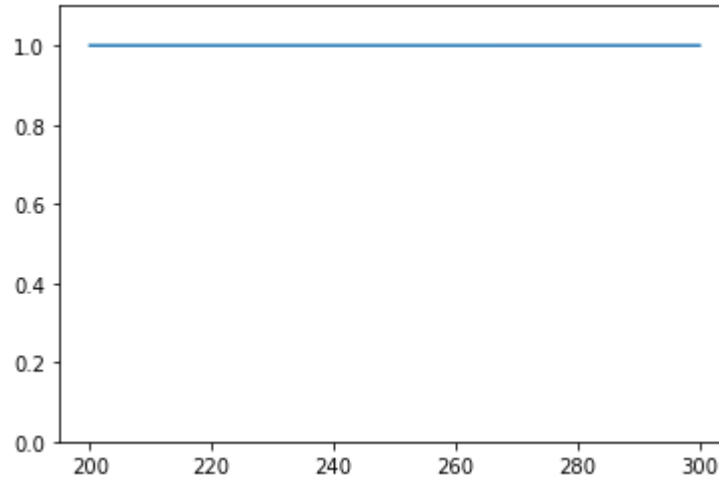


Predicted spectrum:

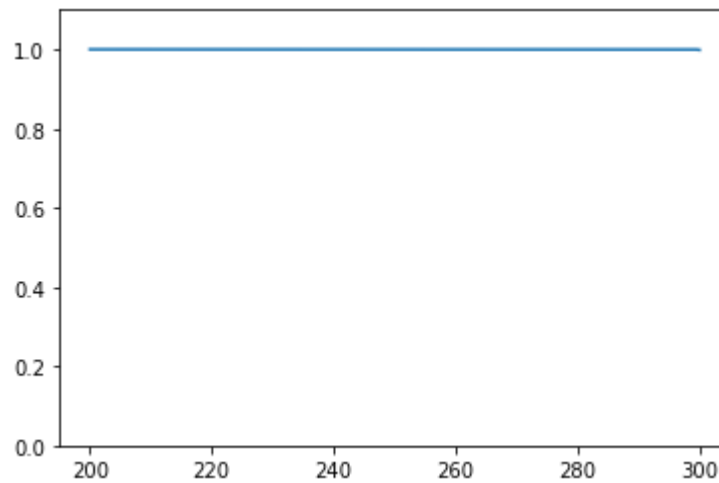


Test 14

True spectrum:

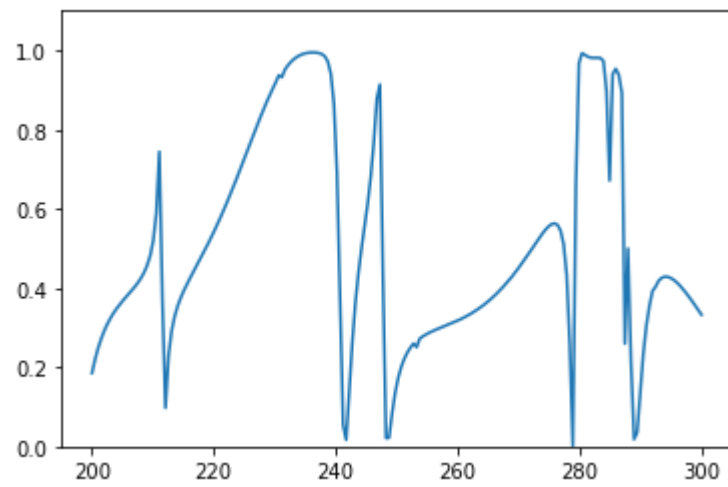


Predicted spectrum:

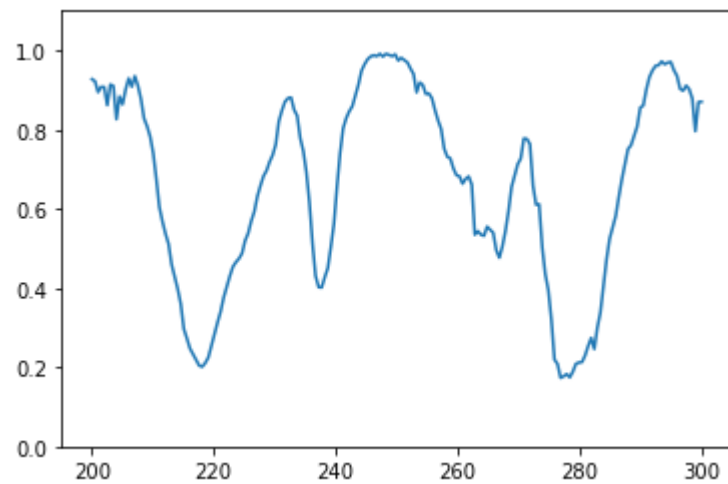


Test 15

True spectrum:

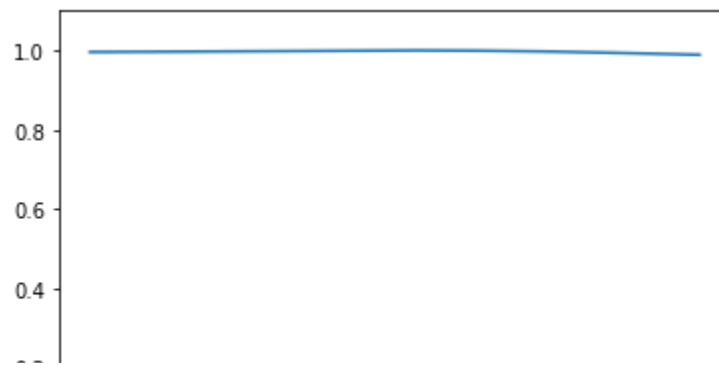


Predicted spectrum:

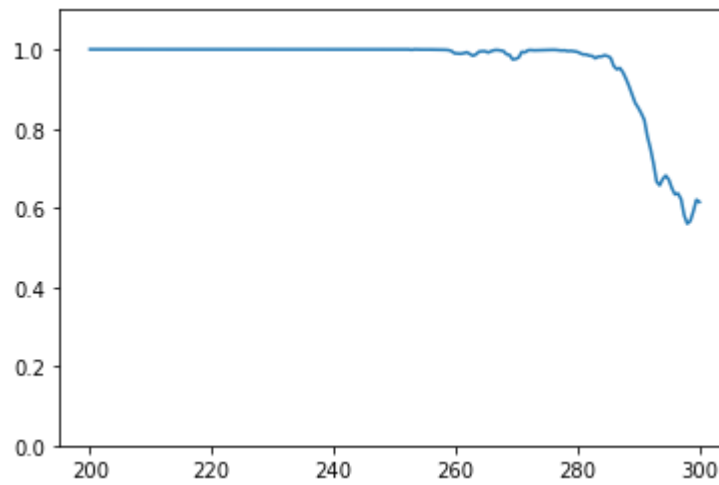


Test 16

True spectrum:

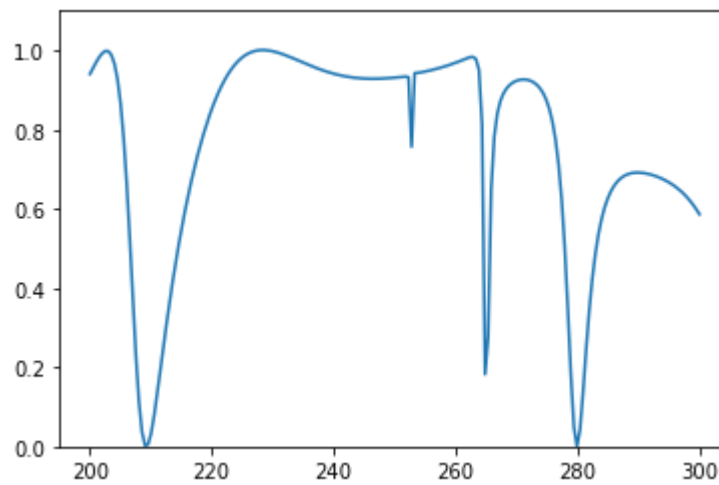


Predicted spectrum:

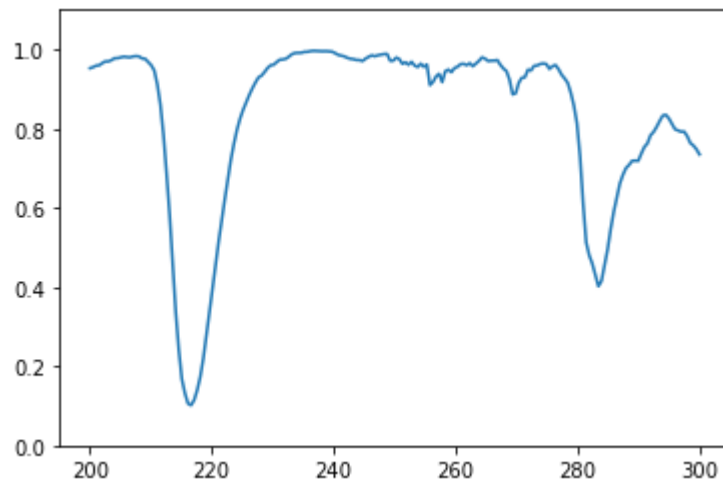


Test 17

True spectrum:

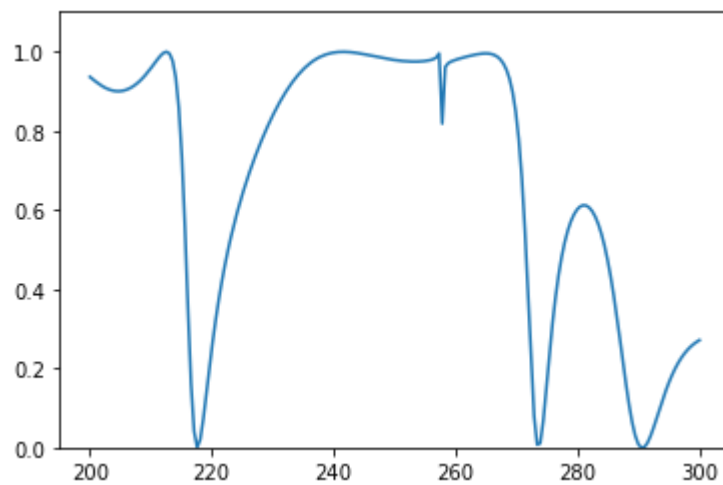


Predicted spectrum:

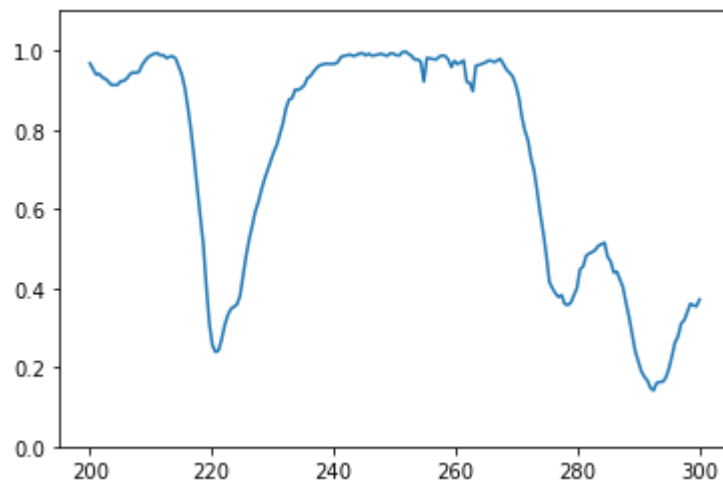


Test 18

True spectrum:

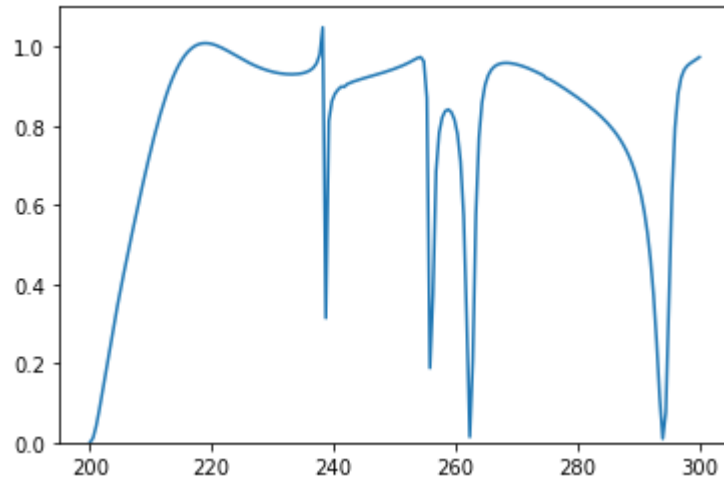


Predicted spectrum:

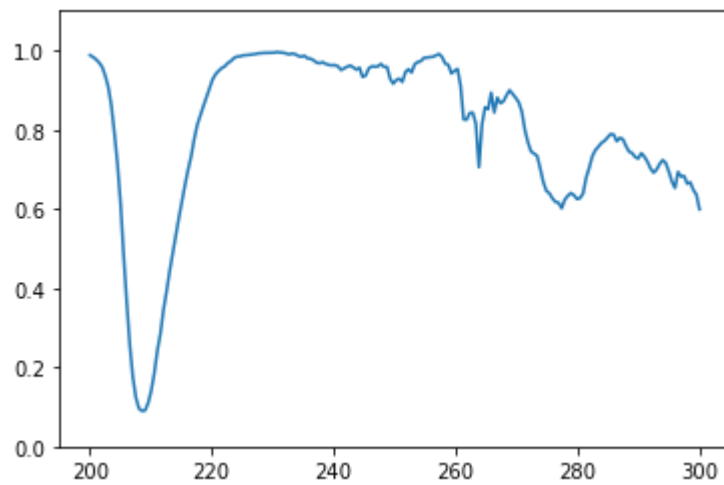


Test 19

True spectrum:

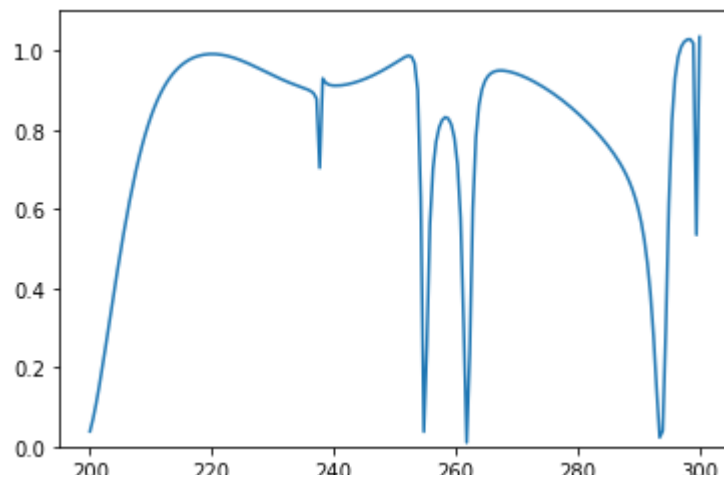


Predicted spectrum:

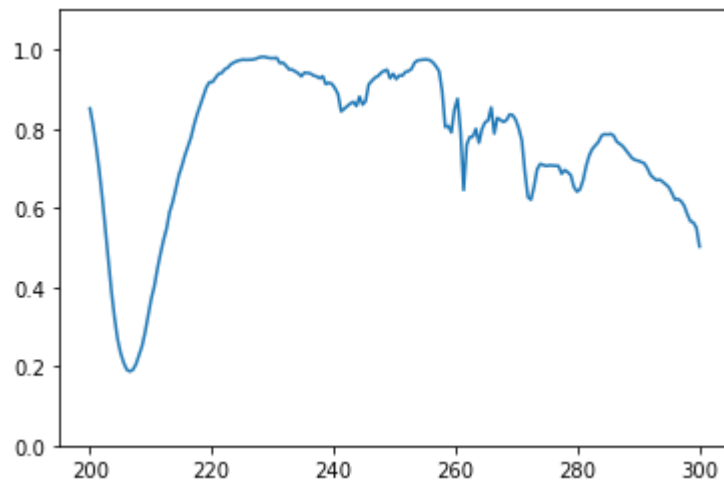


Test 20

True spectrum:

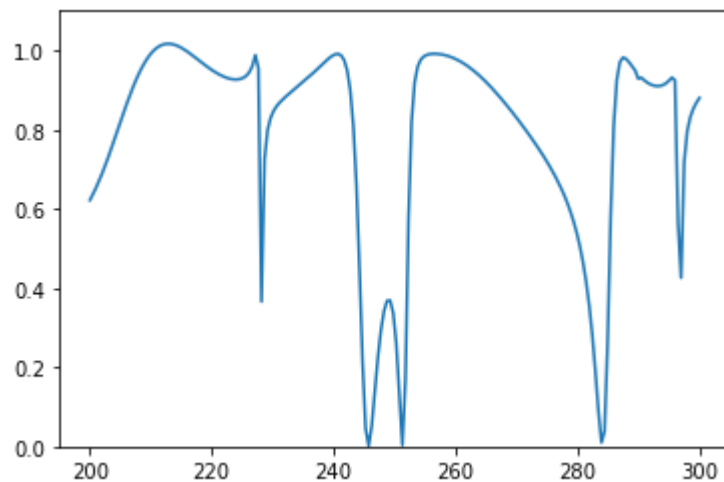


Predicted spectrum:

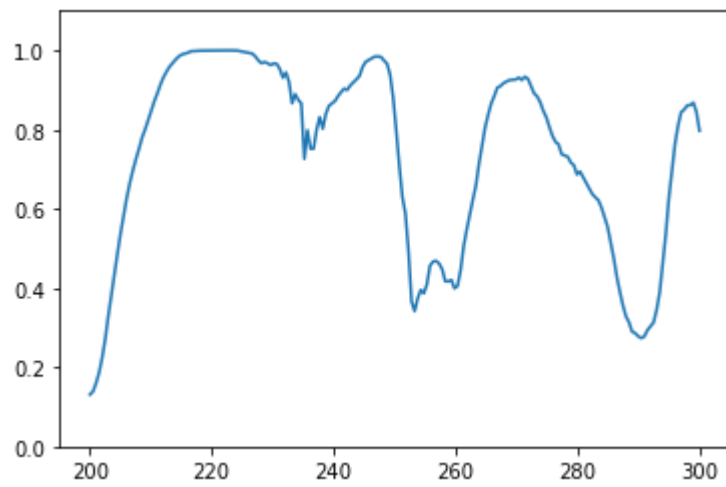


Test 21

True spectrum:

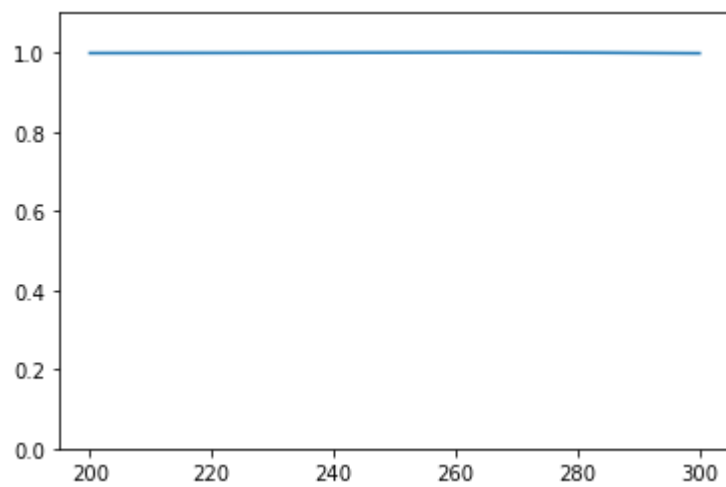


Predicted spectrum:

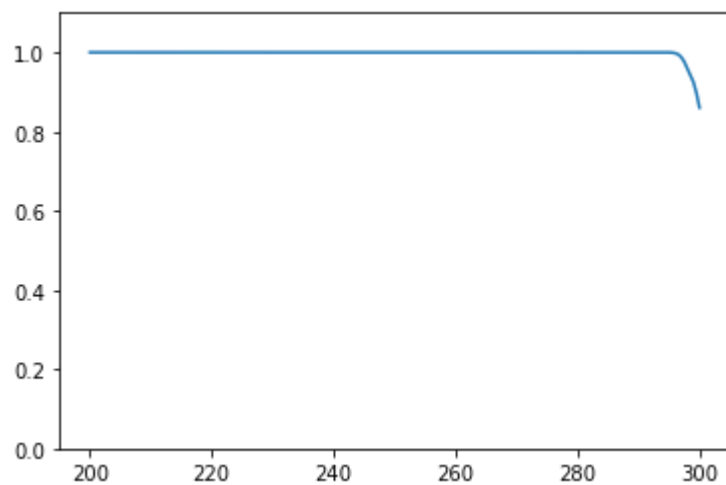


Test 22

True spectrum:

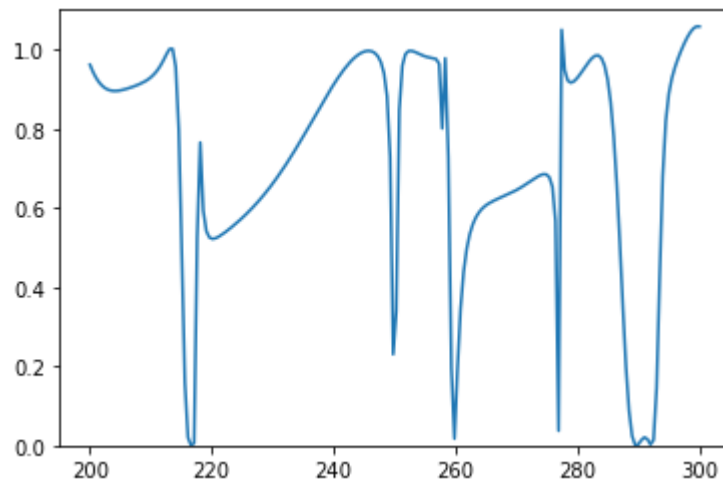


Predicted spectrum:

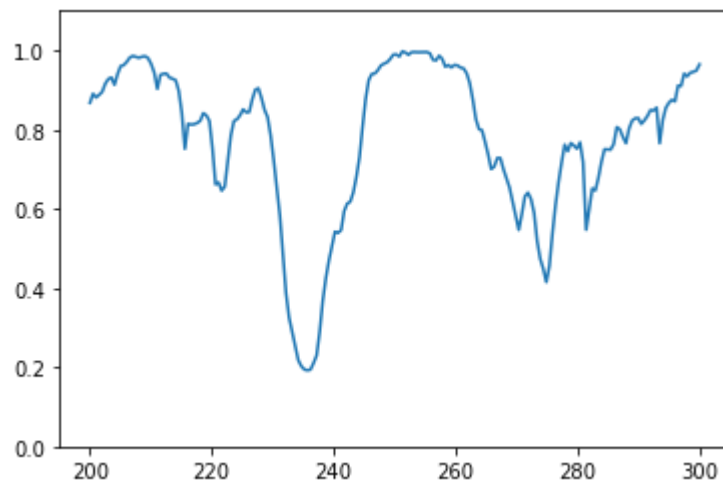


Test 23

True spectrum:

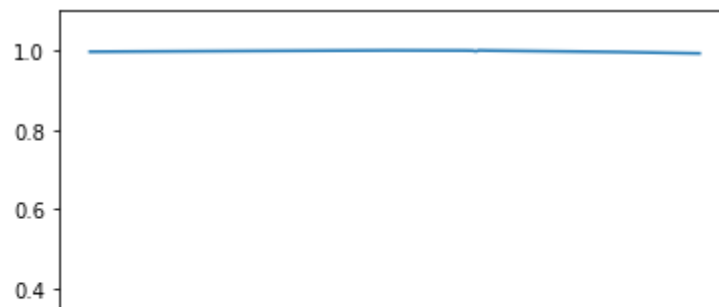


Predicted spectrum:

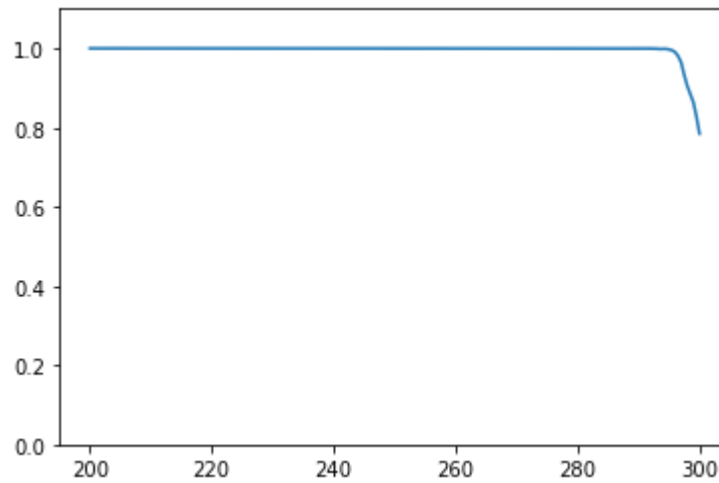


Test 24

True spectrum:

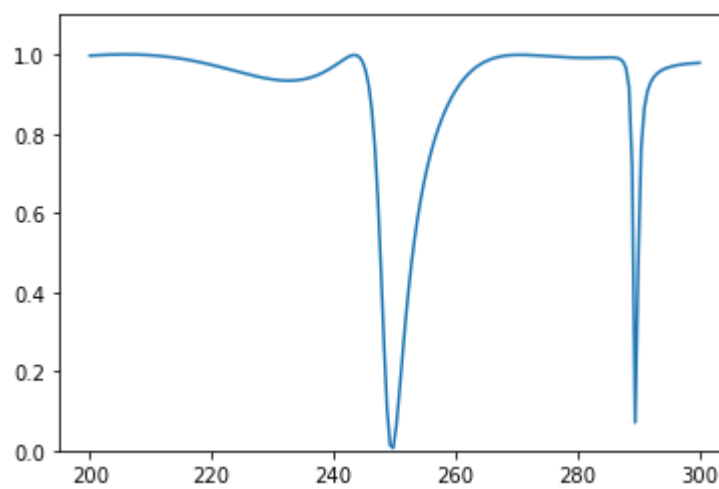


Predicted spectrum:

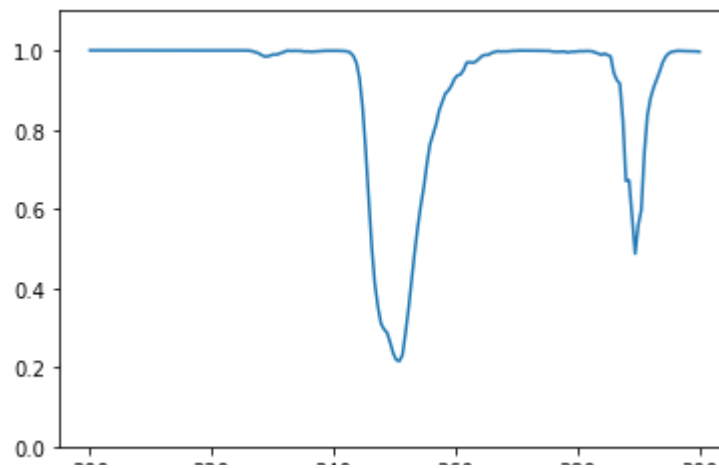


Test 25

True spectrum:

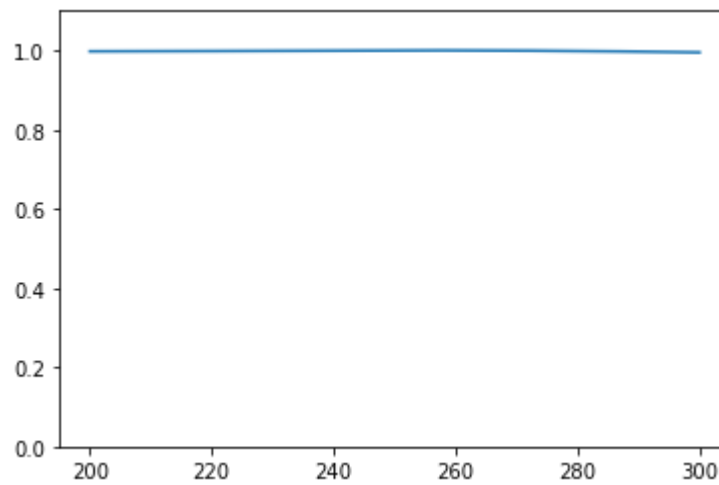


Predicted spectrum:

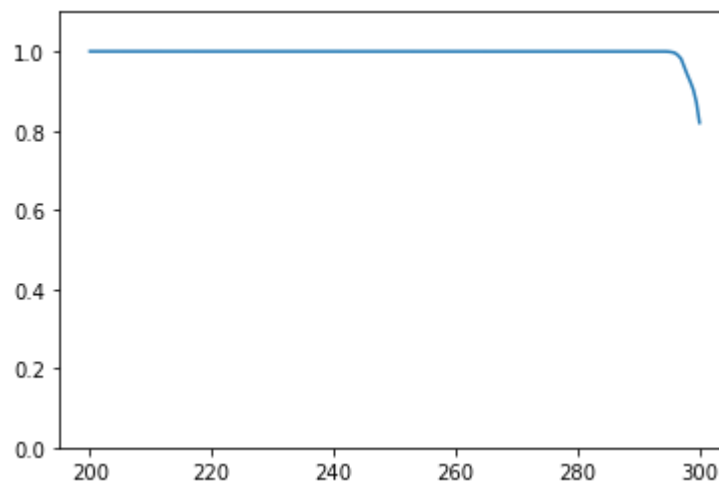


Test 26

True spectrum:

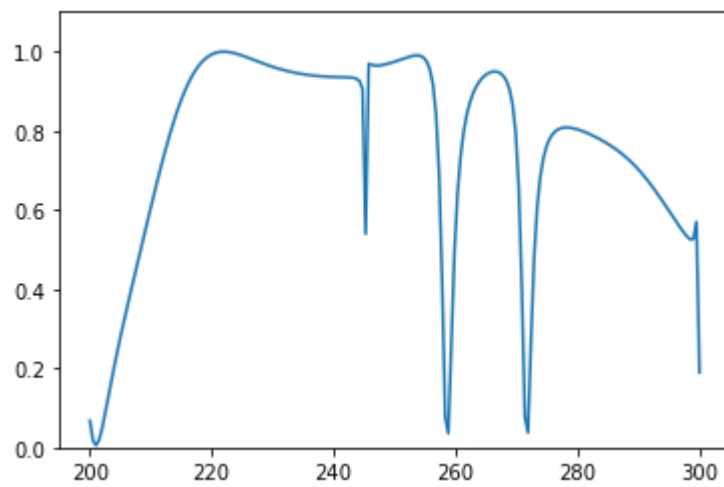


Predicted spectrum:

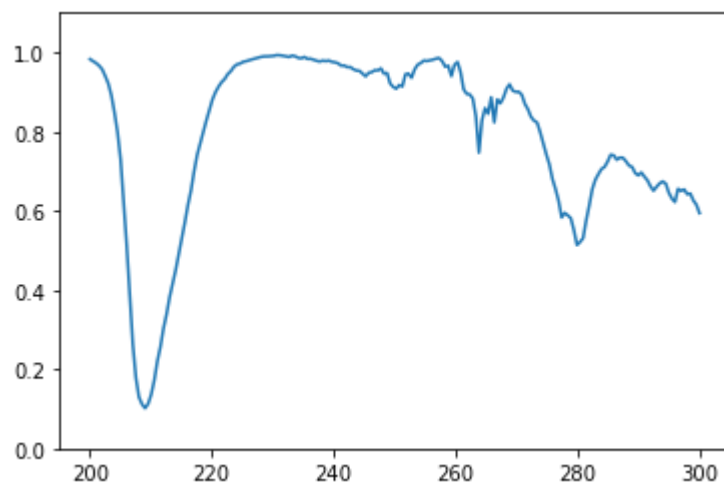


Test 27

True spectrum:



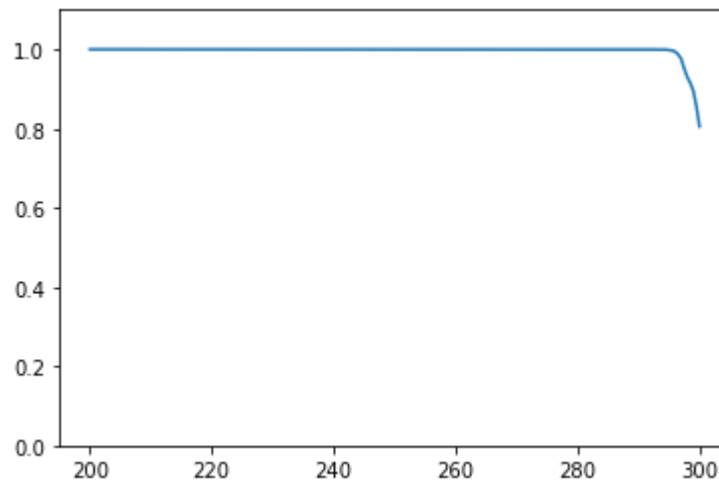
Predicted spectrum:



Test 28
True spectrum:

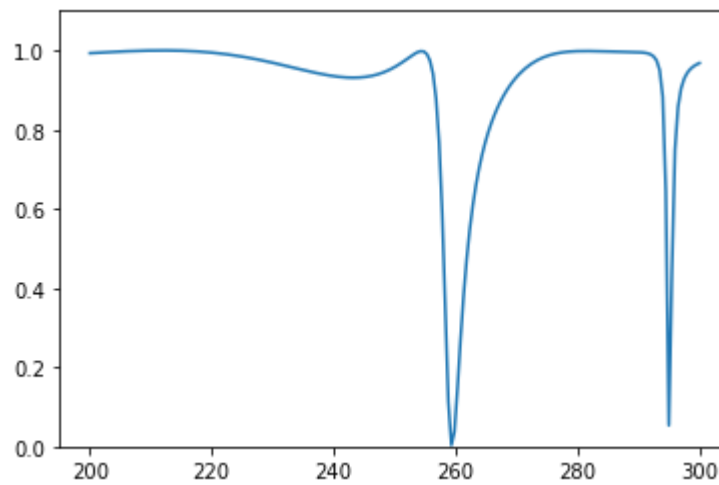


Predicted spectrum:

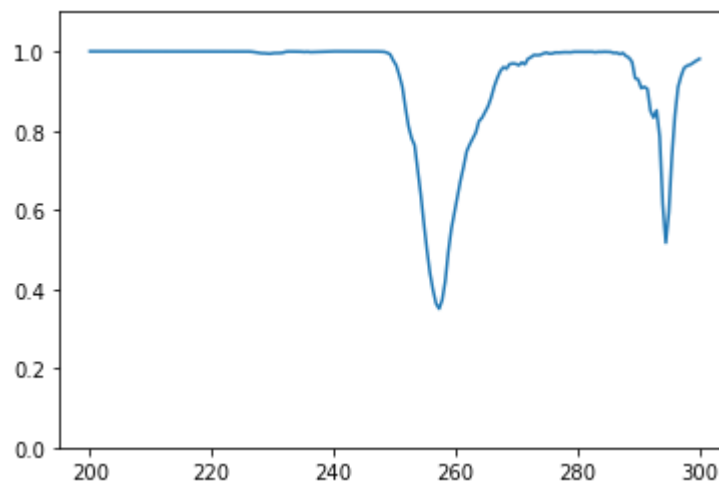


Test 29

True spectrum:

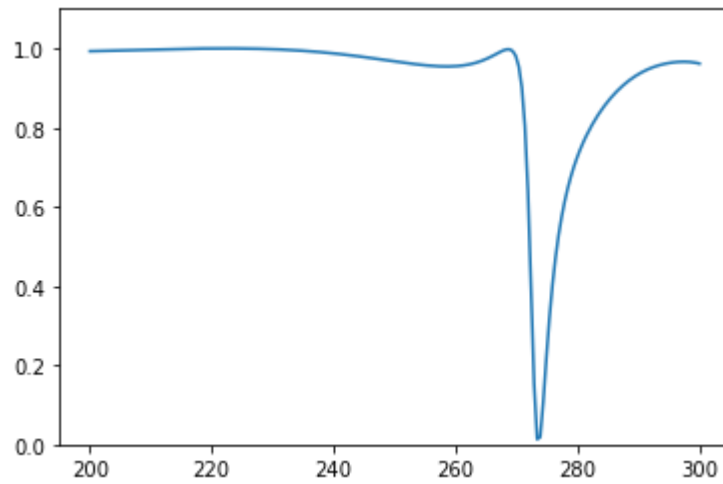


Predicted spectrum:

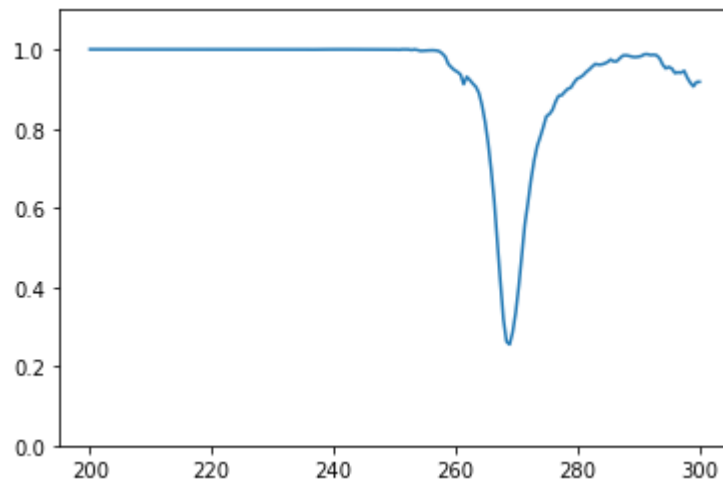


Test 30

True spectrum:

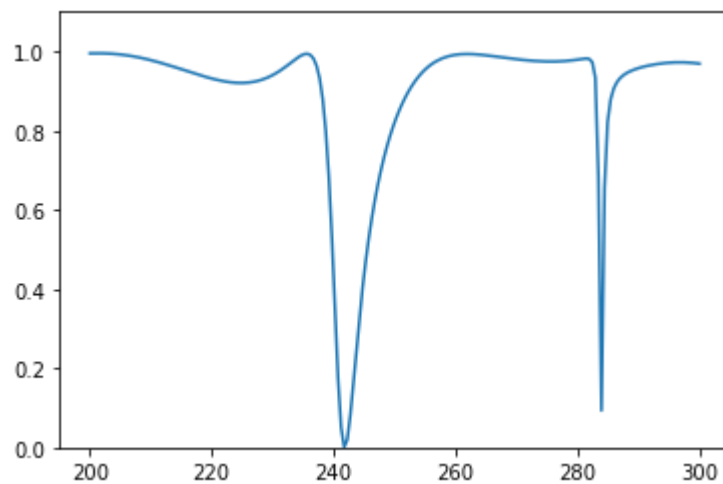


Predicted spectrum:

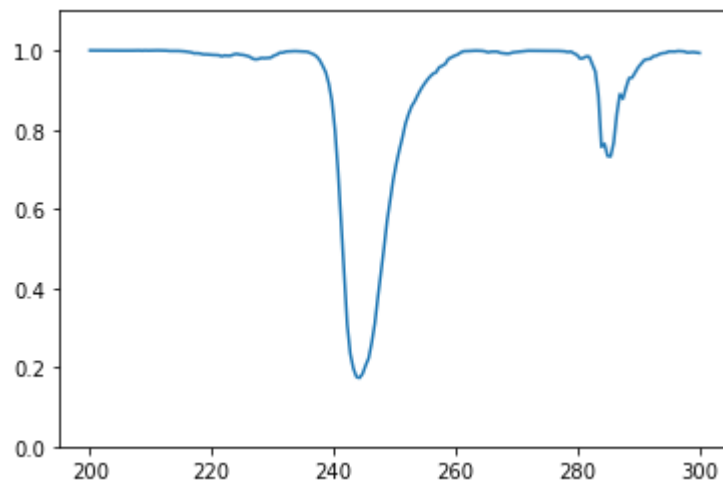


Test 31

True spectrum:

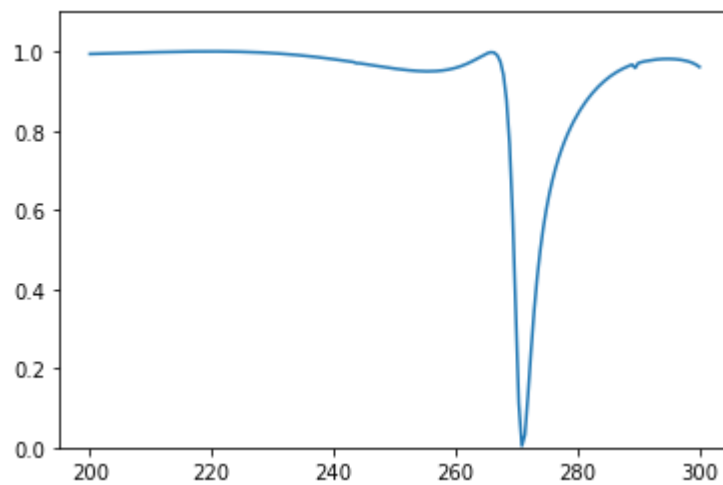


Predicted spectrum:

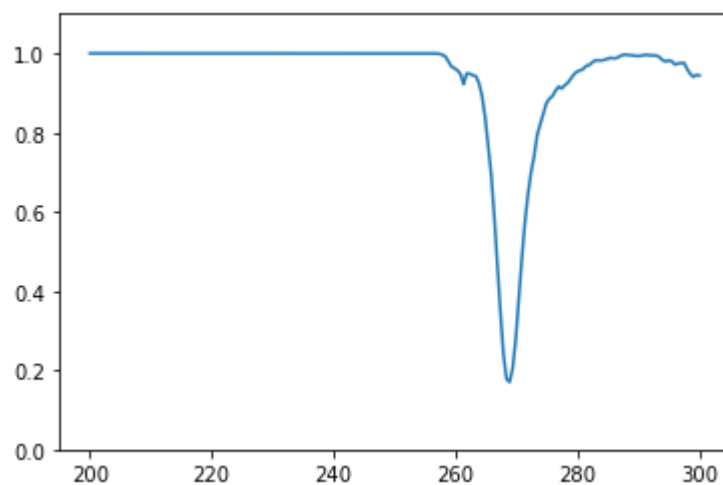


Test 32

True spectrum:

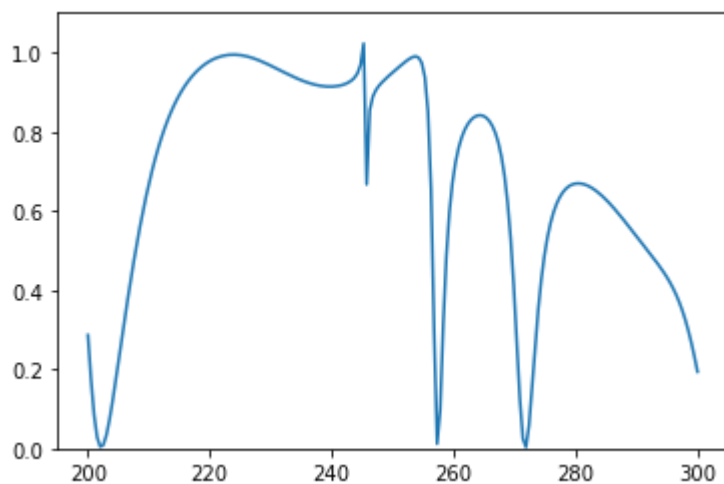


Predicted spectrum:

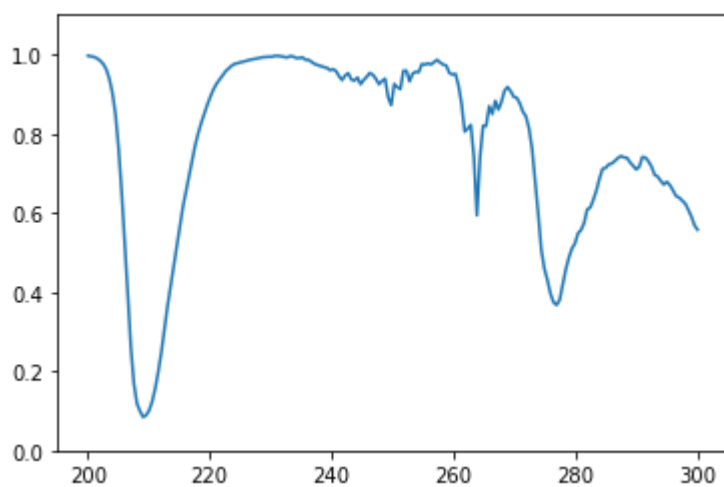


Test 33

True spectrum:

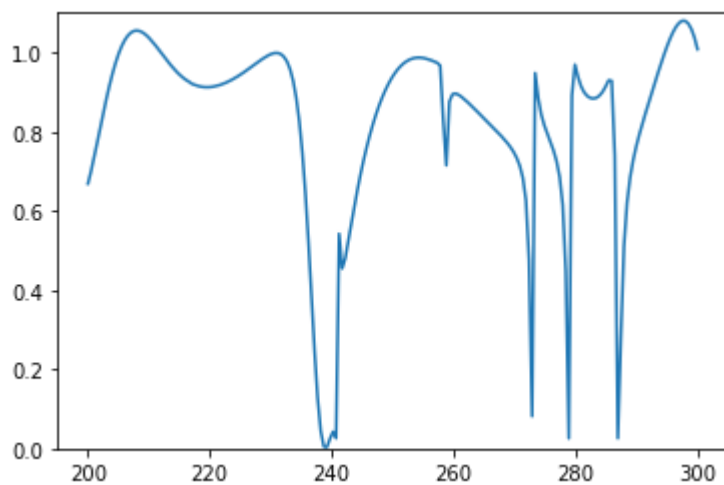


Predicted spectrum:

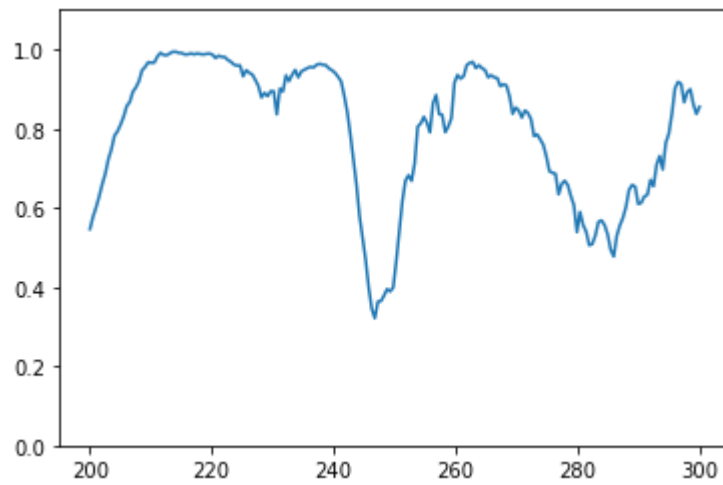


Test 34

True spectrum:

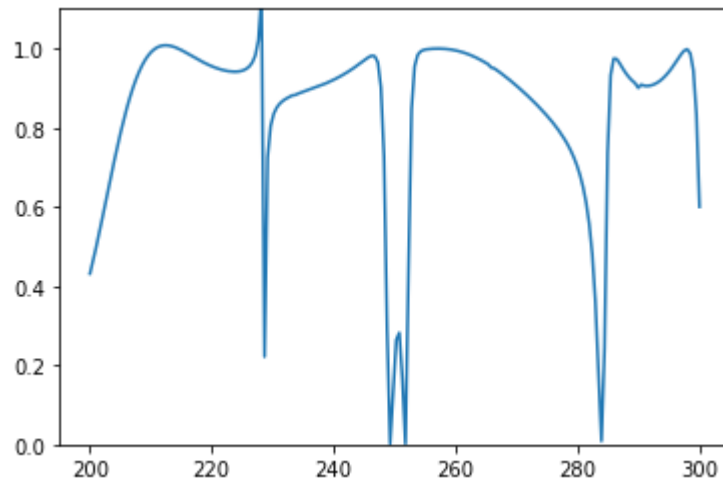


Predicted spectrum:

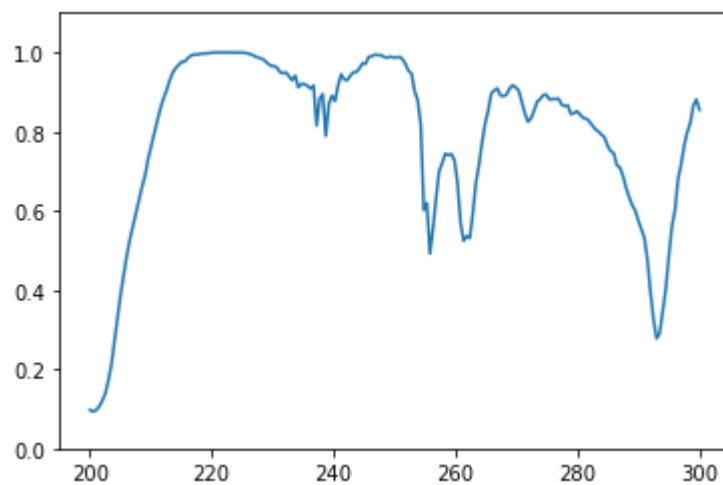


Test 35

True spectrum:

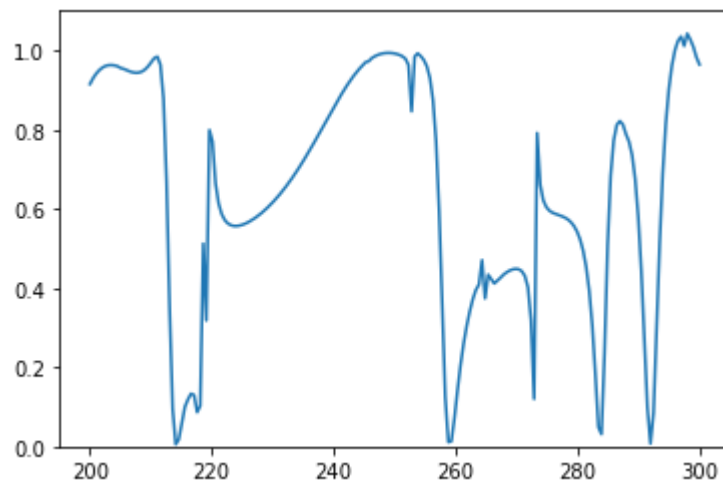


Predicted spectrum:

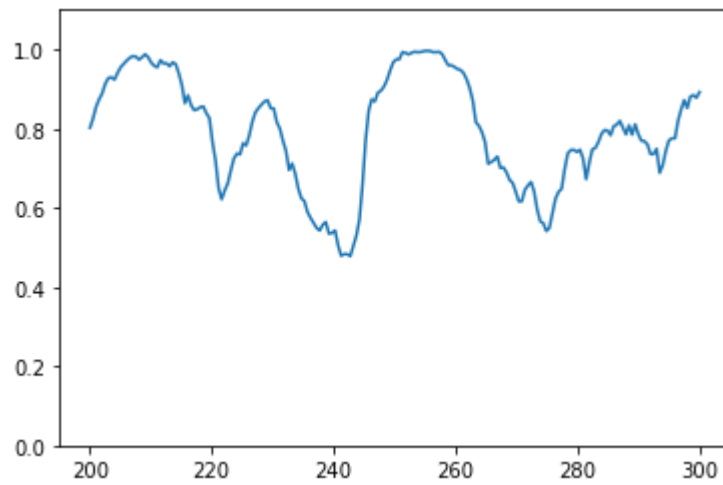


Test 36

True spectrum:

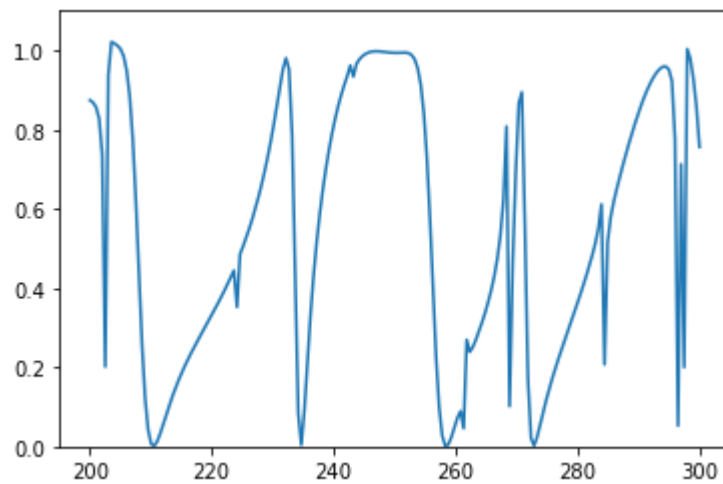


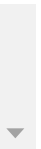
Predicted spectrum:



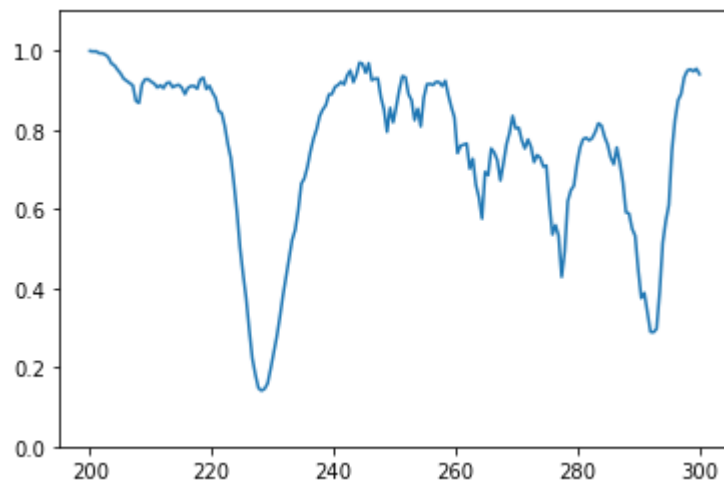
Test 37

True spectrum:



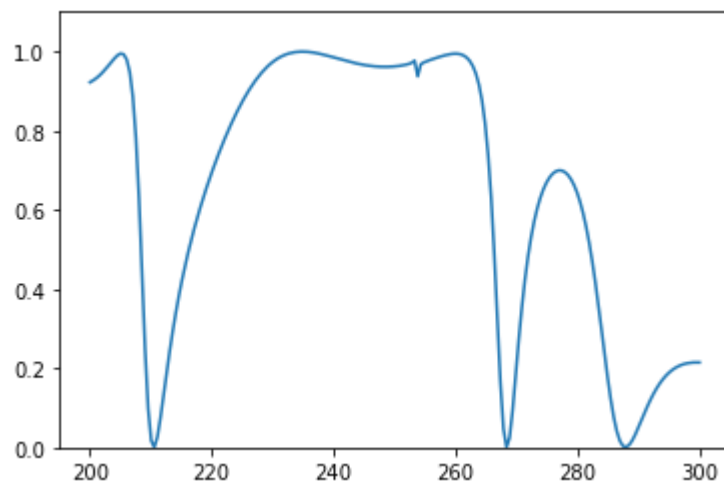


Predicted spectrum:

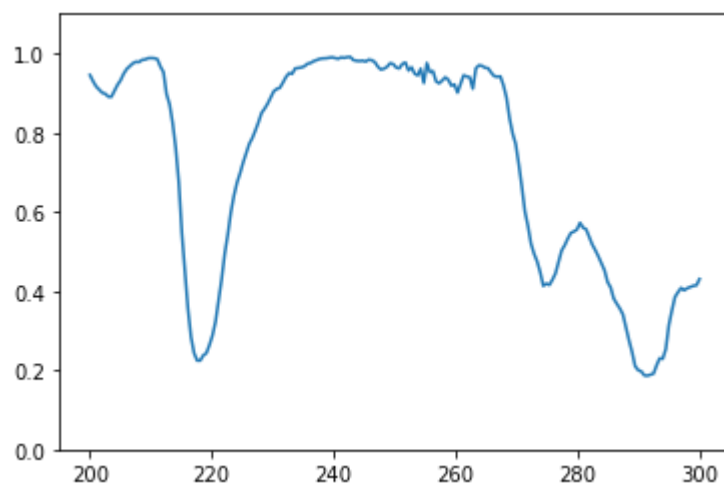


Test 38

True spectrum:

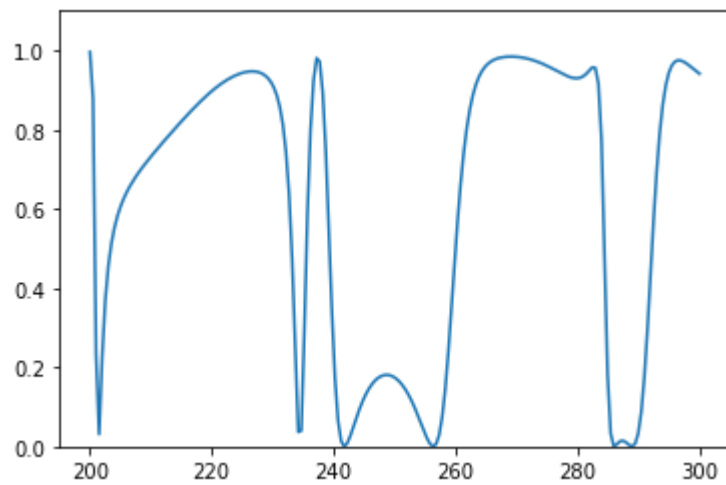


Predicted spectrum:

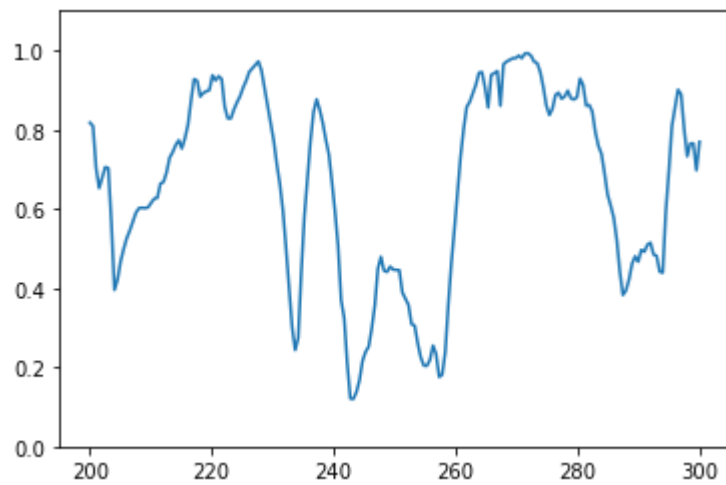


Test 39

True spectrum:

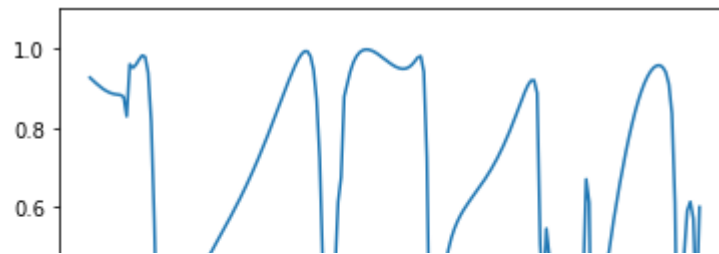


Predicted spectrum:

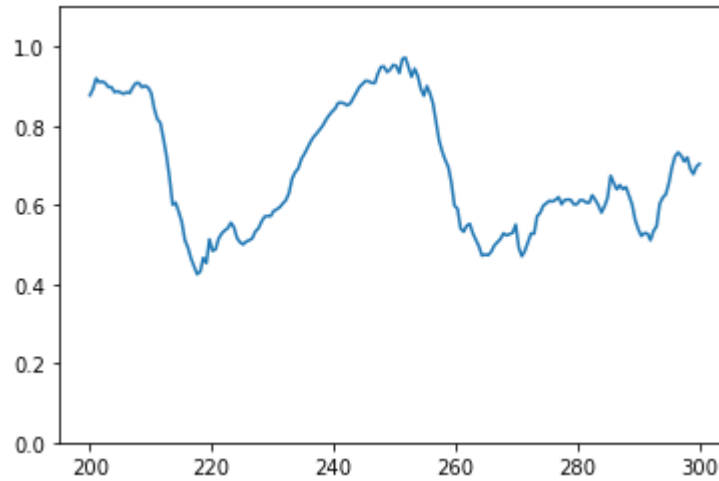


Test 40

True spectrum:

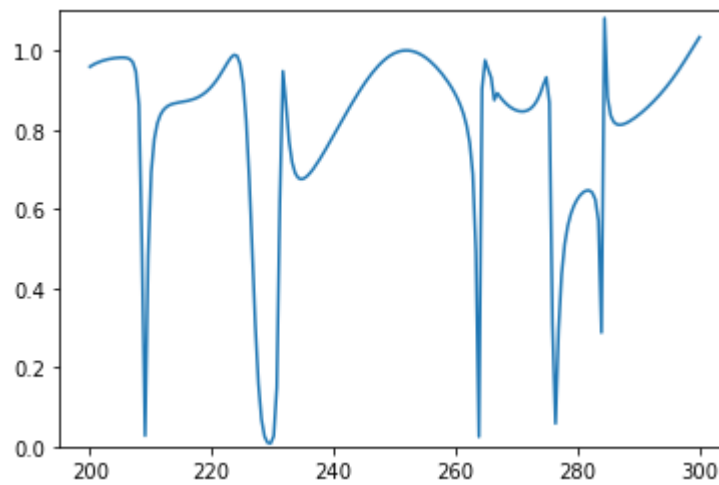


Predicted spectrum:

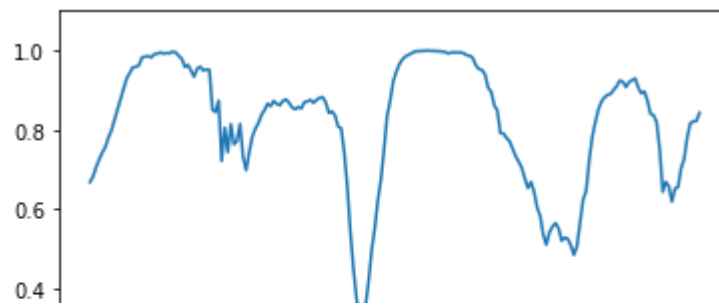


Test 41

True spectrum:

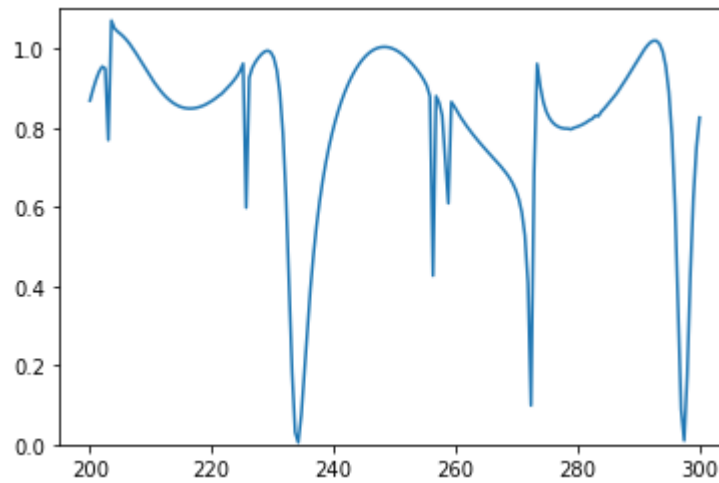


Predicted spectrum:

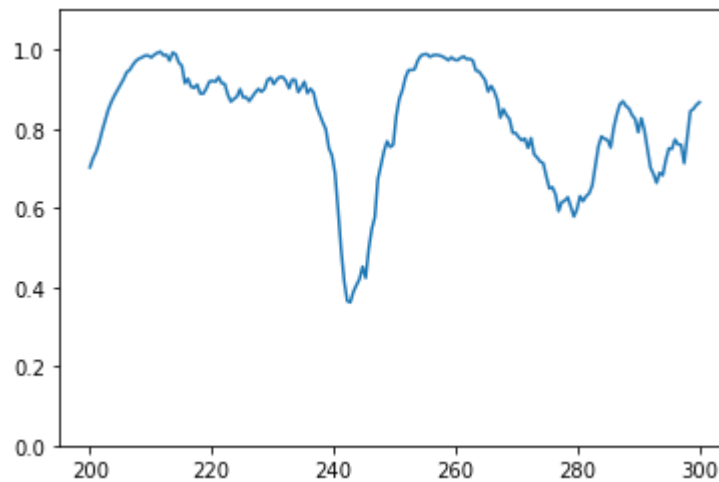


Test 42

True spectrum:

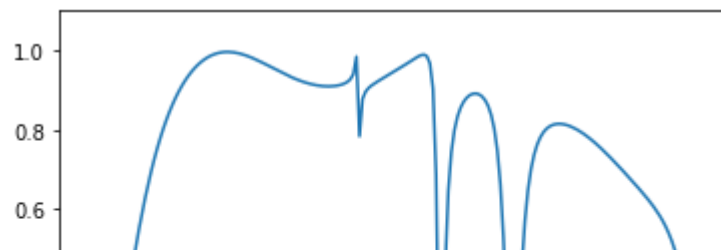


Predicted spectrum:

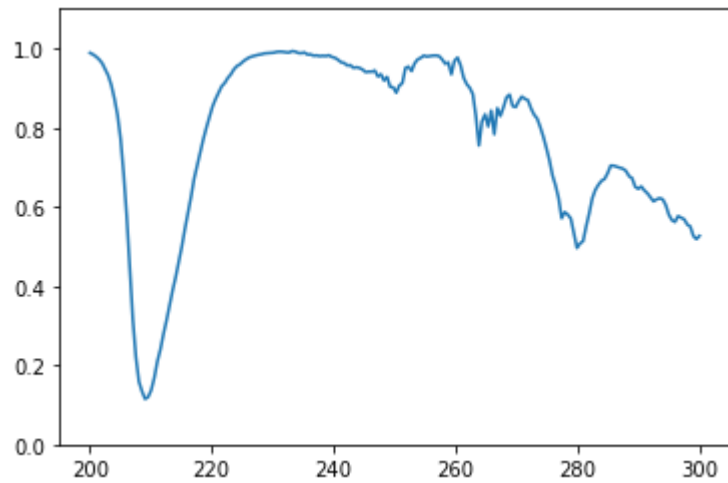


Test 43

True spectrum:

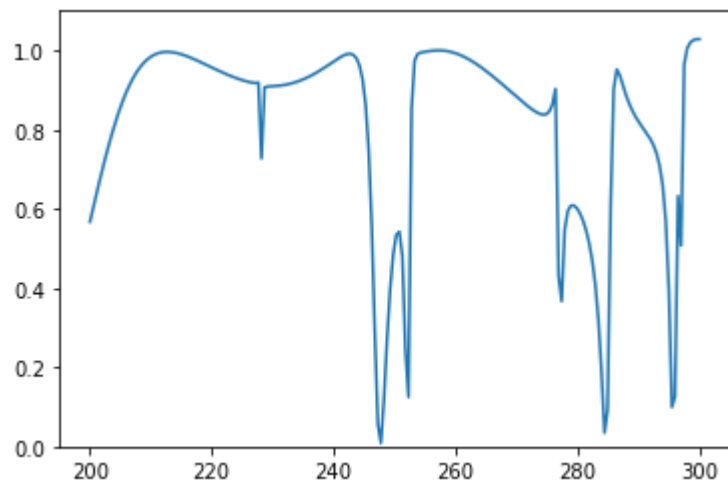


Predicted spectrum:

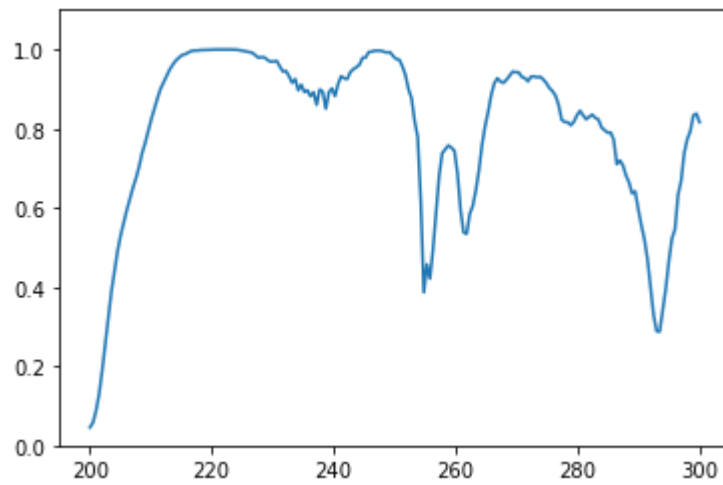


Test 44

True spectrum:

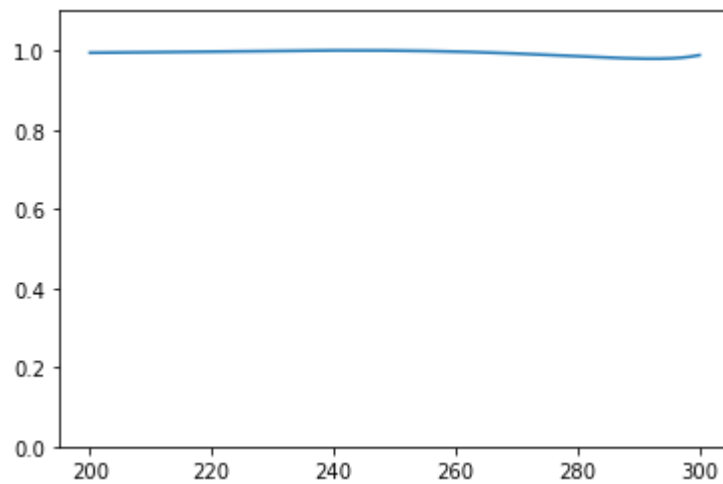


Predicted spectrum:

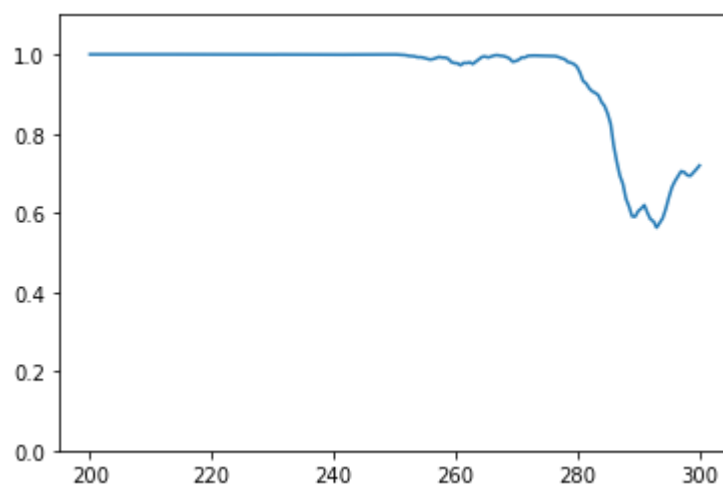


Test 45

True spectrum:

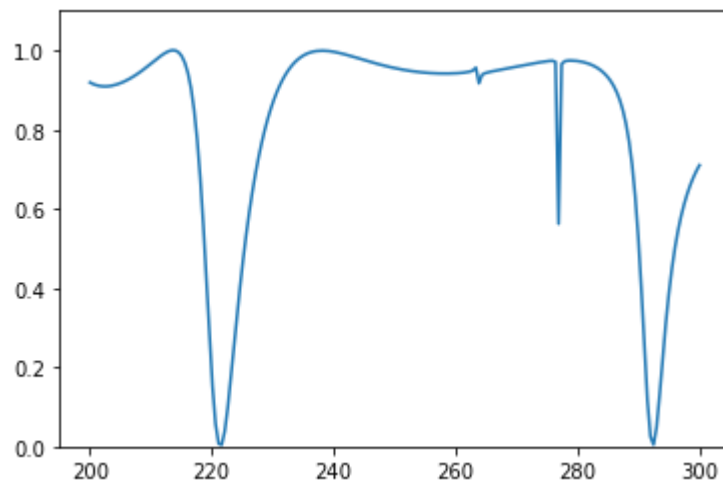


Predicted spectrum:

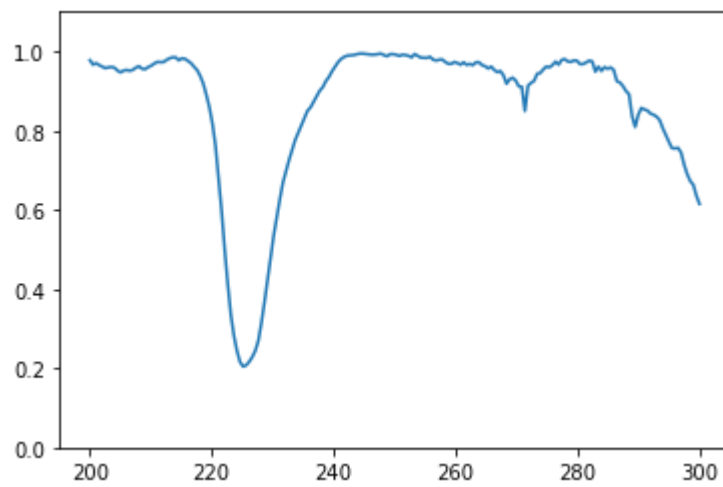


Test 46

True spectrum:

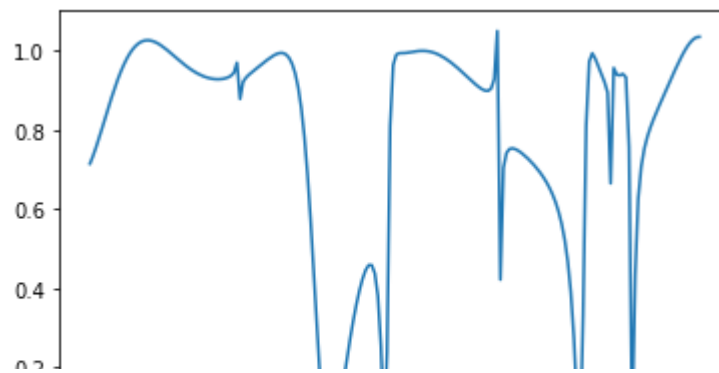


Predicted spectrum:

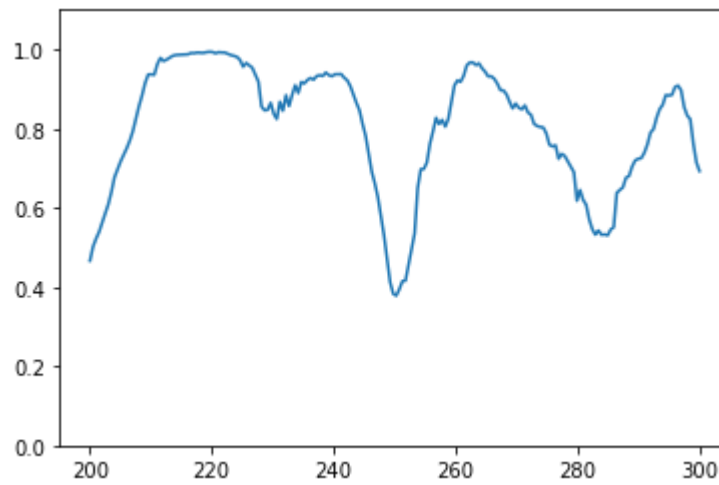


Test 47

True spectrum:

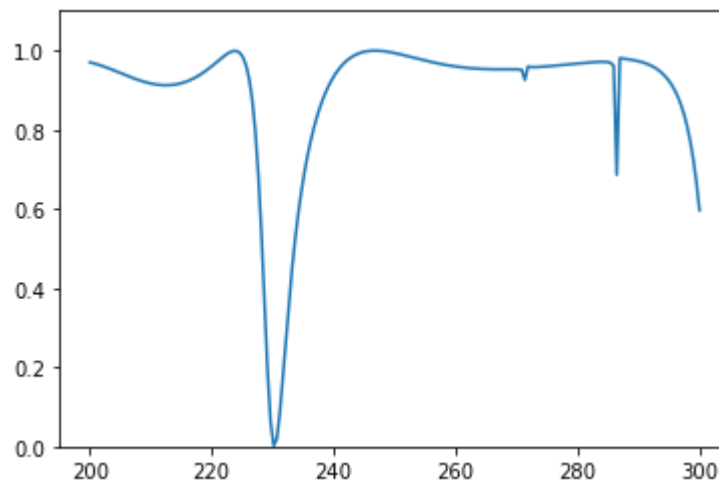


Predicted spectrum:

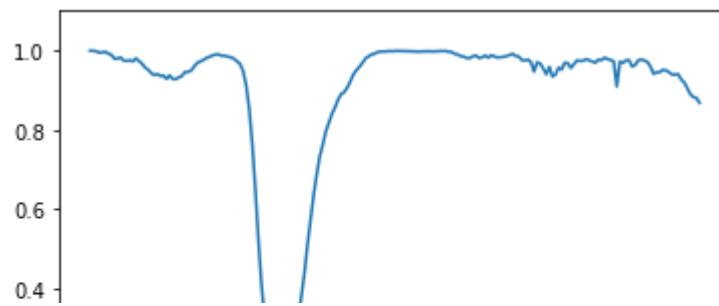


Test 48

True spectrum:

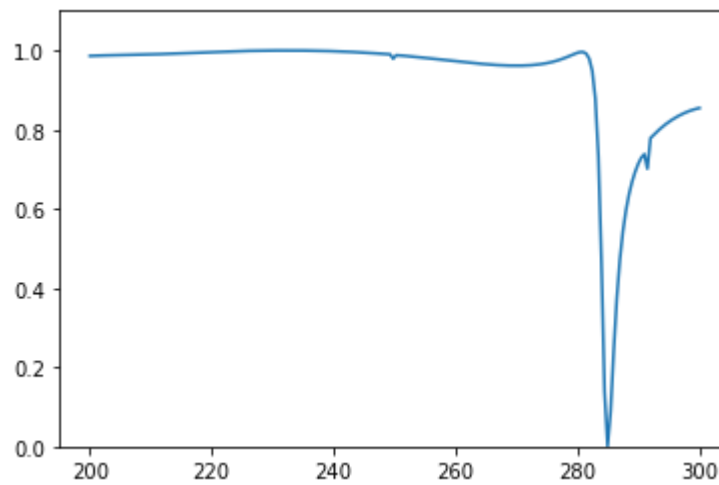


Predicted spectrum:

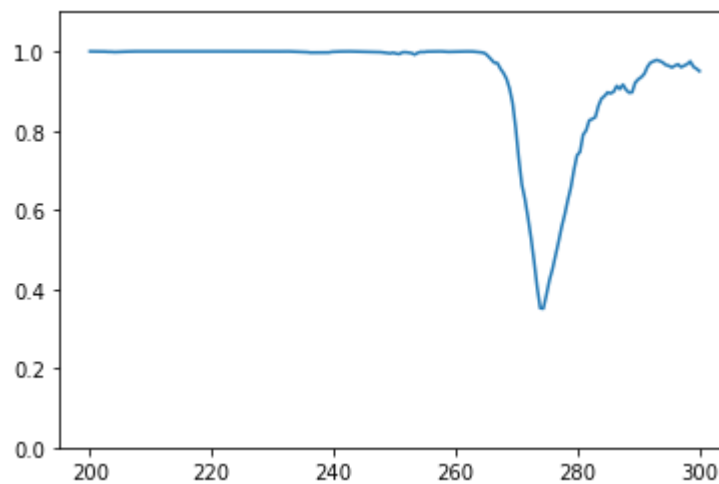


Test 49

True spectrum:



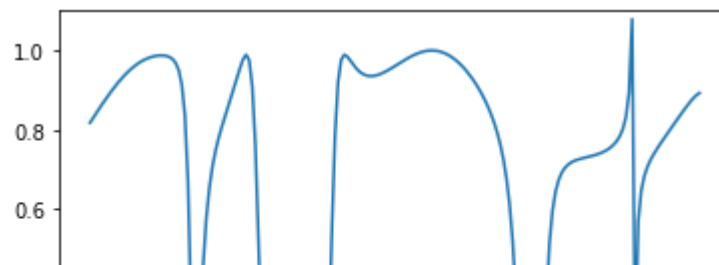
Predicted spectrum:



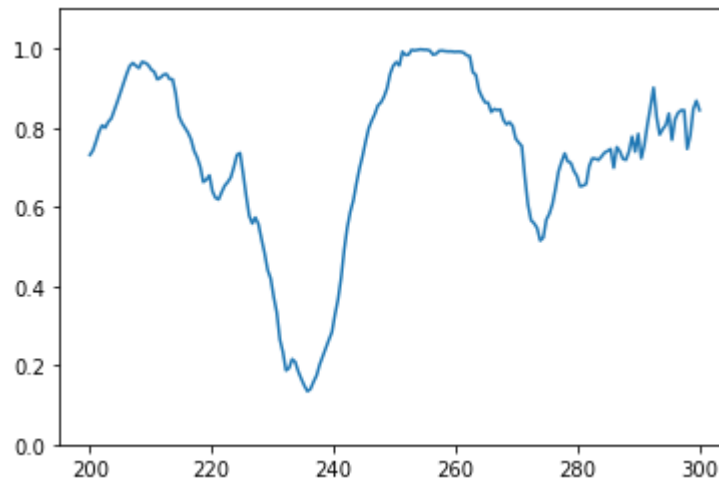
Test 50

True spectrum:



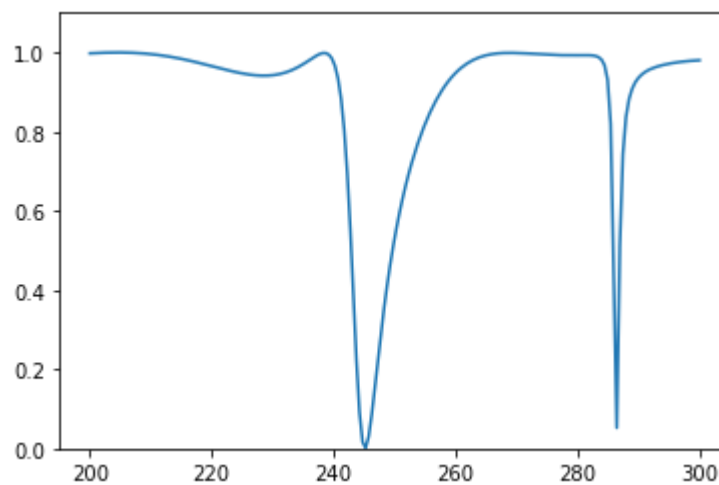


Predicted spectrum:

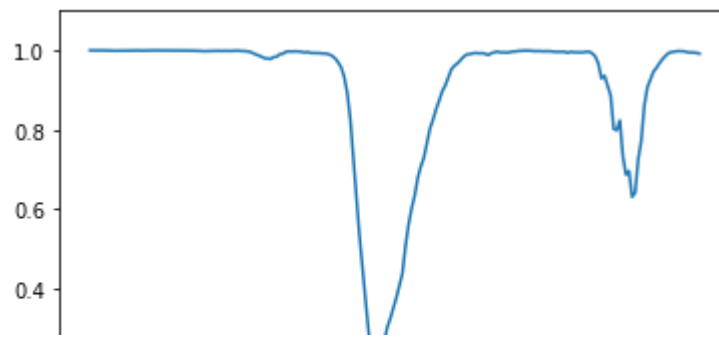


Test 51

True spectrum:

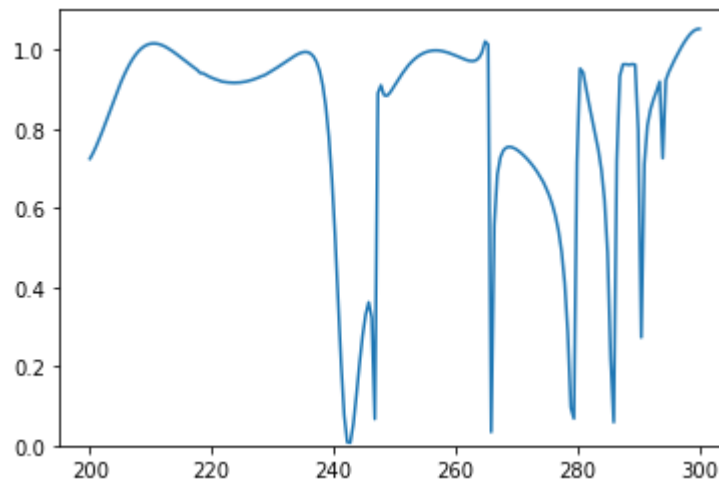


Predicted spectrum:

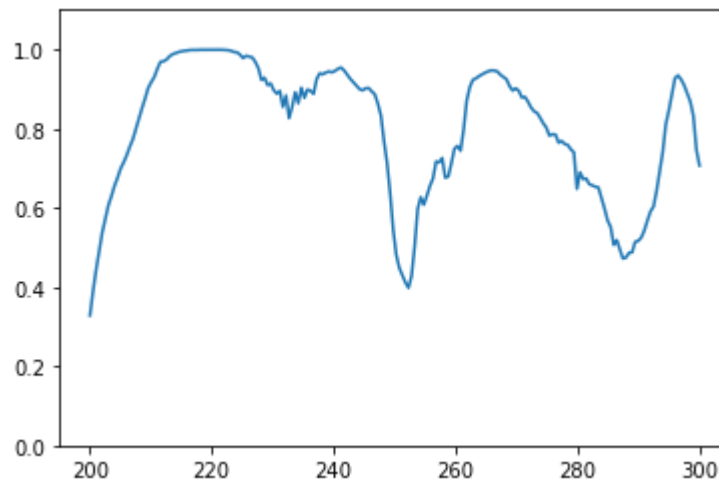


Test 52

True spectrum:

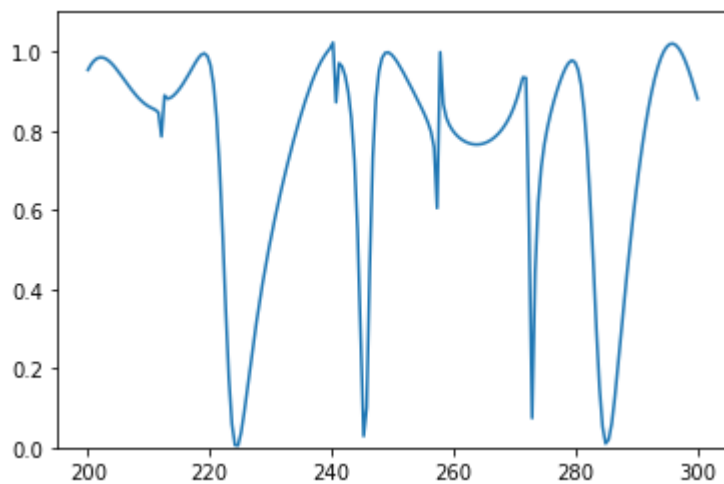


Predicted spectrum:

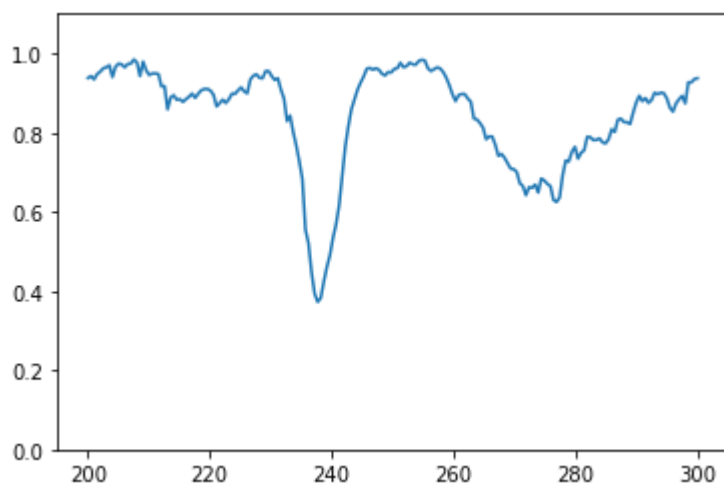


Test 53

True spectrum:

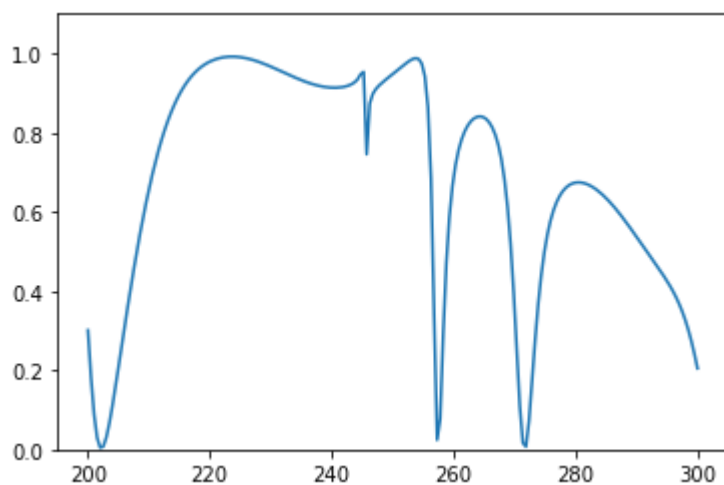


Predicted spectrum:

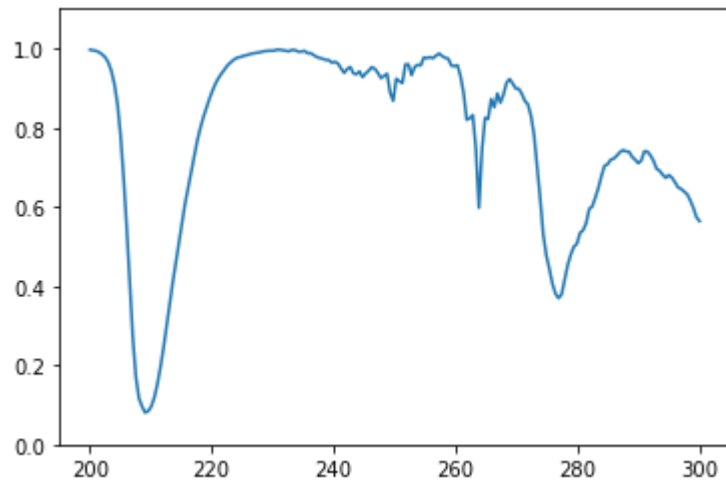


Test 54

True spectrum:

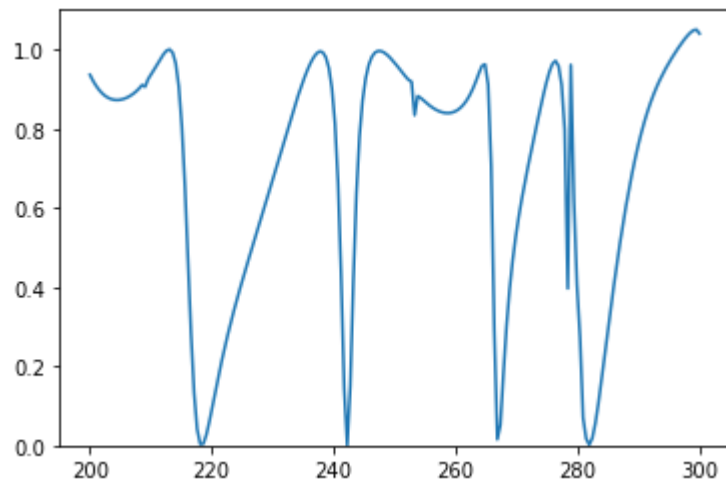


Predicted spectrum:

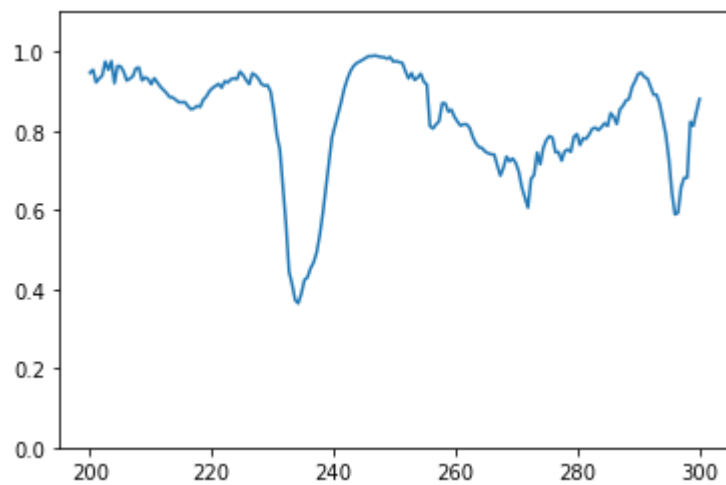


Test 55

True spectrum:

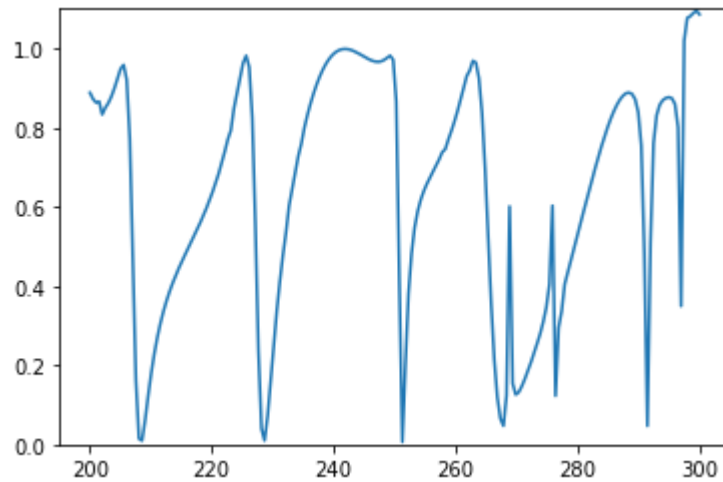


Predicted spectrum:

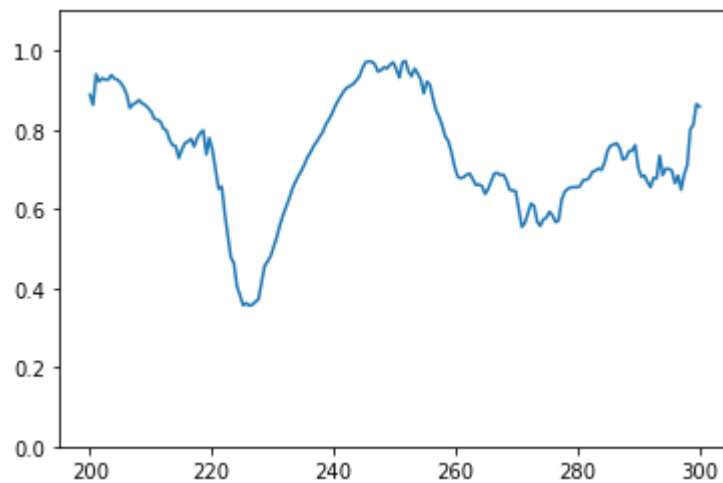


Test 56

True spectrum:

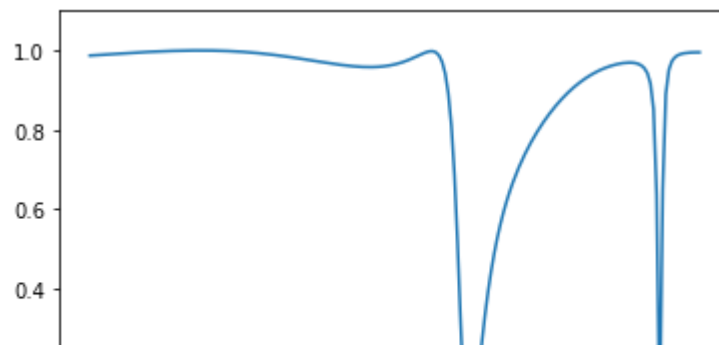


Predicted spectrum:

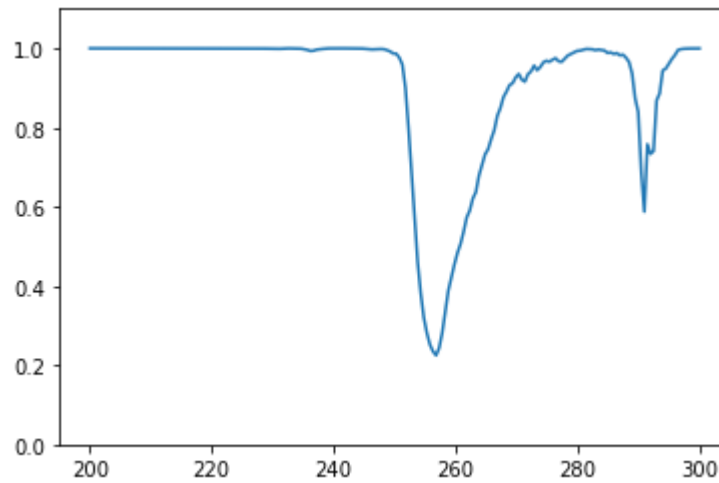


Test 57

True spectrum:

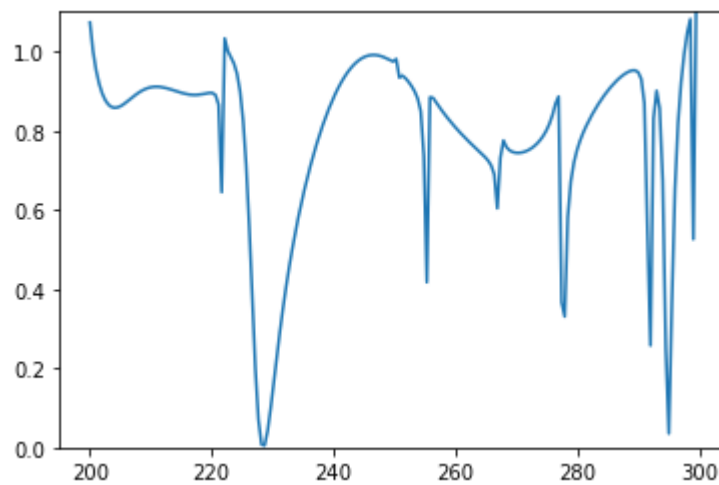


Predicted spectrum:

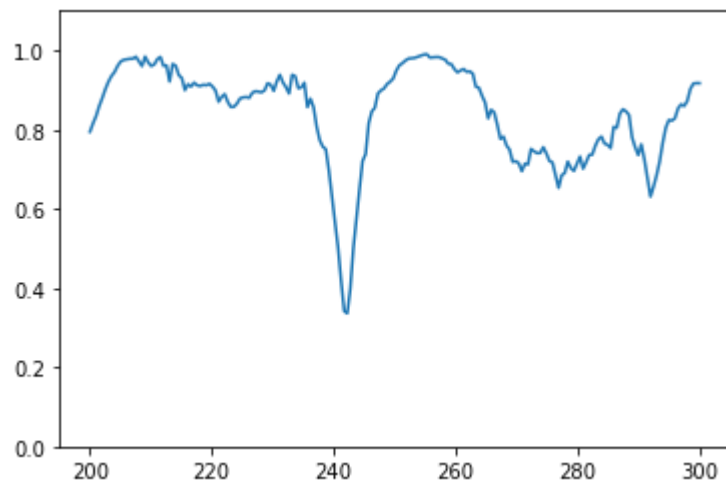


Test 58

True spectrum:

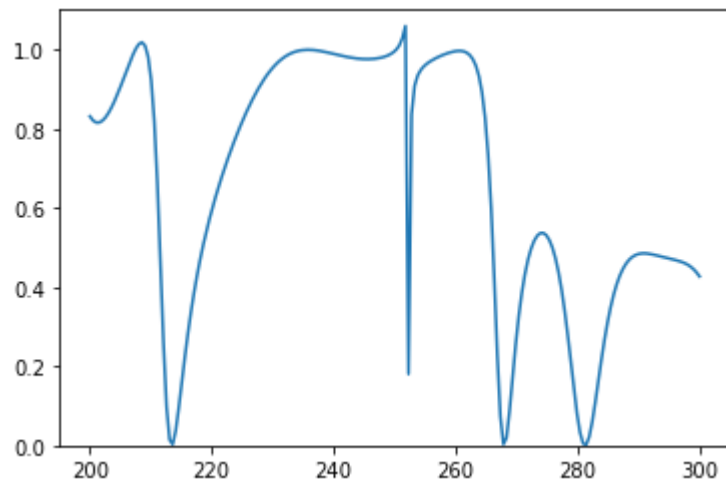


Predicted spectrum:

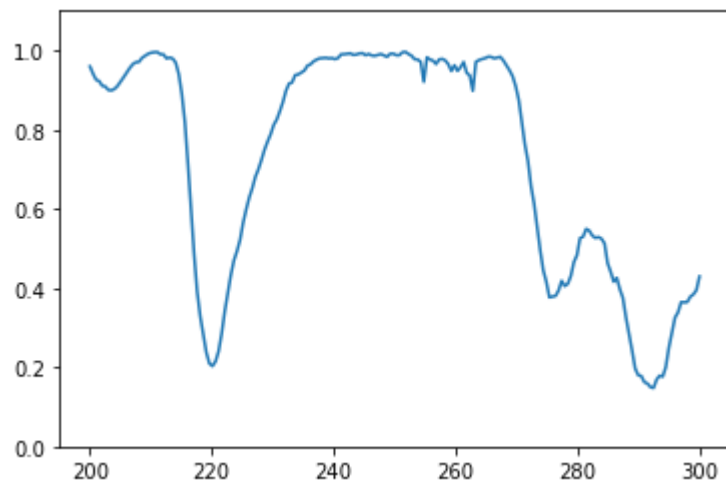


Test 59

True spectrum:

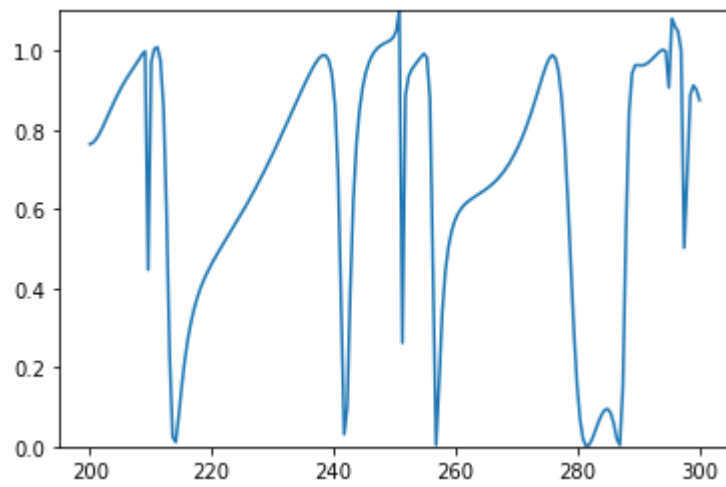


Predicted spectrum:

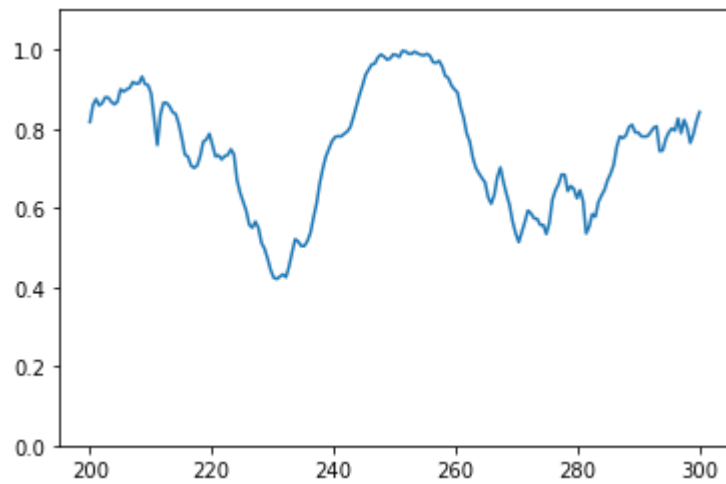


Test 60

True spectrum:

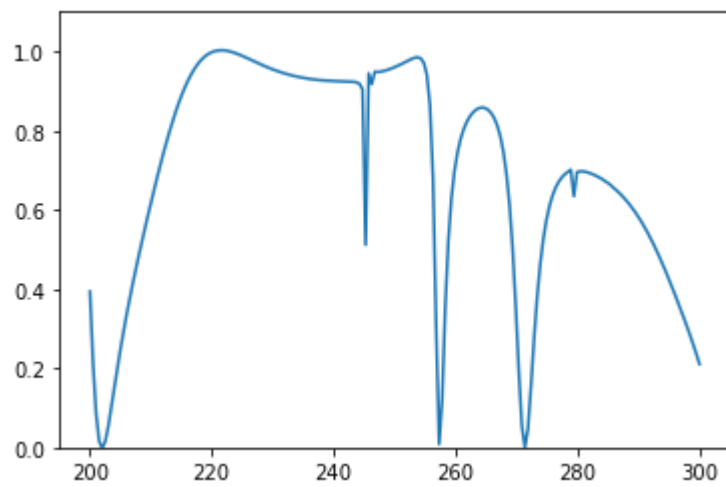


Predicted spectrum:

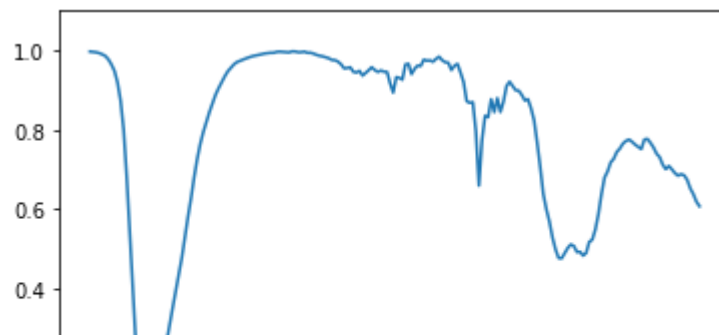


Test 61

True spectrum:

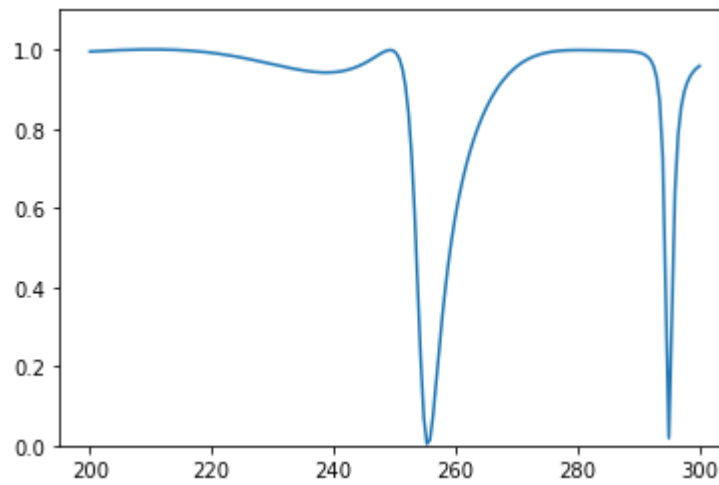


Predicted spectrum:

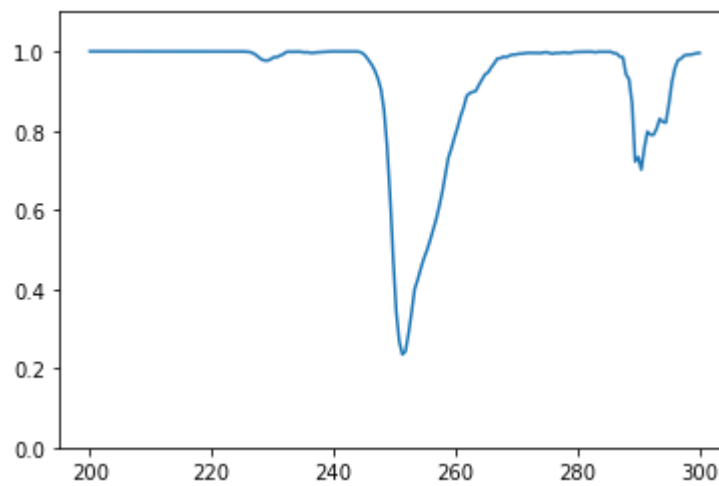


Test 62

True spectrum:

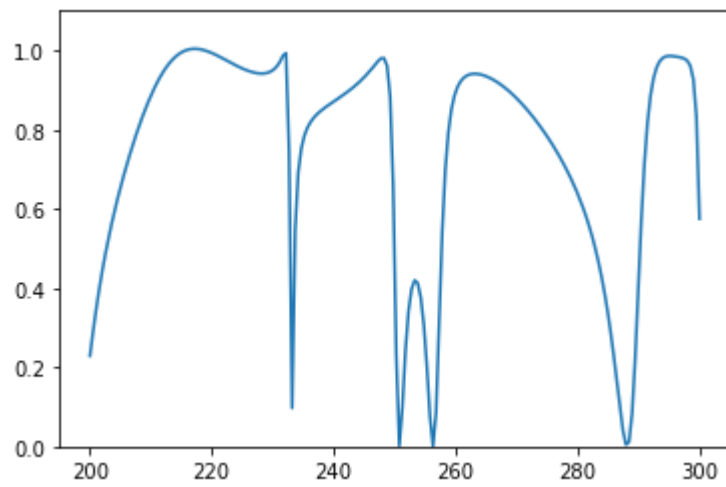


Predicted spectrum:

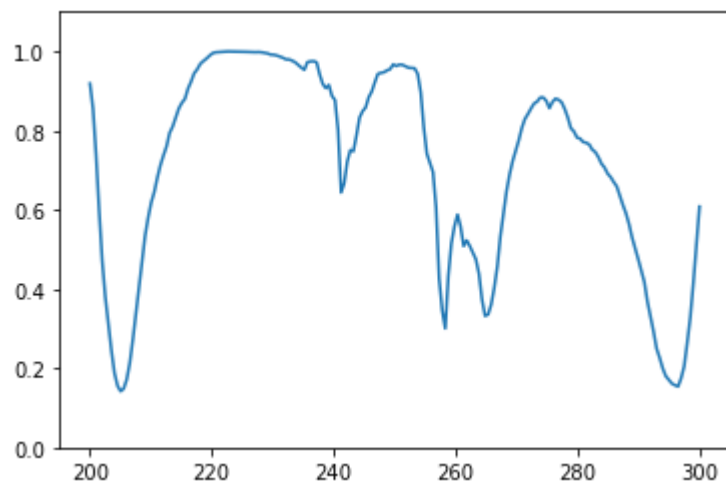


Test 63

True spectrum:

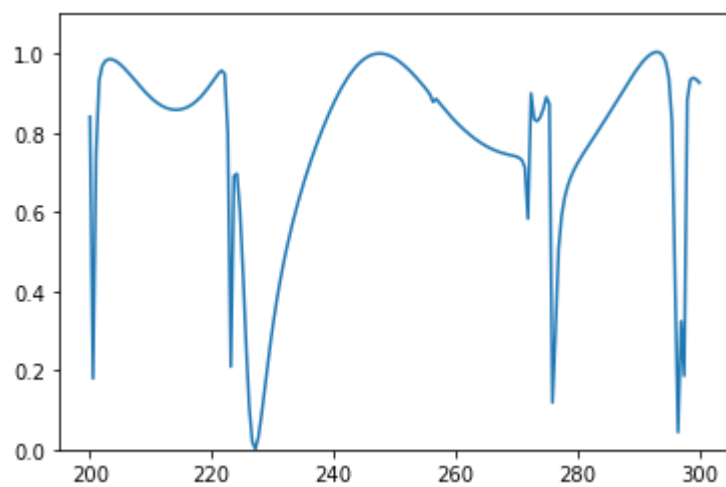


Predicted spectrum:

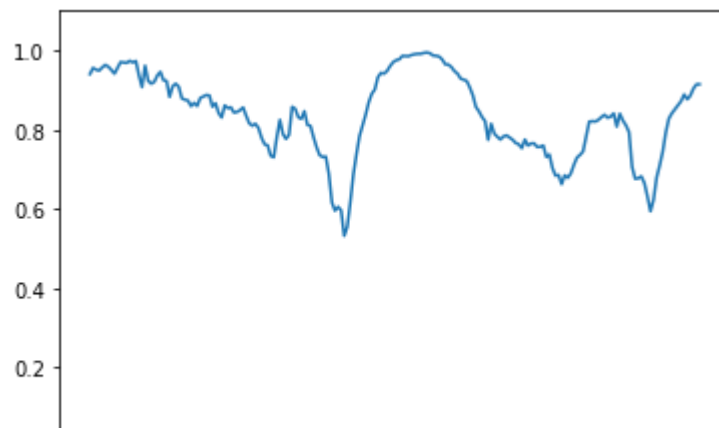


Test 64

True spectrum:

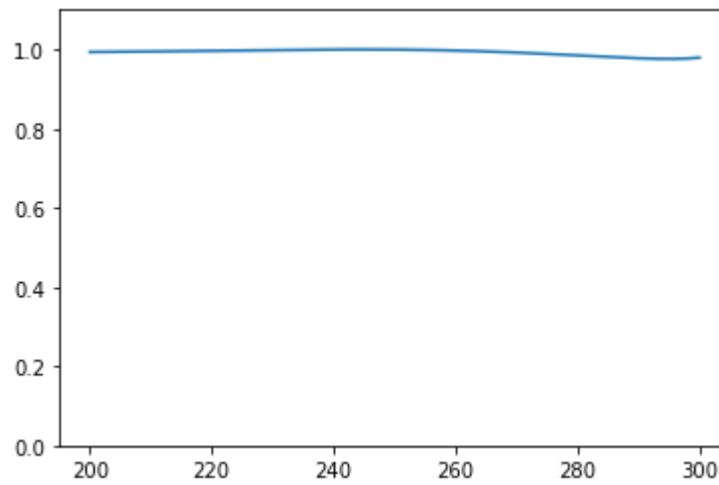


Predicted spectrum:

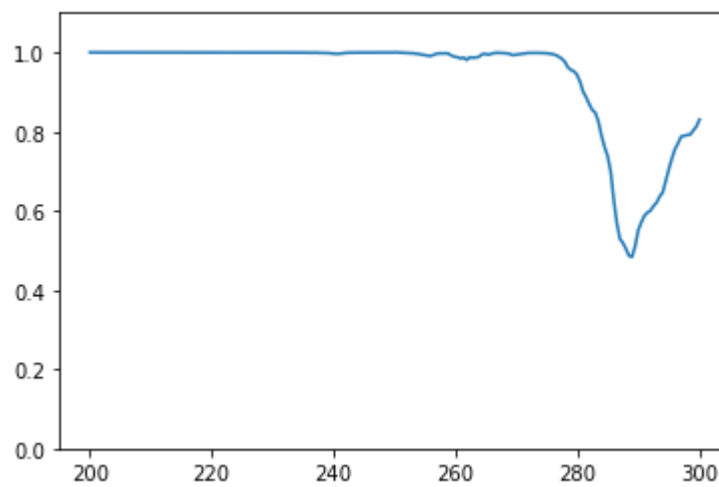


Test 65

True spectrum:

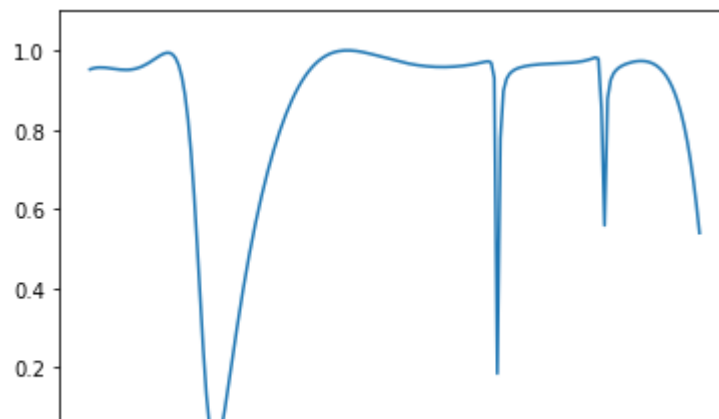


Predicted spectrum:

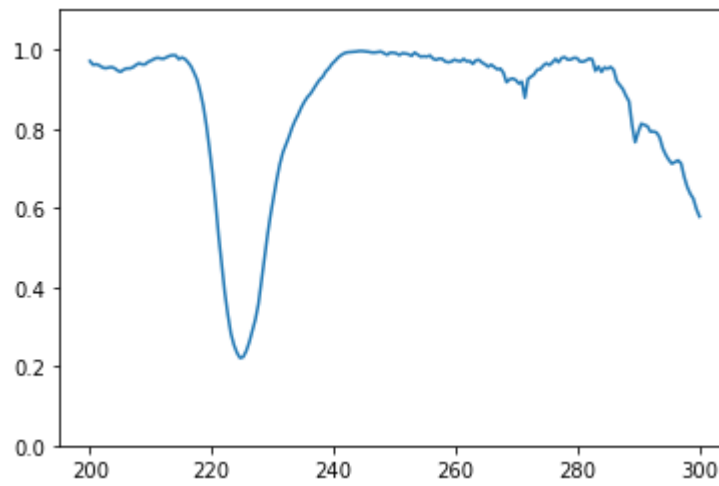


Test 66

True spectrum:

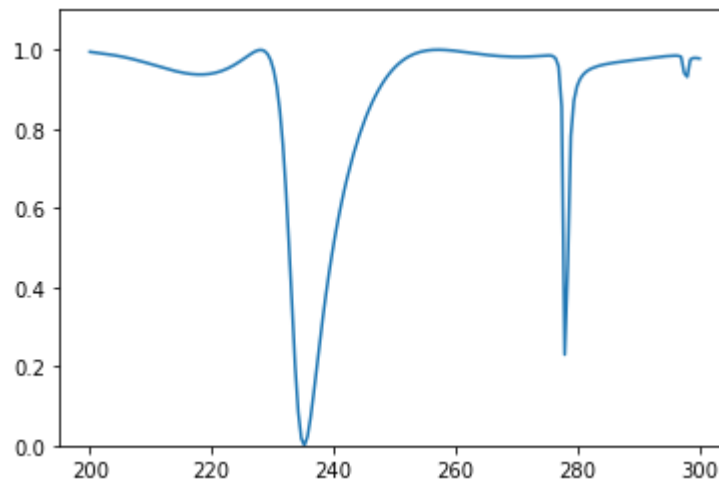


Predicted spectrum:

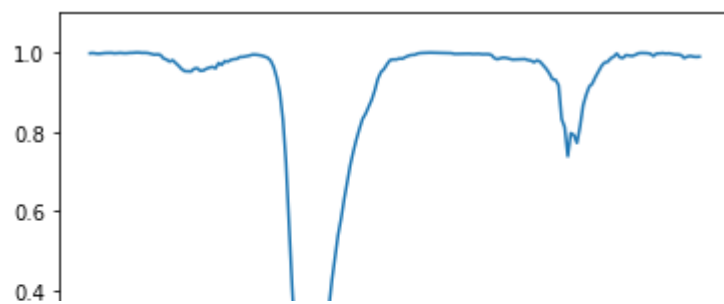


Test 67

True spectrum:

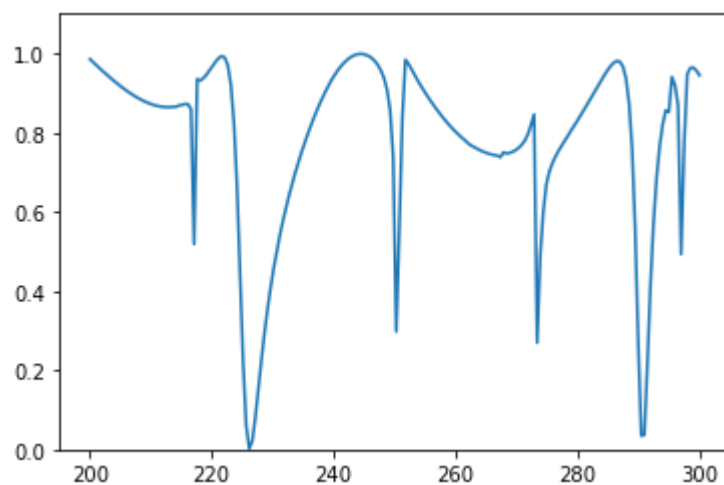


Predicted spectrum:

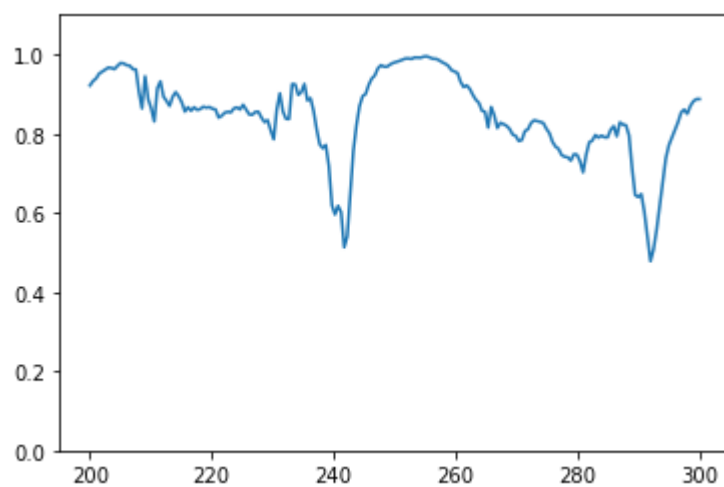


Test 68

True spectrum:

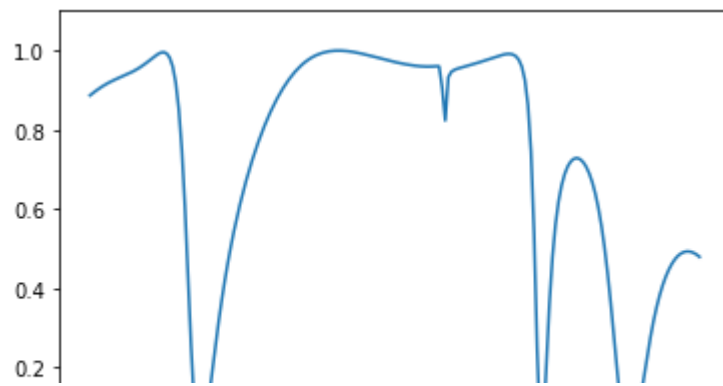


Predicted spectrum:

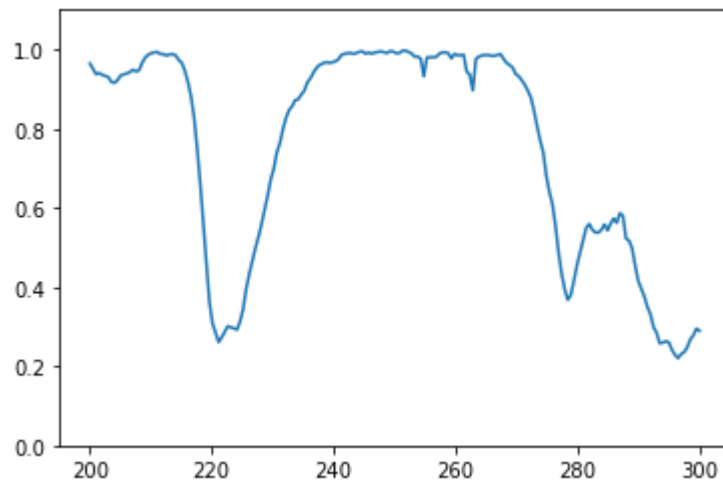


Test 69

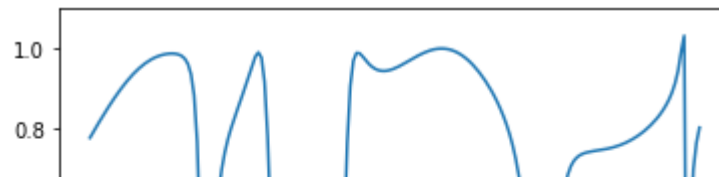
True spectrum:



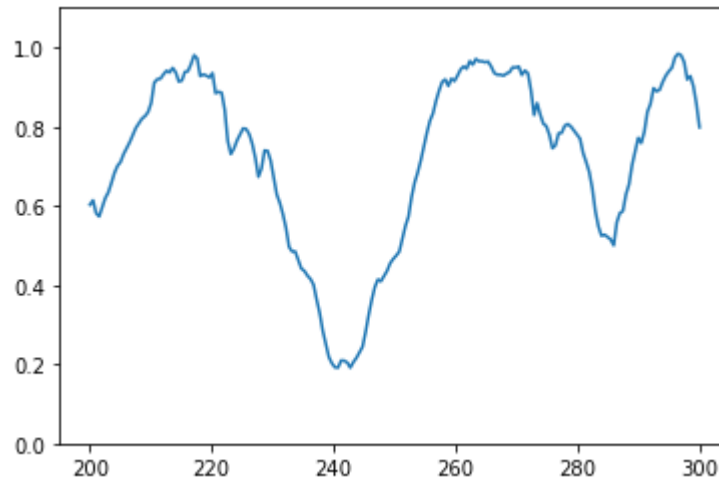
Predicted spectrum:



Test 70
True spectrum:

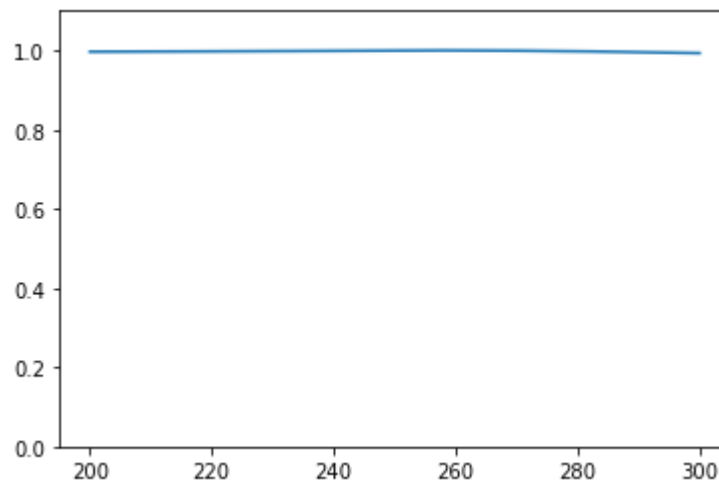


Predicted spectrum:

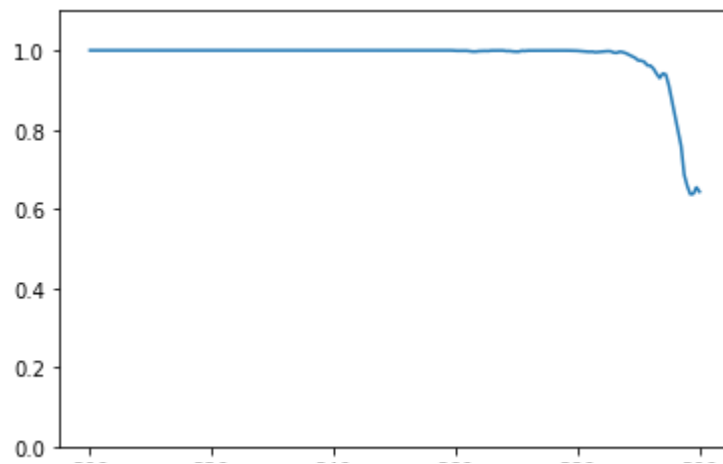


Test 71

True spectrum:

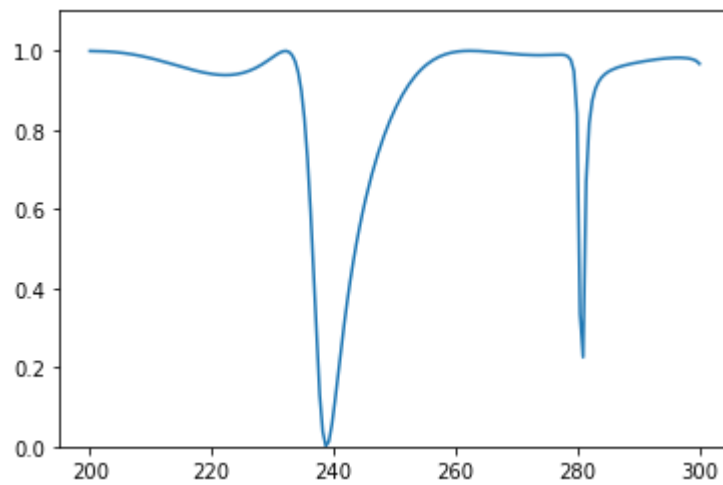


Predicted spectrum:

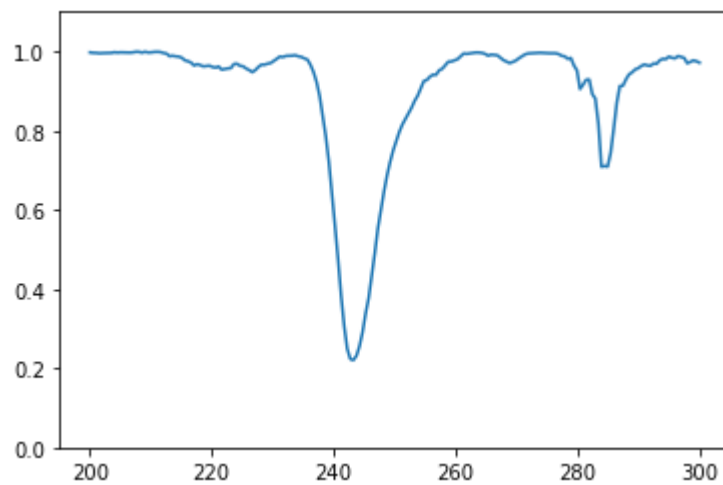


Test 72

True spectrum:

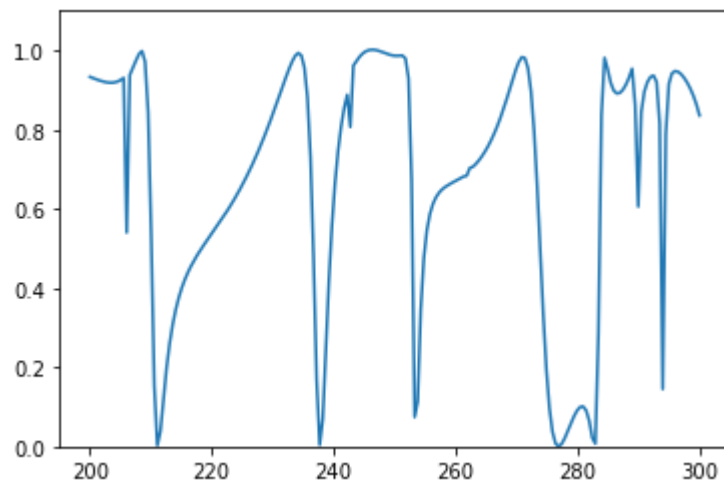


Predicted spectrum:

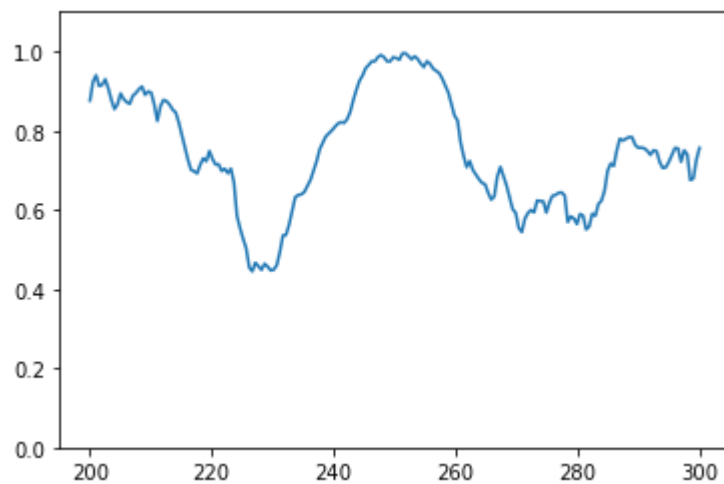


Test 73

True spectrum:

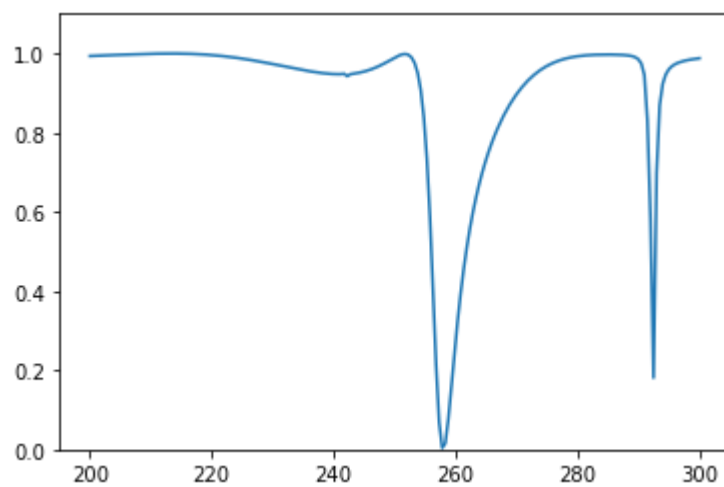


Predicted spectrum:

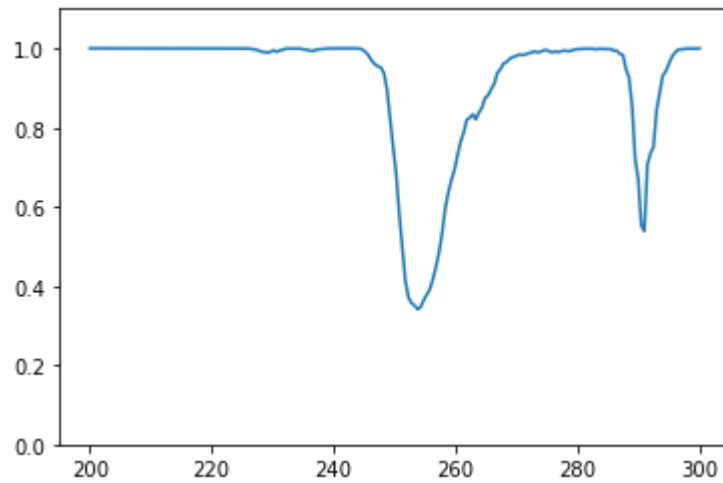


Test 74

True spectrum:

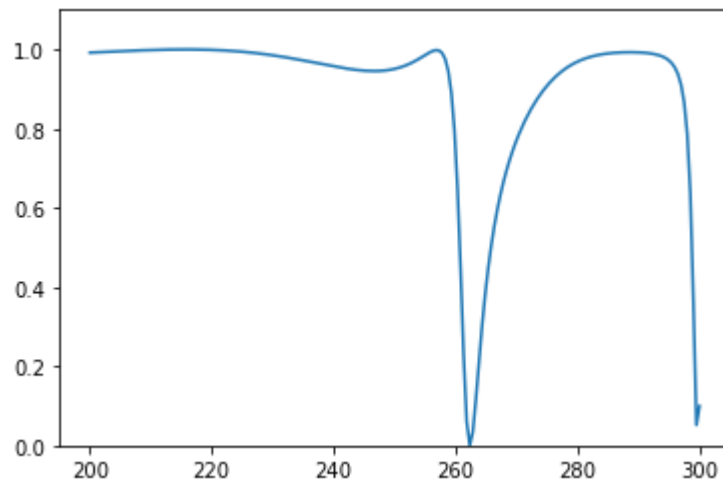


Predicted spectrum:

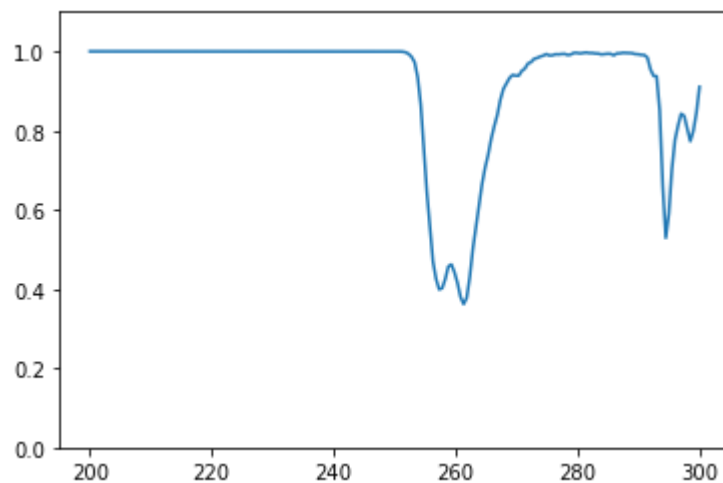


Test 75

True spectrum:

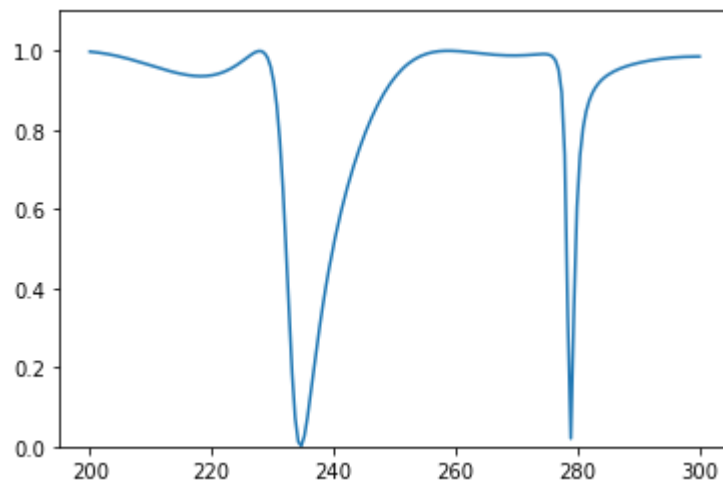


Predicted spectrum:

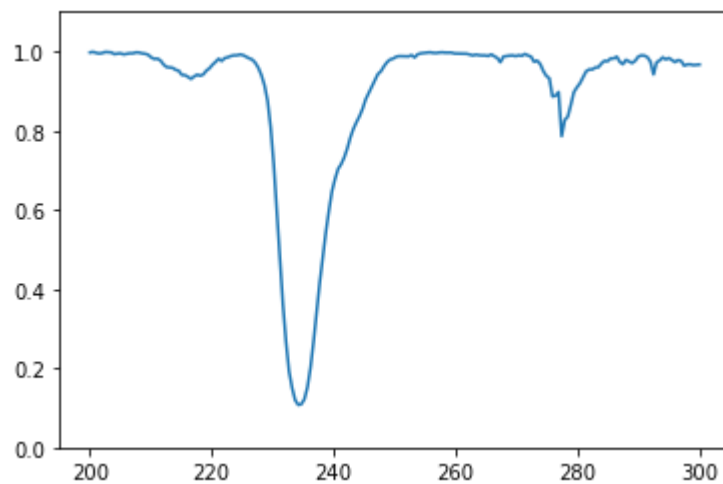


Test 76

True spectrum:

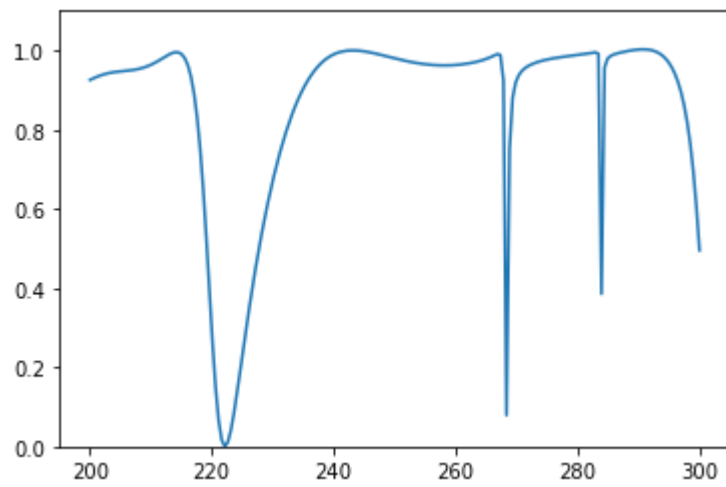


Predicted spectrum:

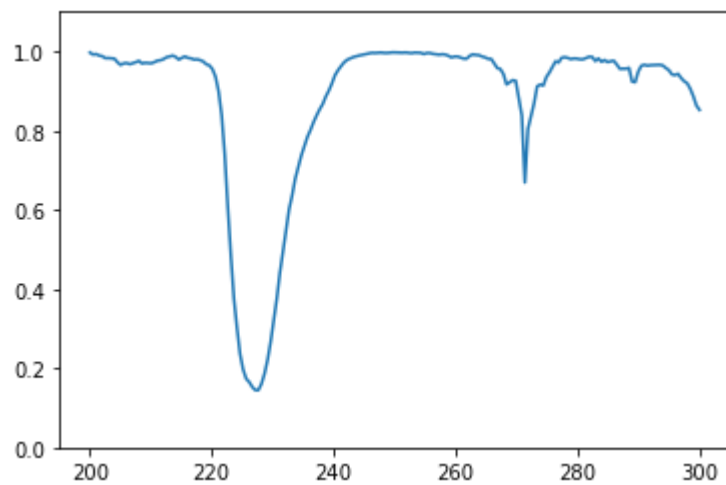


Test 77

True spectrum:

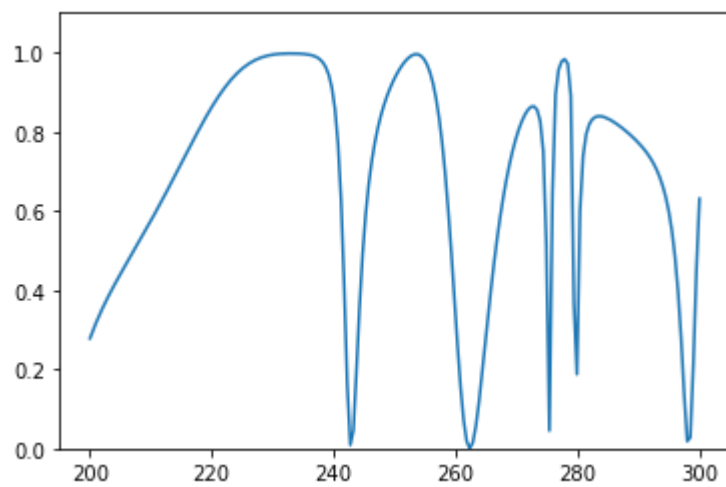


Predicted spectrum:

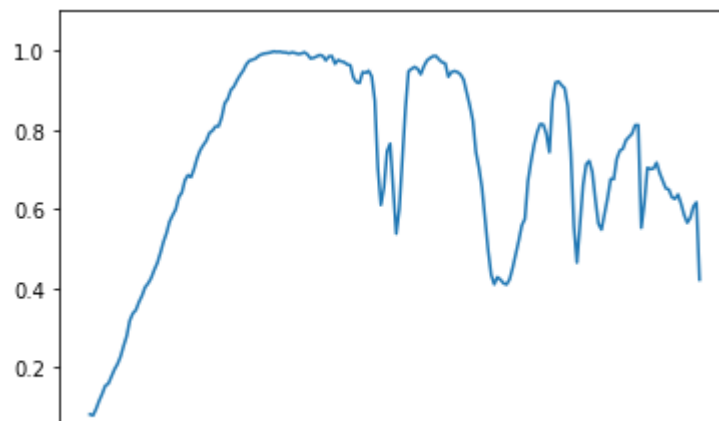


Test 78

True spectrum:

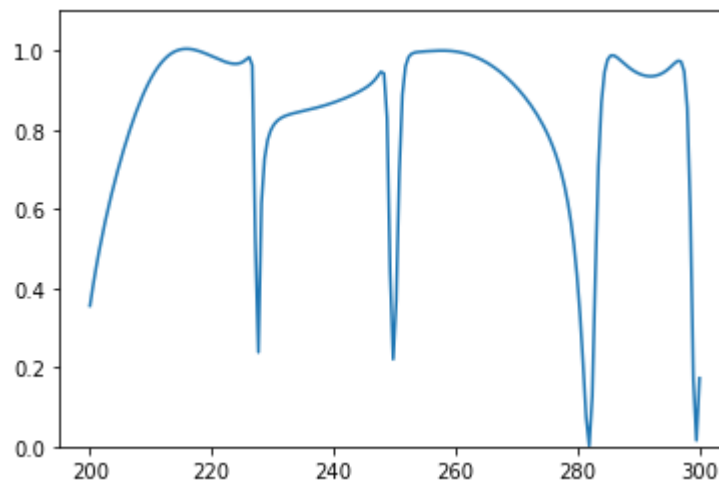


Predicted spectrum:

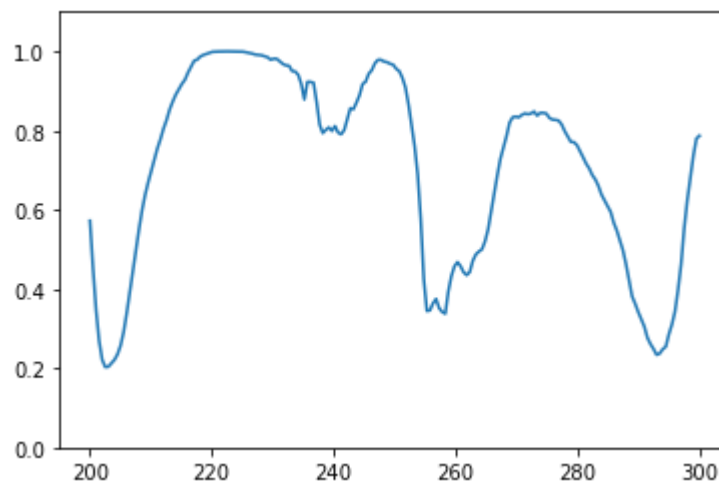


Test 79

True spectrum:

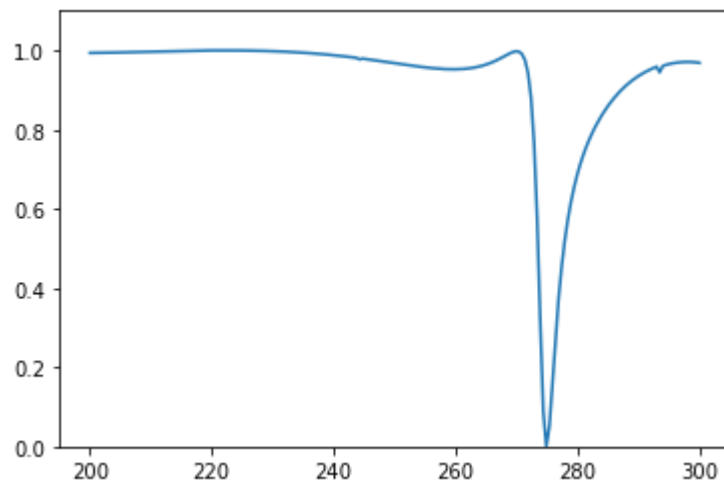


Predicted spectrum:

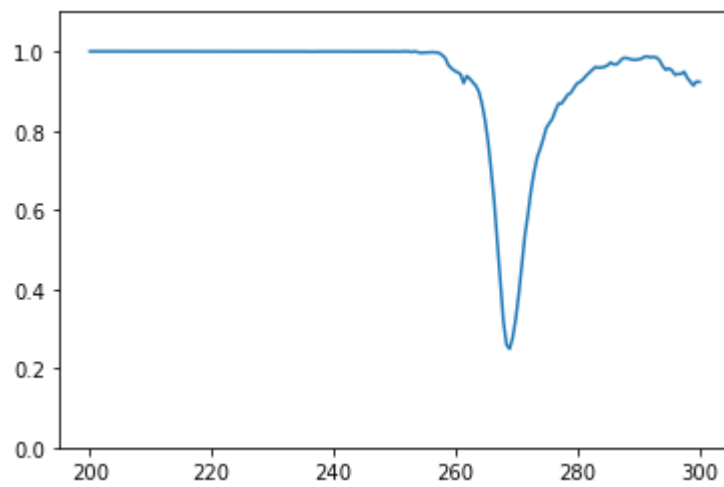


Test 80

True spectrum:

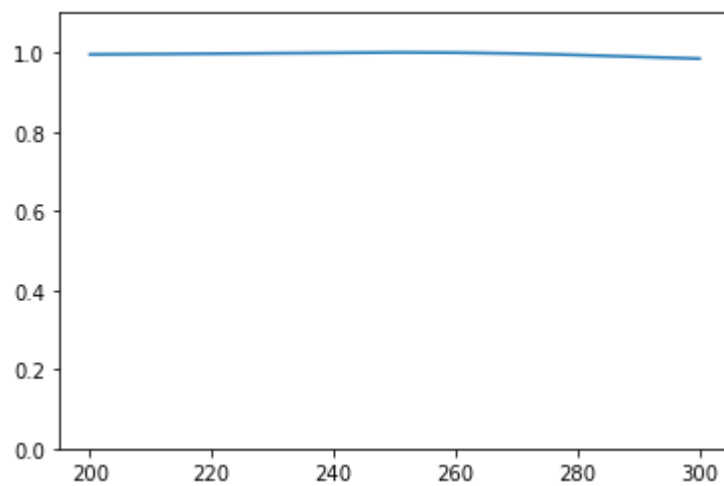


Predicted spectrum:

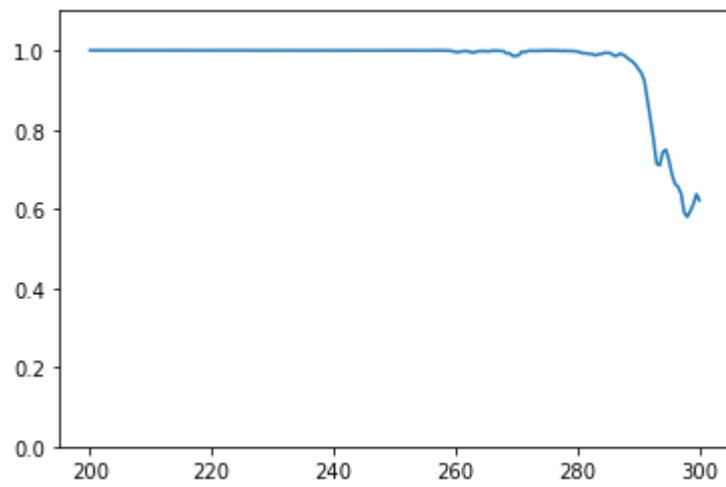


Test 81

True spectrum:

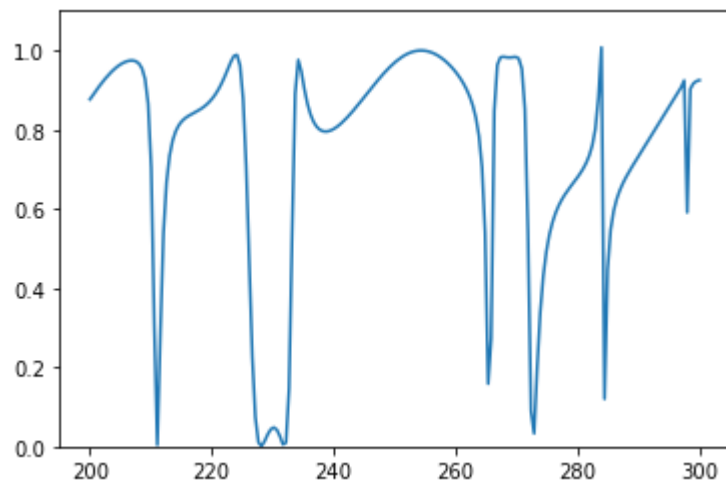


Predicted spectrum:

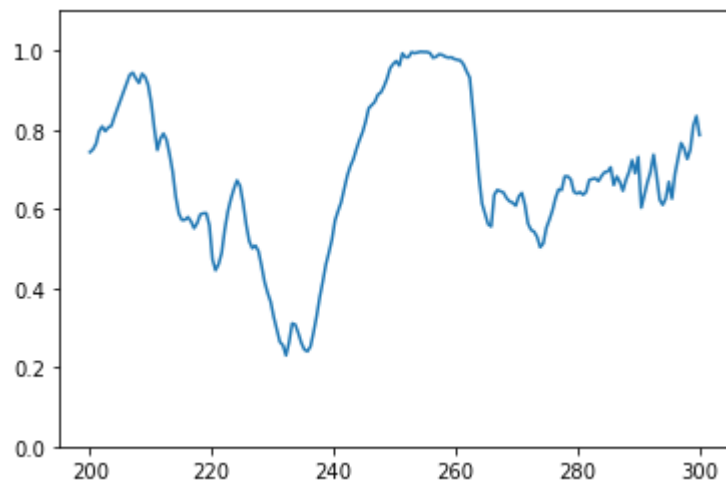


Test 82

True spectrum:

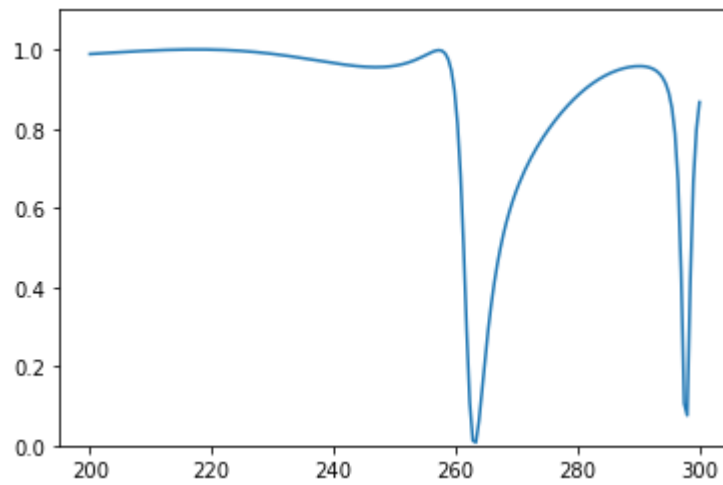


Predicted spectrum:

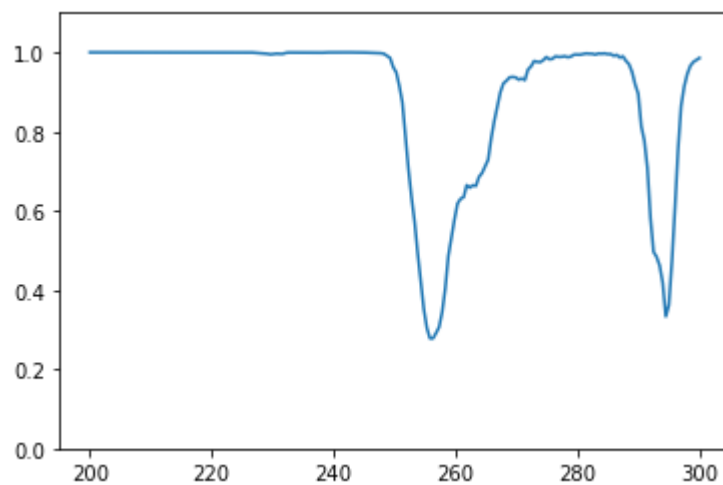


Test 83

True spectrum:

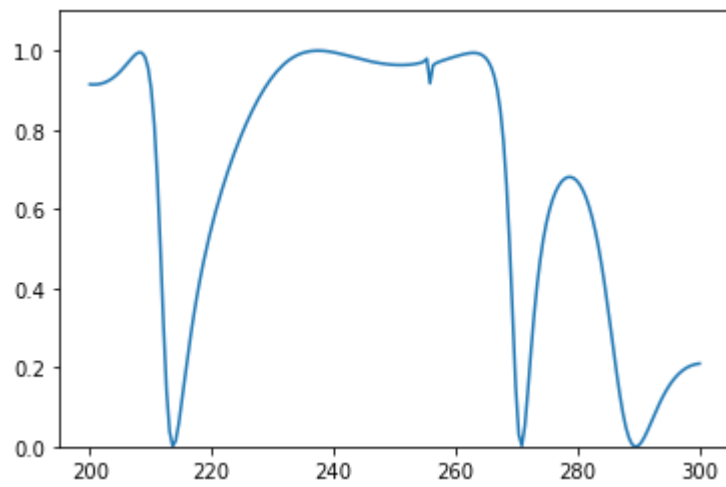


Predicted spectrum:

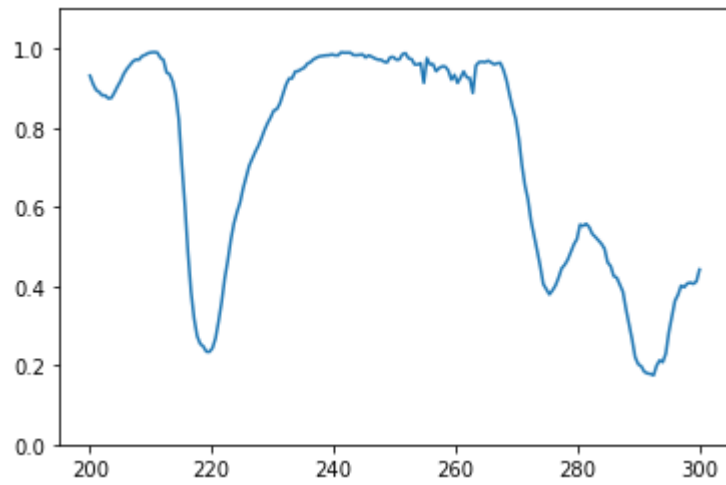


Test 84

True spectrum:

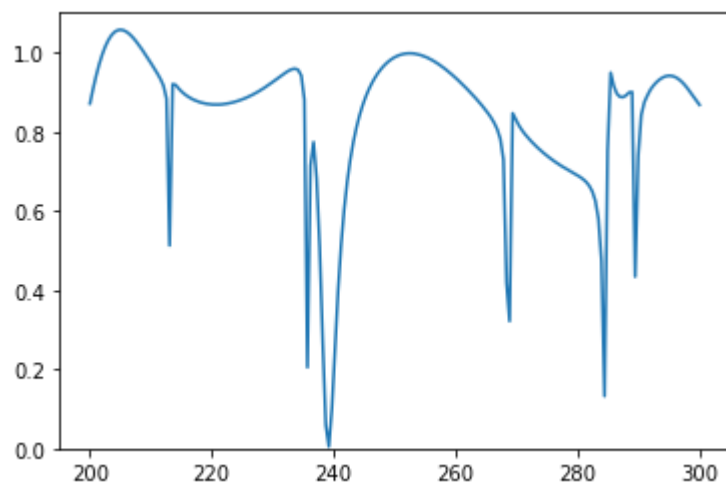


Predicted spectrum:

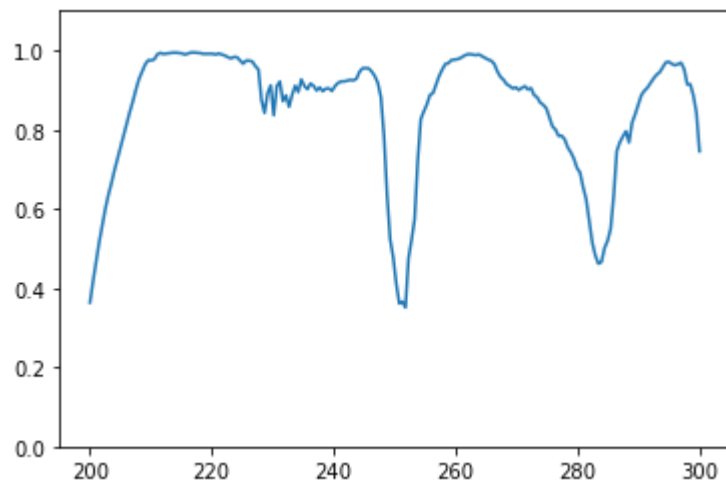


Test 85

True spectrum:

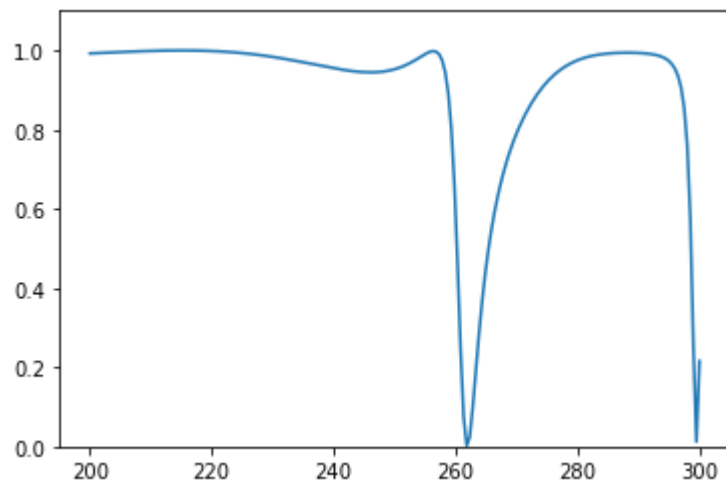


Predicted spectrum:

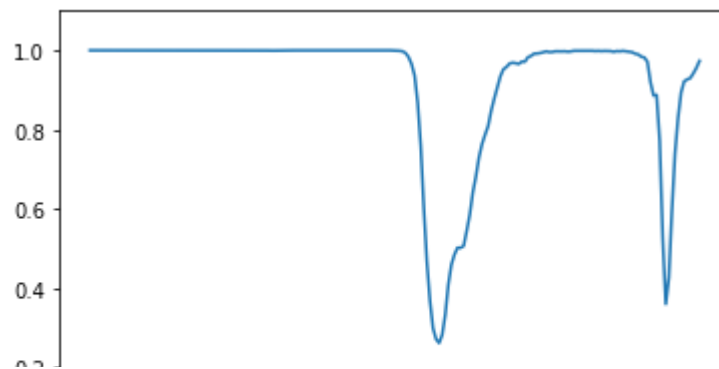


Test 86

True spectrum:

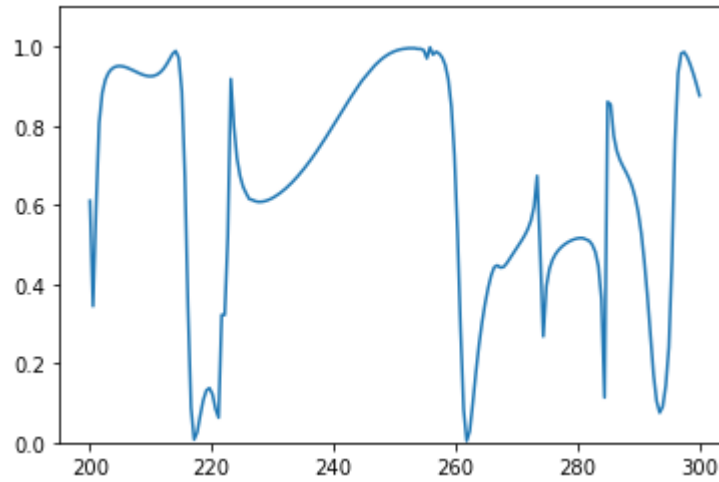


Predicted spectrum:

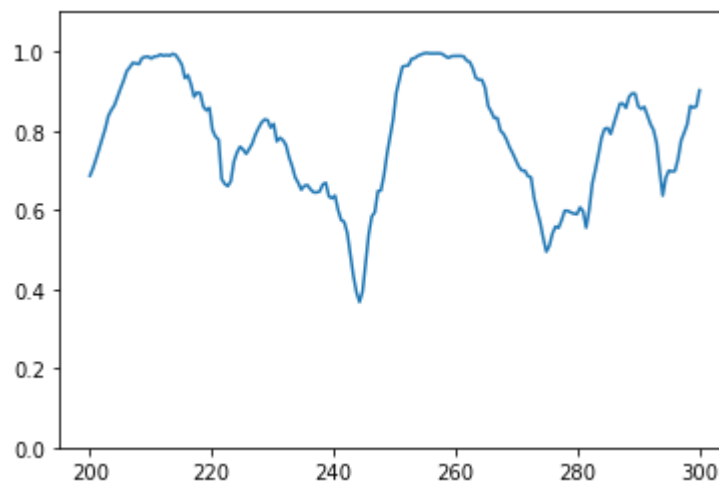


Test 87

True spectrum:

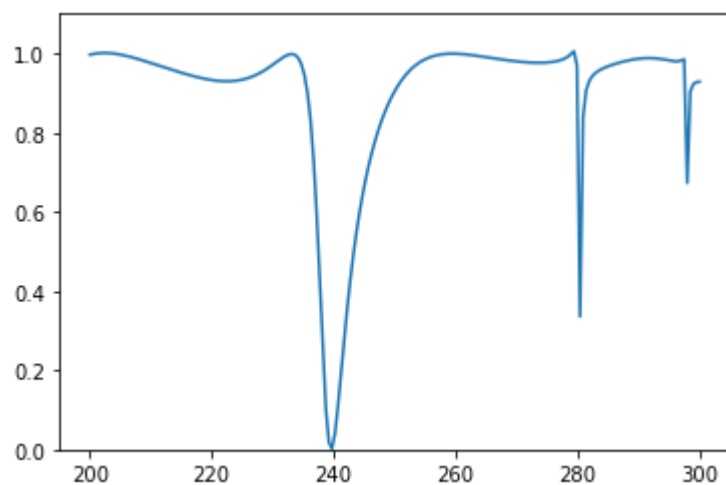


Predicted spectrum:

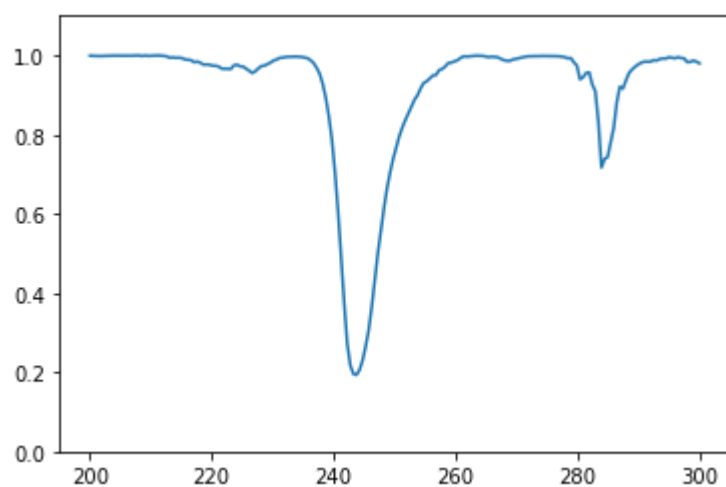


Test 88

True spectrum:

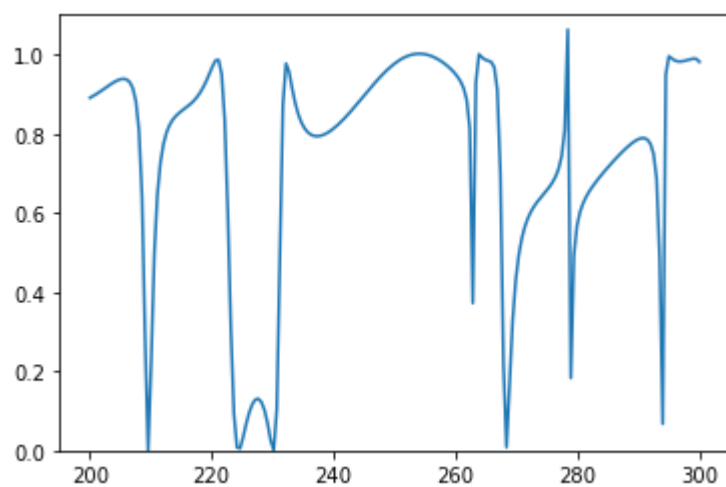


Predicted spectrum:

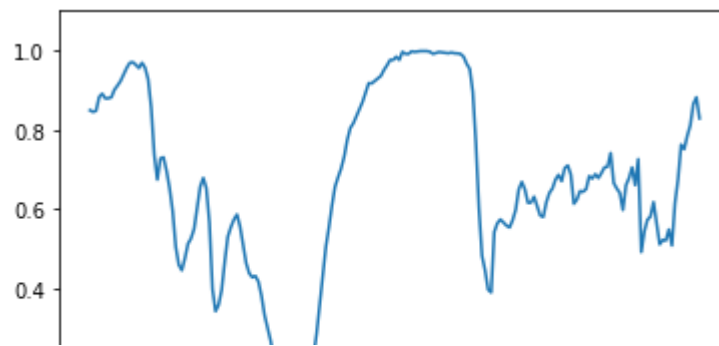


Test 89

True spectrum:

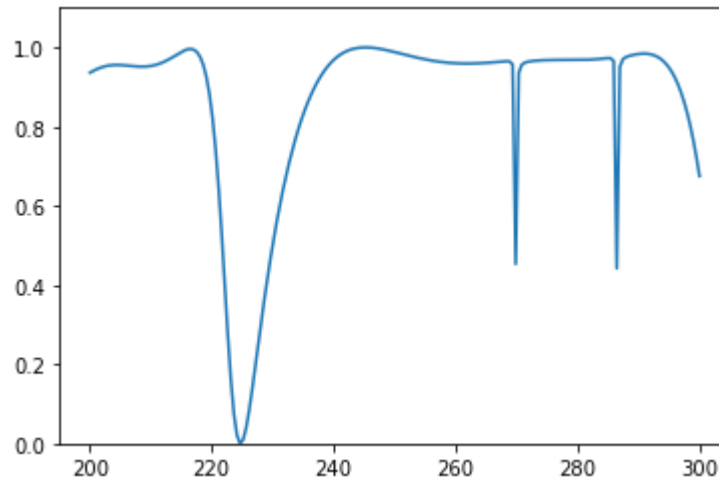


Predicted spectrum:

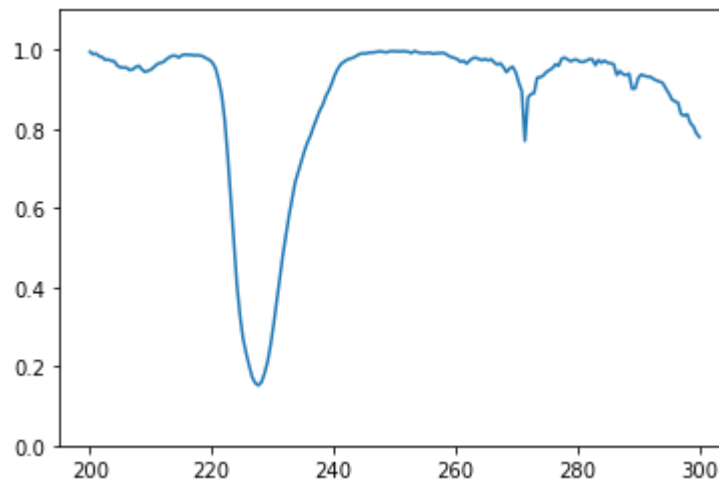


Test 90

True spectrum:

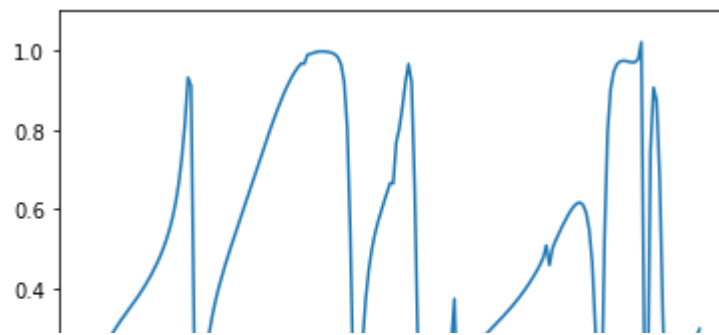


Predicted spectrum:

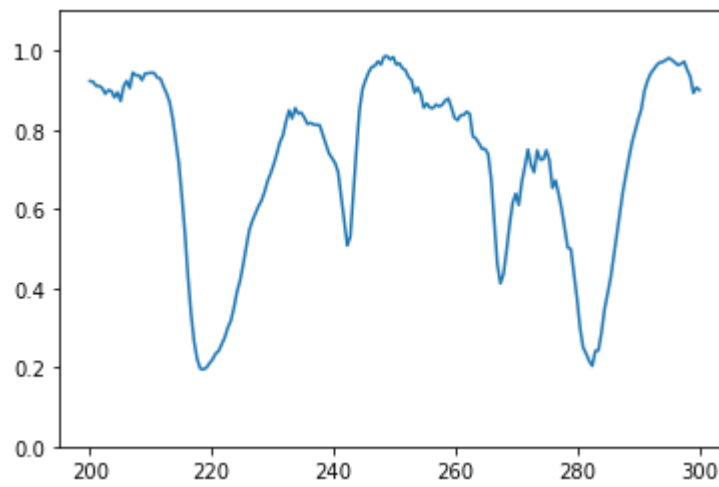


Test 91

True spectrum:

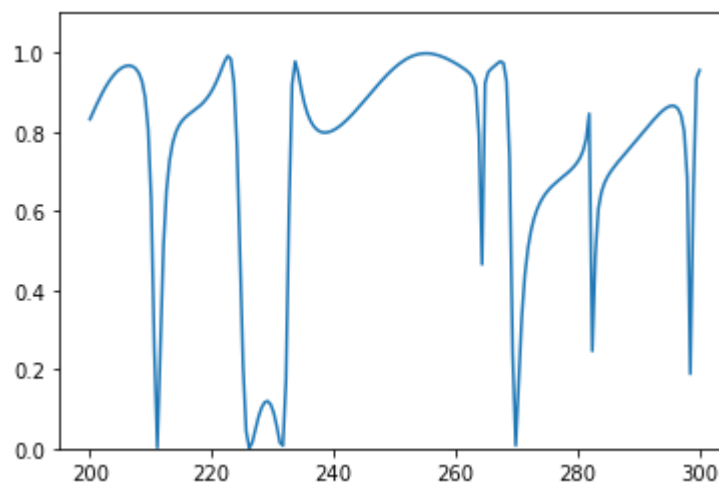


Predicted spectrum:

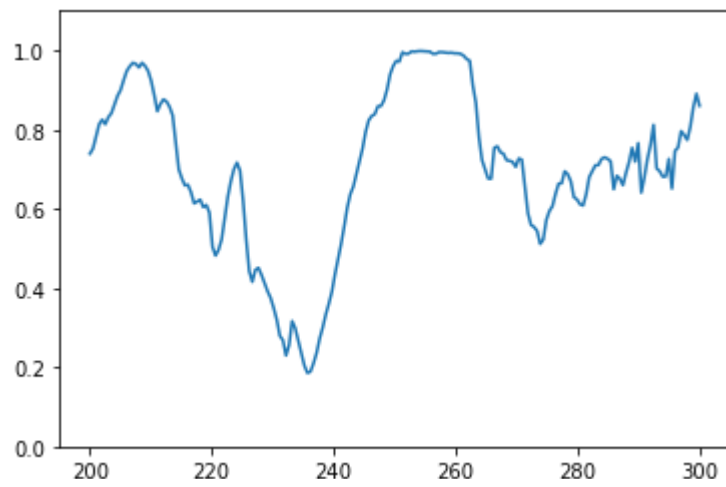


Test 92

True spectrum:

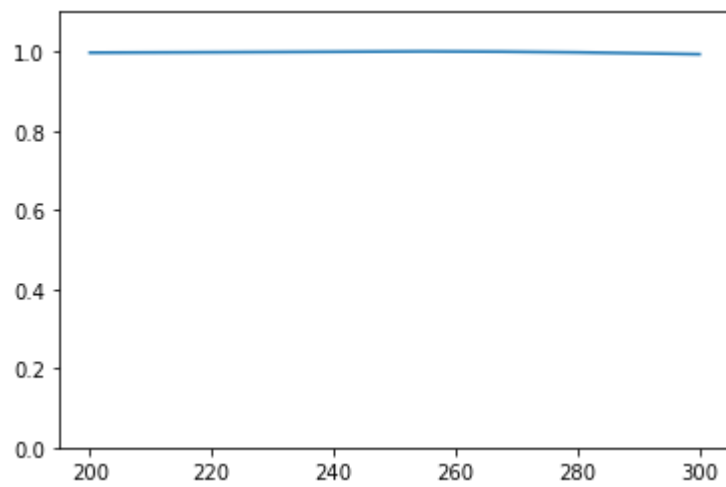


Predicted spectrum:

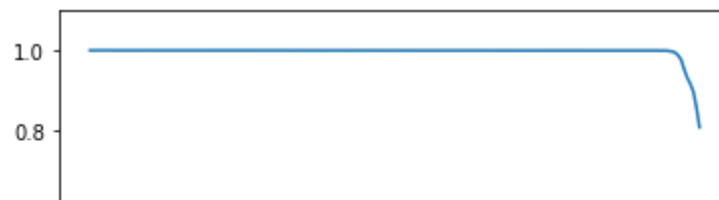


Test 93

True spectrum:

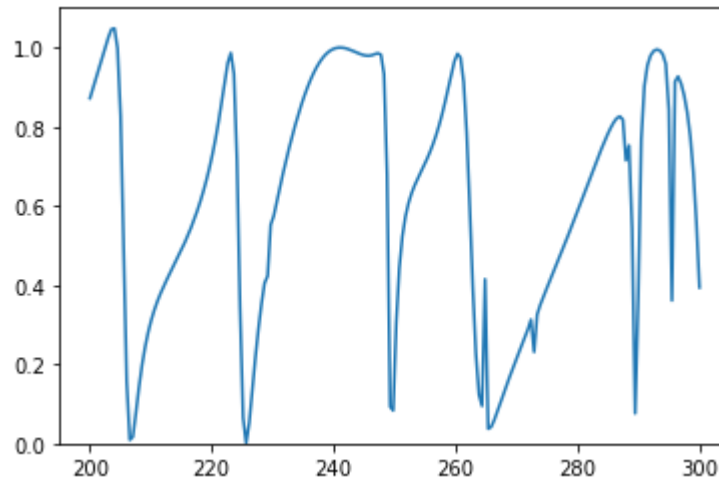


Predicted spectrum:

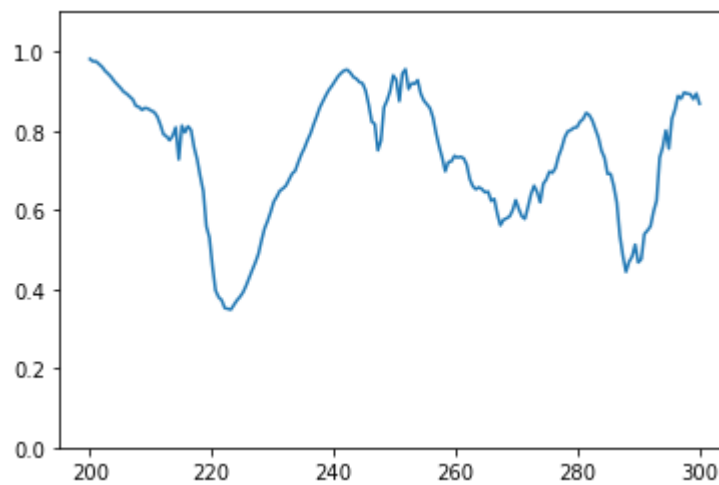


Test 94

True spectrum:

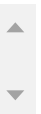


Predicted spectrum:



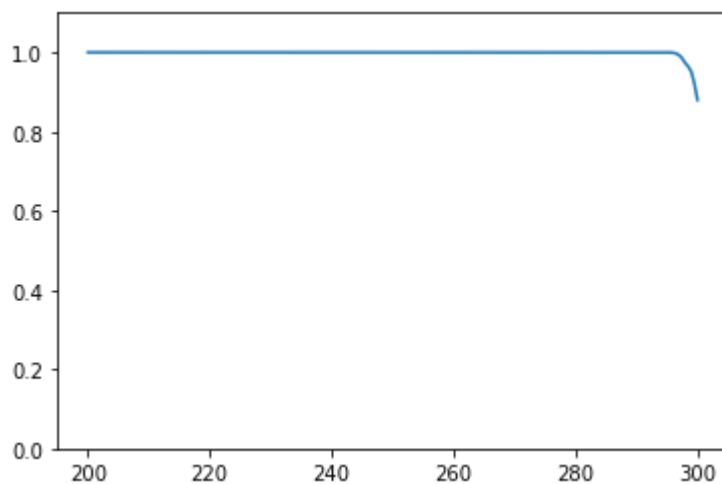
Test 95

True spectrum:



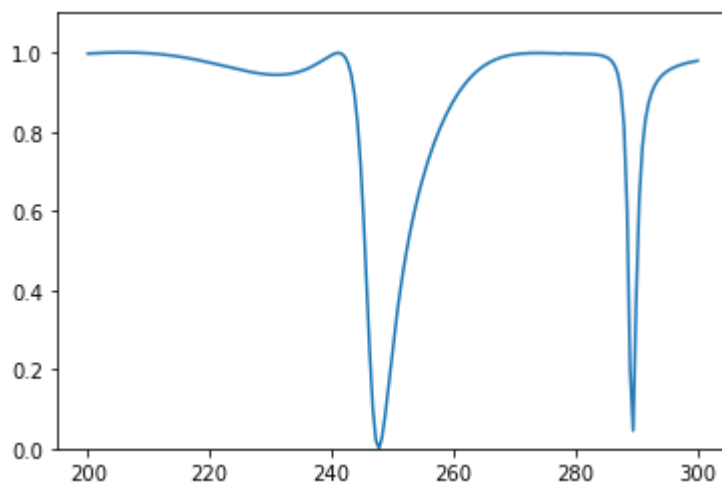


Predicted spectrum:

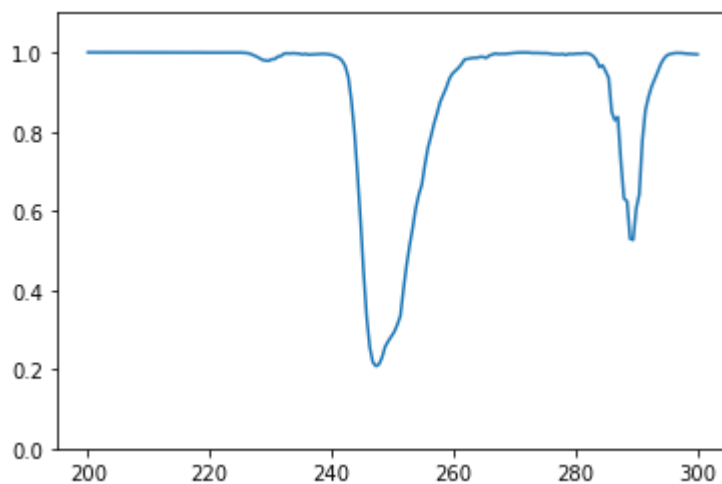


Test 96

True spectrum:

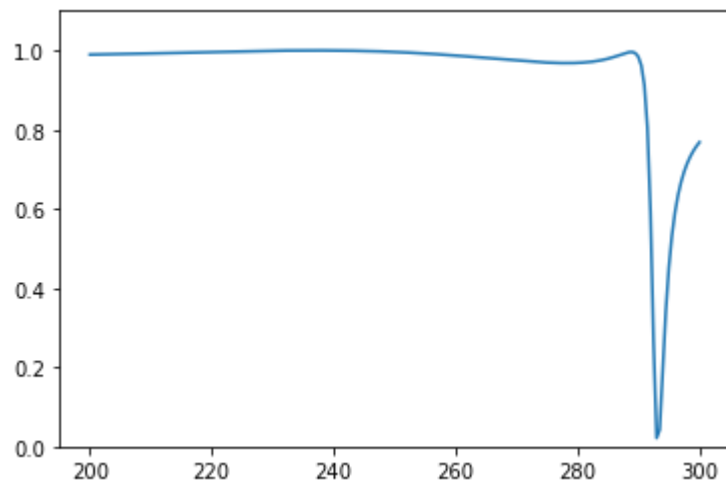


Predicted spectrum:

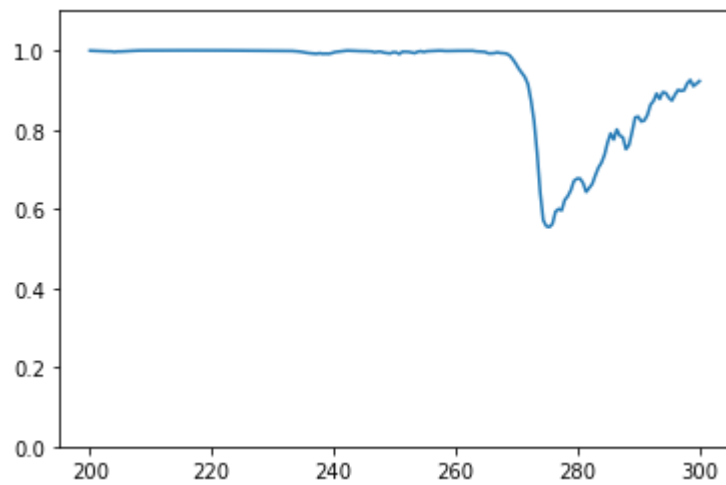


Test 97

True spectrum:



Predicted spectrum:

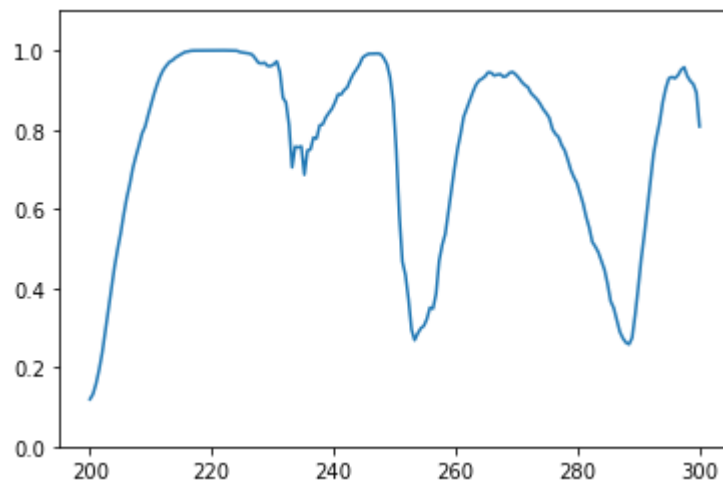


Test 98

True spectrum:

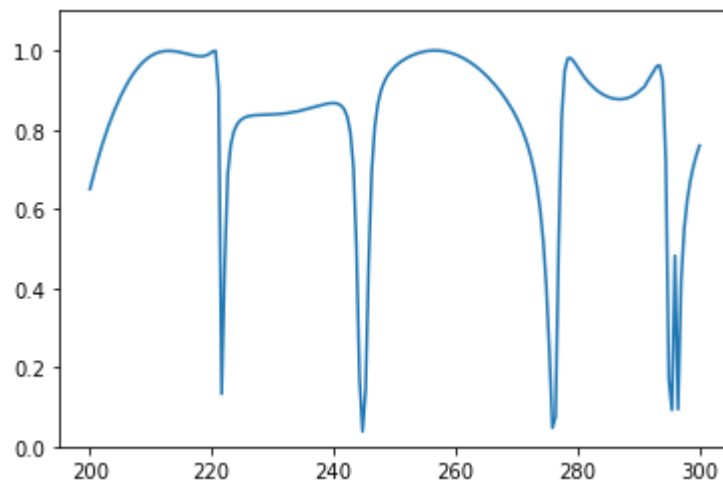


Predicted spectrum:



Test 99

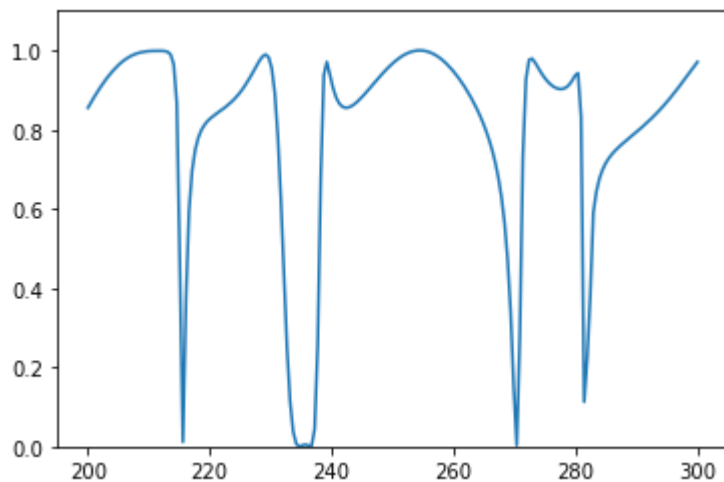
True spectrum:



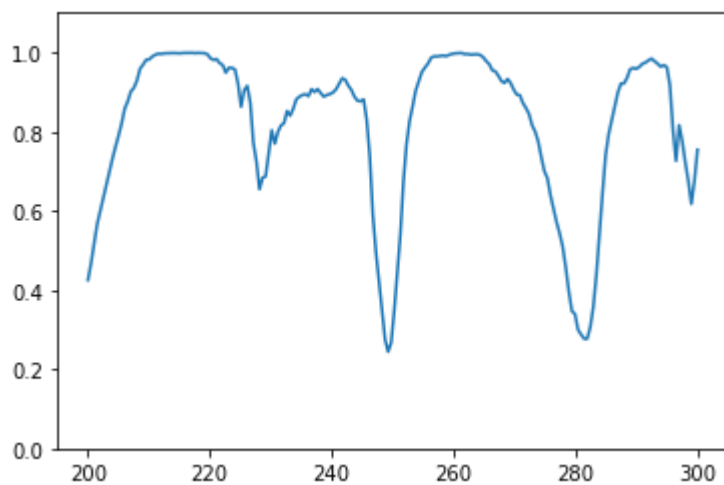
Predicted spectrum:

Test 100

True spectrum:

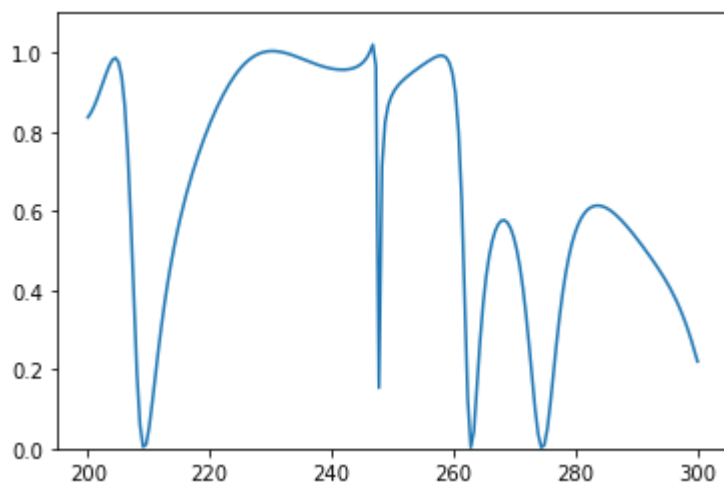


Predicted spectrum:

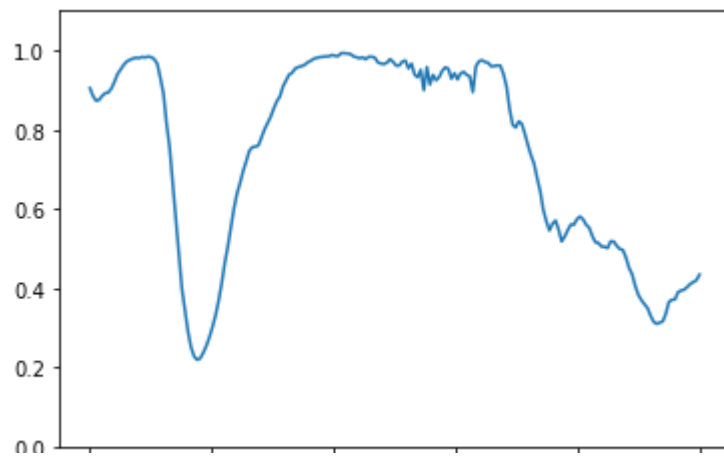


Test 101

True spectrum:

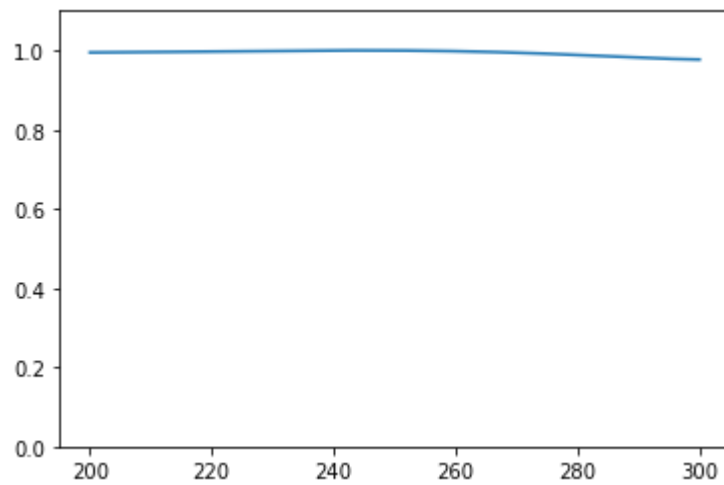


Predicted spectrum:



Test 102

True spectrum:

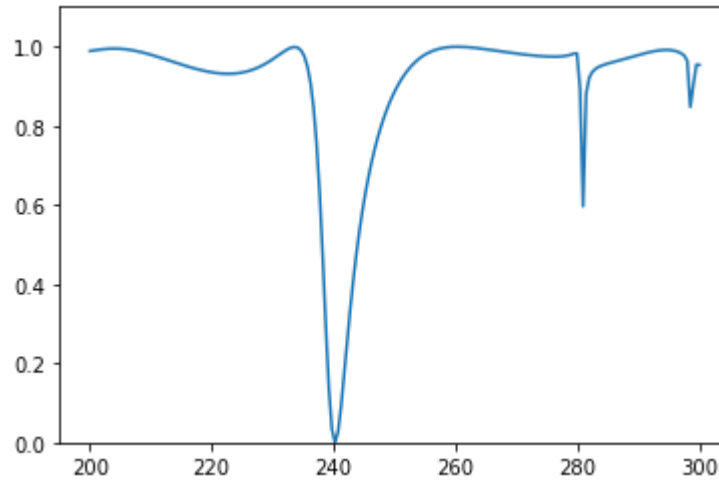


Predicted spectrum:

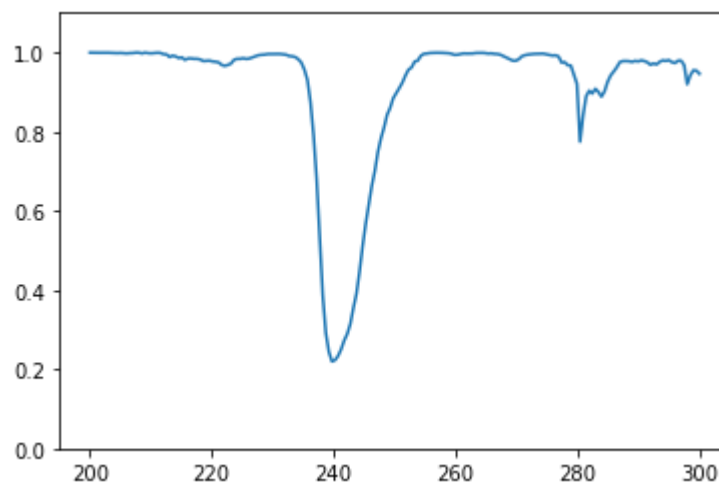


Test 103

True spectrum:

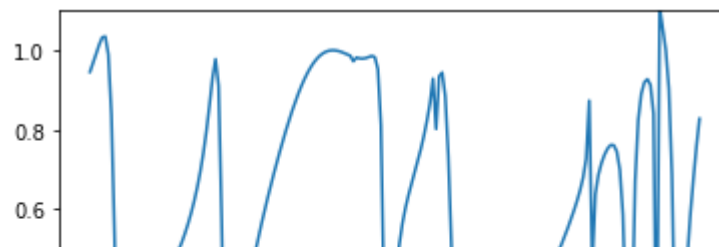


Predicted spectrum:

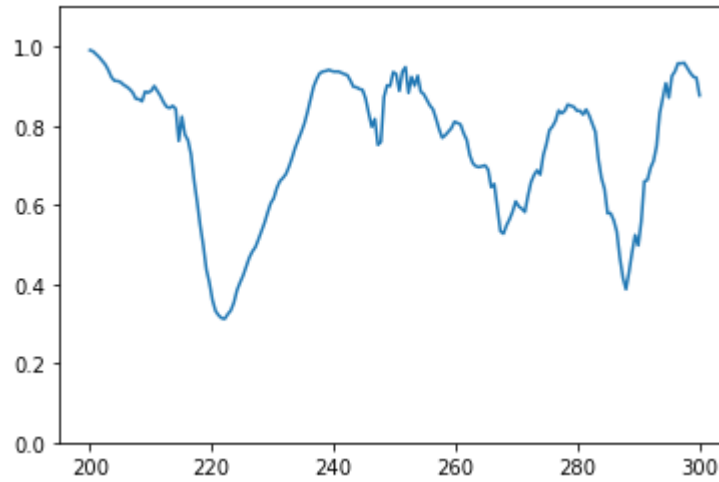


Test 104

True spectrum:

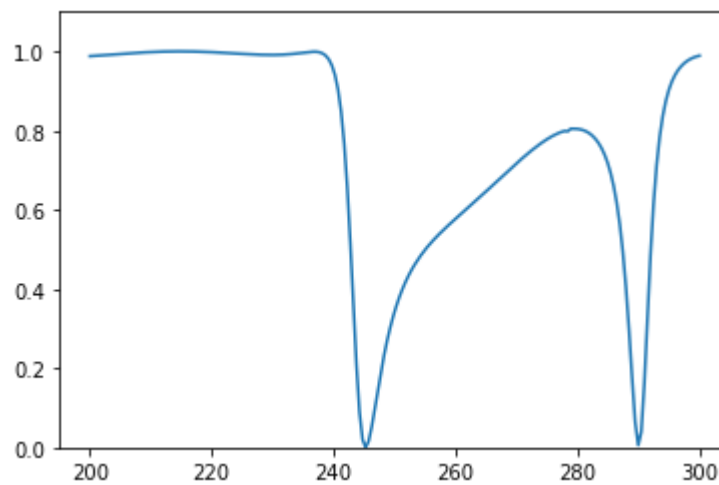


Predicted spectrum:

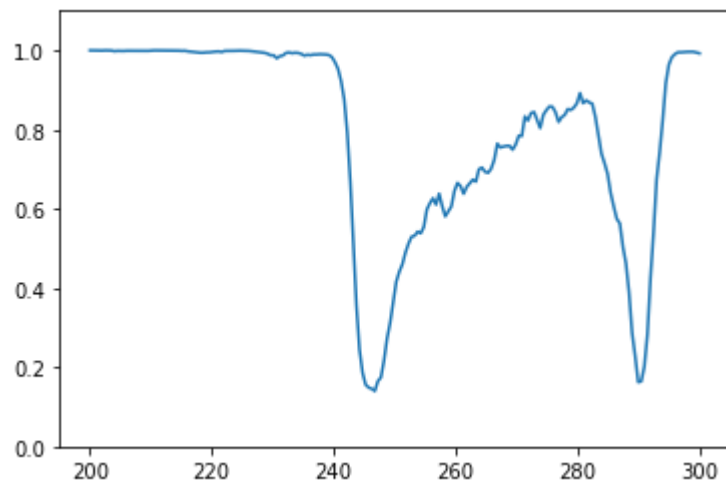


Test 105

True spectrum:

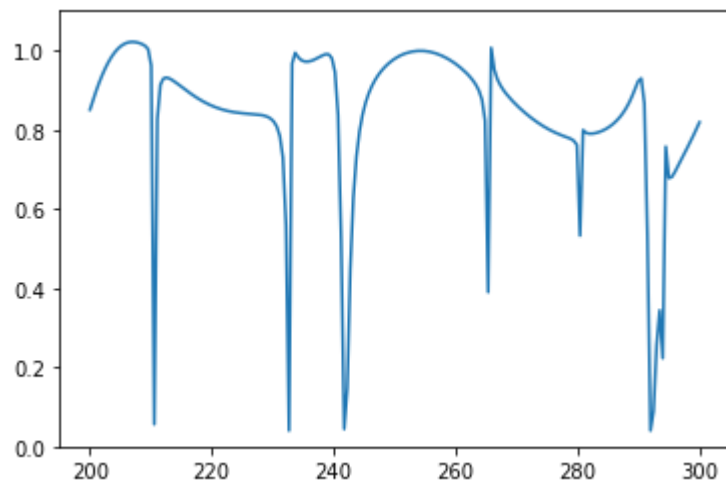


Predicted spectrum:

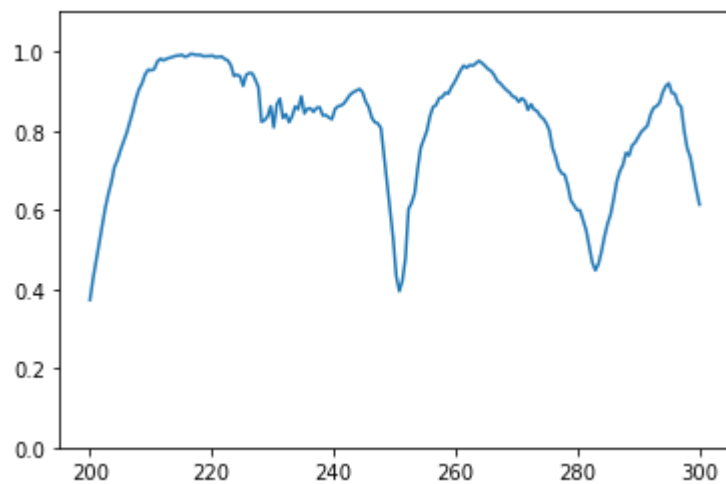


Test 106

True spectrum:

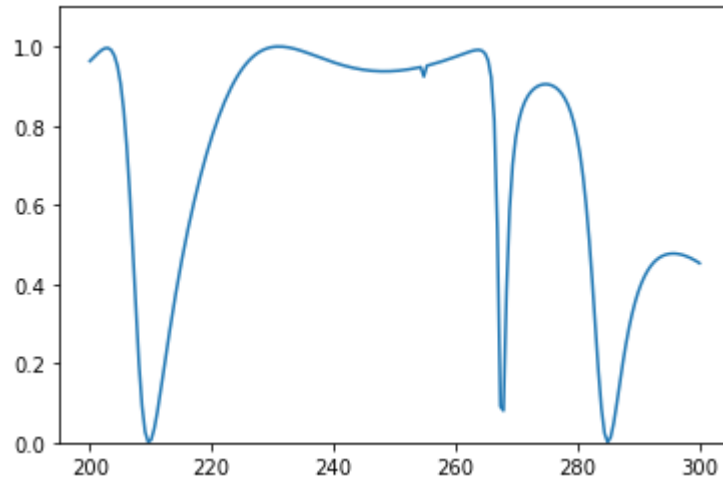


Predicted spectrum:

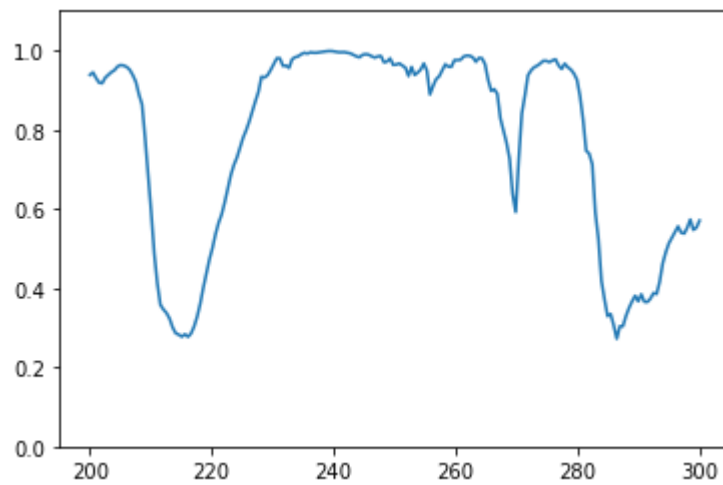


Test 107

True spectrum:



Predicted spectrum:



Test 108

True spectrum:

