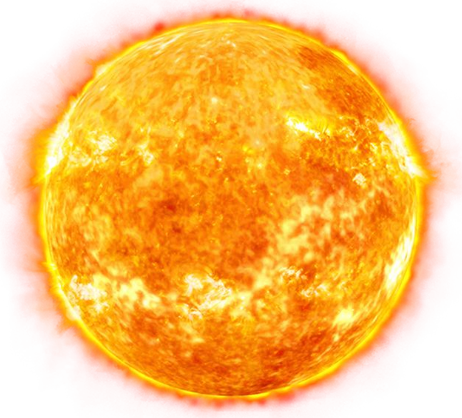
1 2 9 0 FACULDADE DE CIÊNCIAS E TECNOLOGIA UNIVERSIDADE D COIMBRA

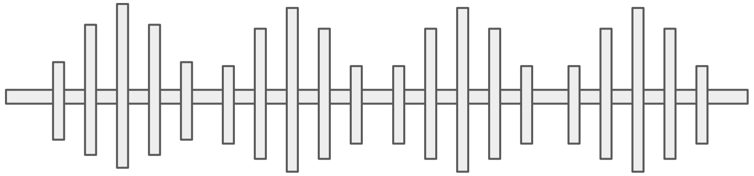
MEF/MAIE

Advanced Techniques in Data Analysis

Professor José Ricardo Gonçalo

Small Separations in Red Giants Stars: quest for late-evolution signatures using Asteroseismology

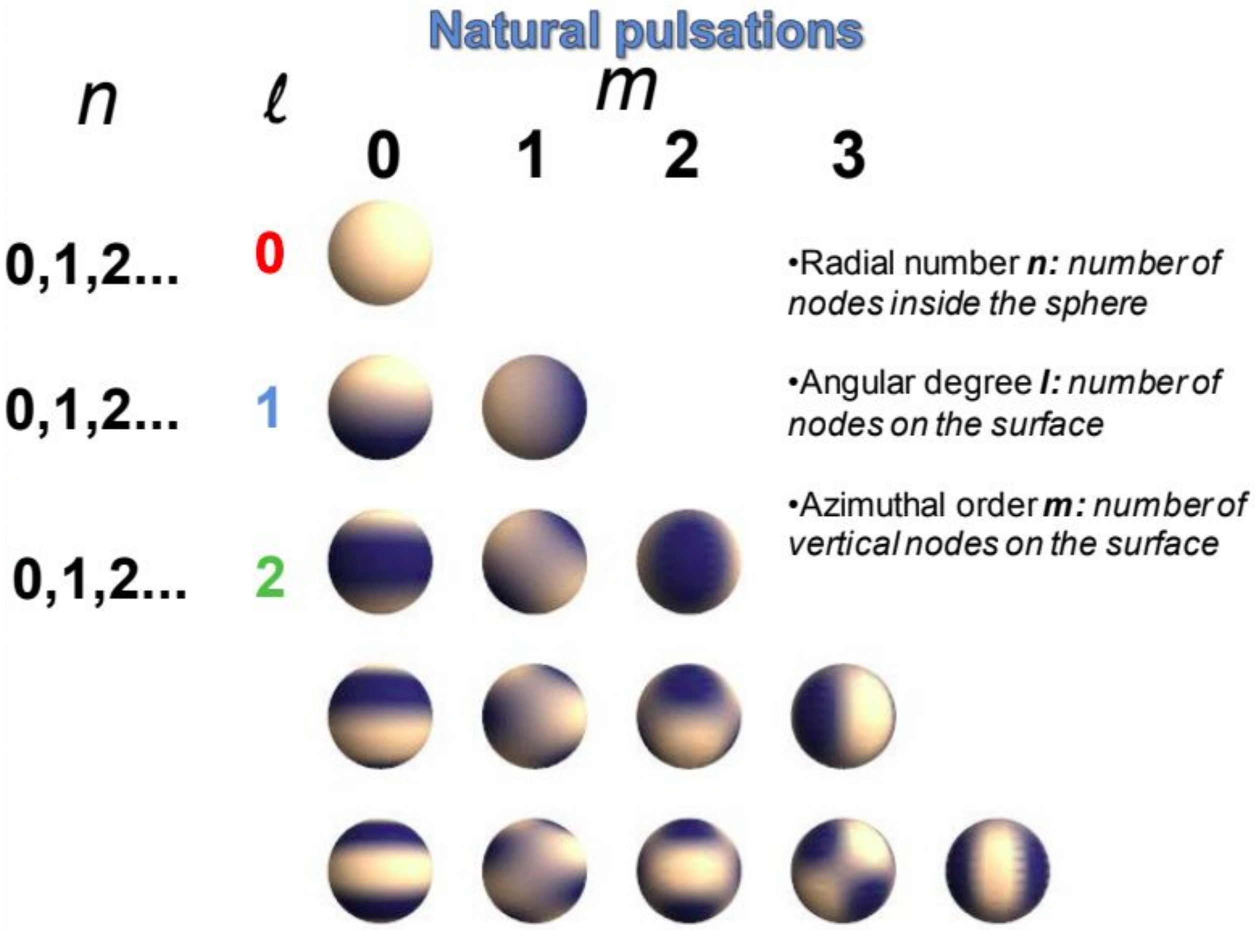


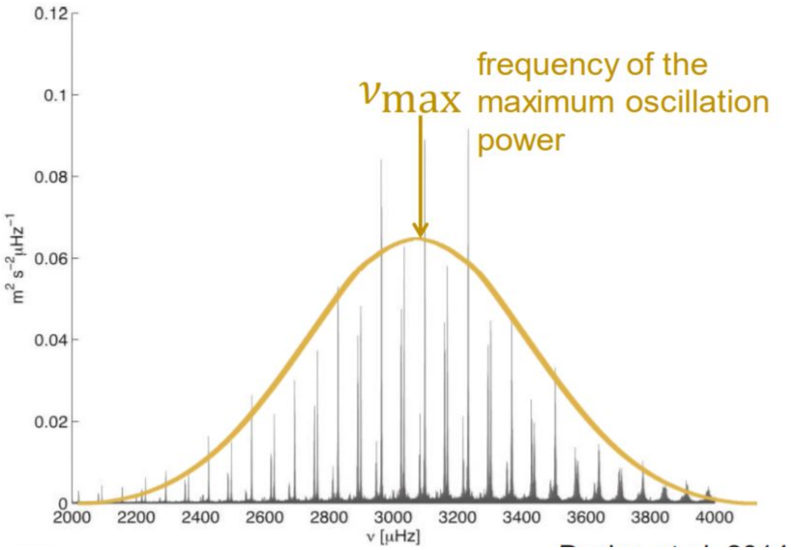


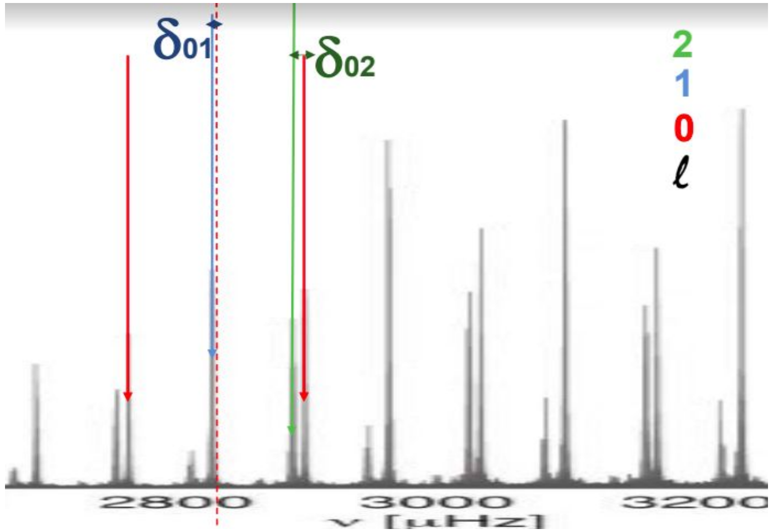
Maria Inês Ferreira, 2018296679
Simão Pedro Neto Sousa, 2017253022

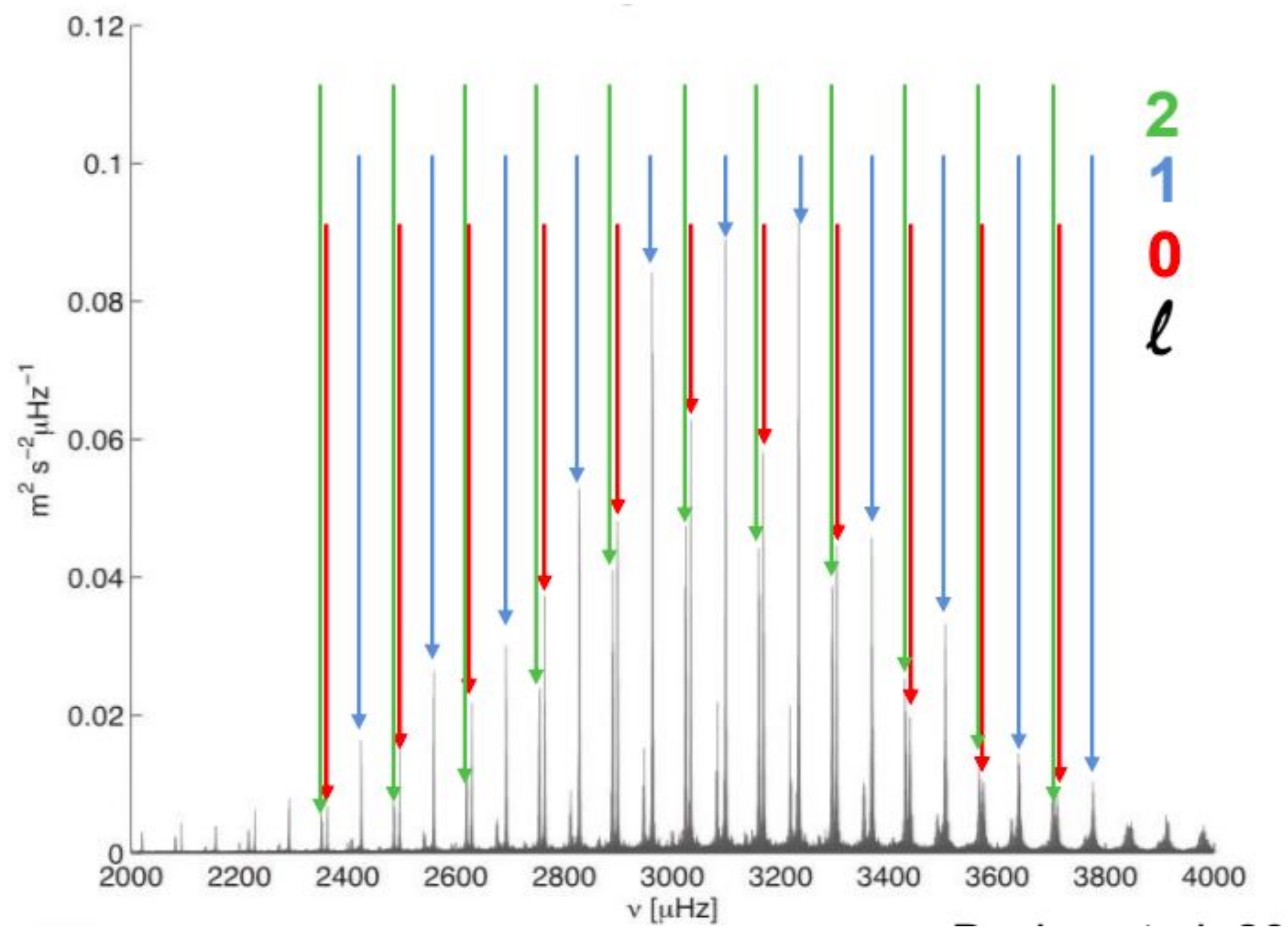
Introduction

Asteroseismology

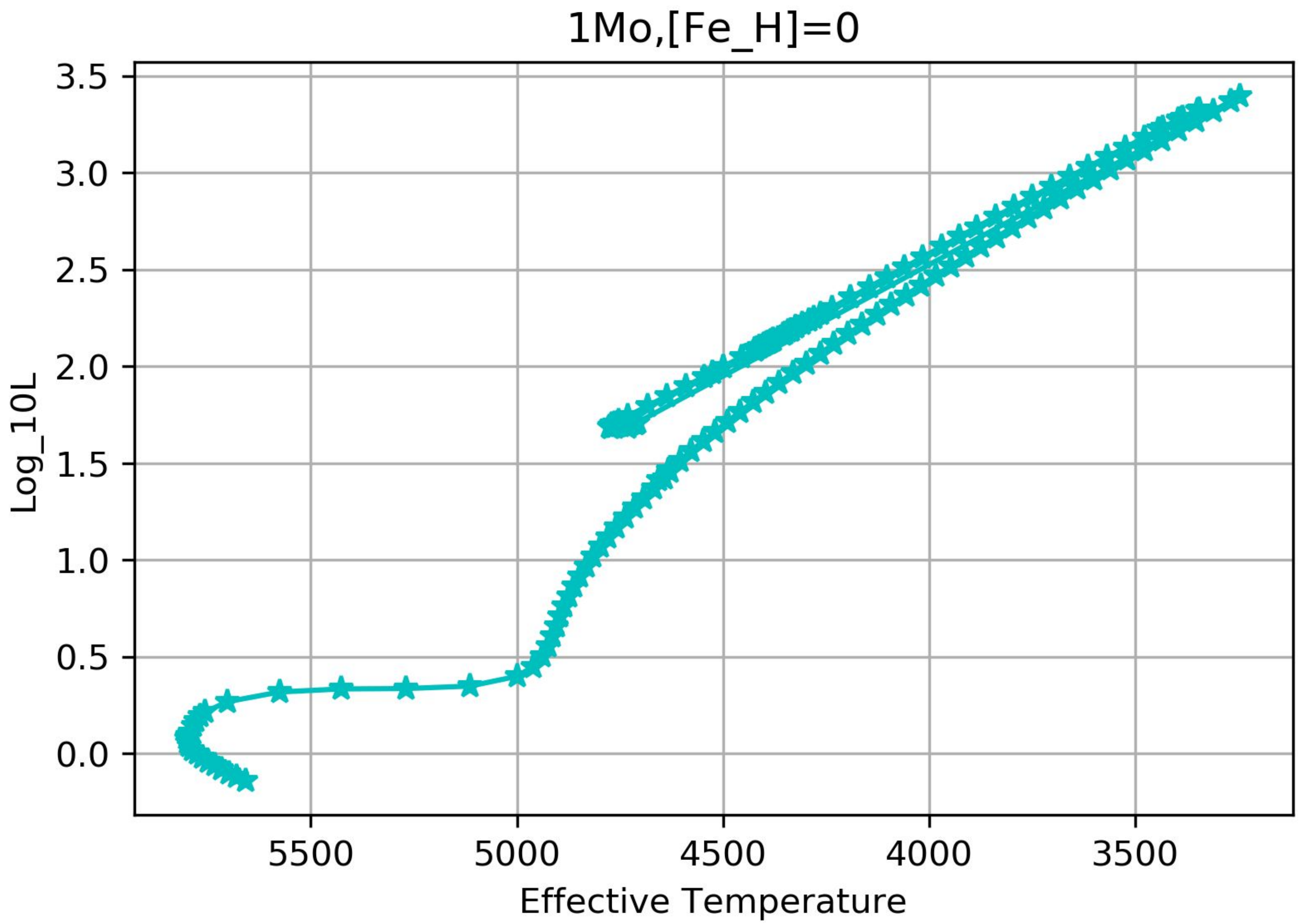



ν_{max}
frequency of the maximum oscillation power
$m^2 s^{-2} \mu Hz^{-1}$
ν [μHz]


δ_{01}
δ_{02}
2
1
0
ℓ
2800
3000
3200
ν [μHz]


2
1
0
ℓ
$m^2 s^{-2} \mu Hz^{-1}$
ν [μHz]

Hertzsprung-Russell Diagram



Data used

A library of frequencies, amplitudes, and lifetimes of more than 250,000 individual l=0 to 3 oscillations modes of 6,179 red giants from APOKASC sample (Pinsonneault et al. 2018) - KALLINGER SAMPLE

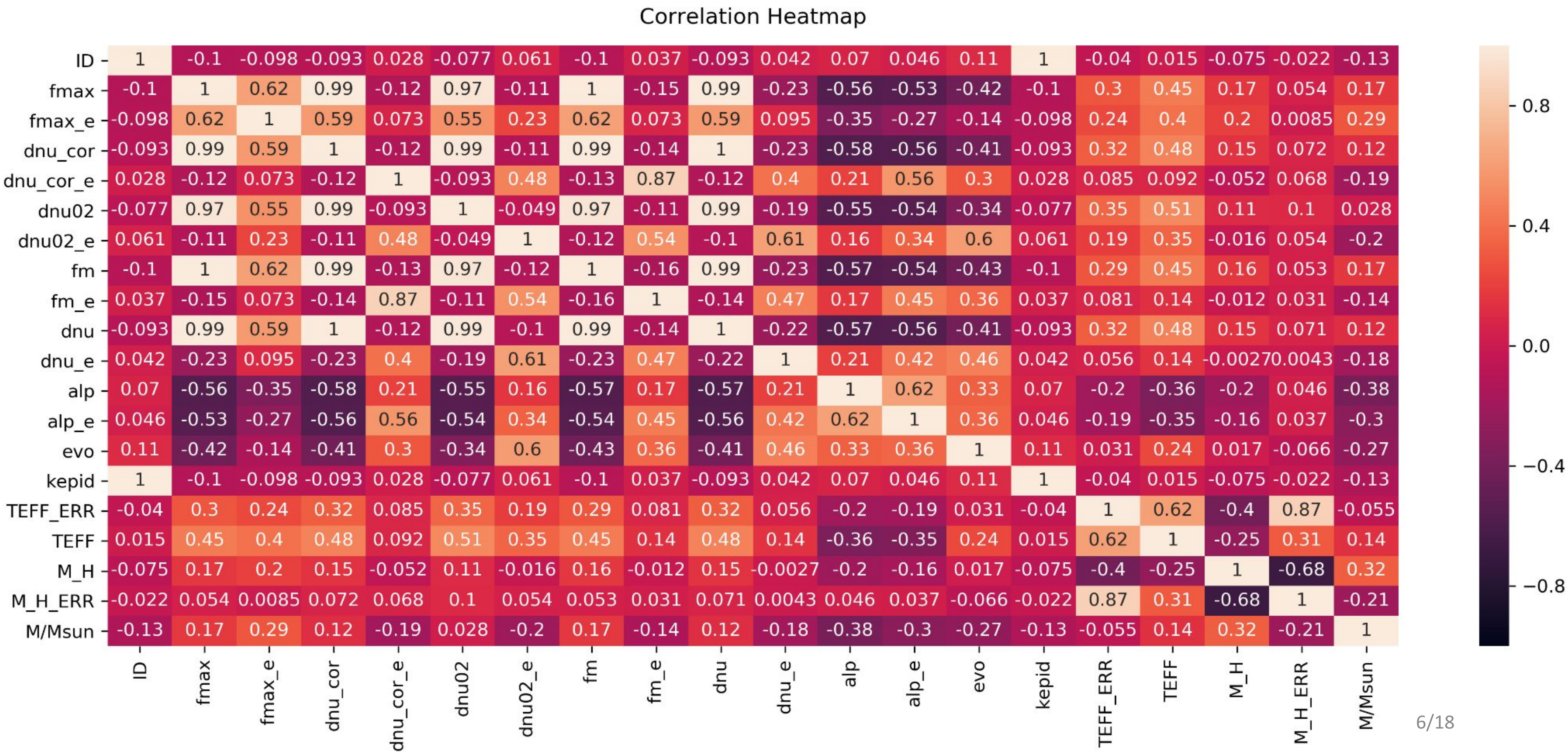
- *fmax*: The frequency of the maximum oscillation power in microHz;
- *dnu*, *dnu02*: The large and small frequency separation determined in the central three radial orders around fmax. All parameters are in microHz;
- *dnu_cor*: Curvature-corrected large separation in microHz;
- *evo*: Evolutionary stage of the star determined from the phase shift of the central radial mode (Kallinger et al. 2012) with the following code: 0 - RGB star, 1 - RC star, 2 - secondary clump star, and 3 - AGB star.

KEPLER SAMPLE:

- *TEFF*: effective temperature of the star (in Kelvin);
- *M_H:* metallicity of the star;
- *M/Msun*: stellar mass in sollar units.

Crossmatch: Sample with 6152 stars.

Correlation Heatmap



Machine Learning Code

```
from numpy import loadtxt
from keras.models import Sequential
from keras.layers import Dense
from matplotlib import pyplot
import pandas as pd
from sklearn.model_selection import train_test_split
import time
import seaborn as sns
import matplotlib.pyplot as plt

```

```
# get the start time-----------------------------------------------------------#
st = time.time()

# load the dataset-------------------------------------------------------------#
df = pd.read_csv('Summ+KeplerM')
#print(df)

#Choose the RGB and RC stars---------------------------------------------------#
stars = df.loc[(df['evo']==0) | (df['evo']==1) ]
model_name = 'Stars_Classifier'
```

```python

#Split in input 'X' and in output 'Y'--------------------------------------#
X = stars[['fmax', 'dnu_cor','dnu02','dnu','TEFF','M_H','M/Msun']]
y = stars[['evo']]

#print(X)
#print(y)


#Correlation heatmap-------------------------------------------------------#
sns.heatmap(stars.corr())
plt.figure(figsize=(16, 6))
heatmap = sns.heatmap(stars.corr(), vmin=-1, vmax=1, annot=True)
heatmap.set_title('Correlation Heatmap', fontdict={'fontsize':12}, pad=12)

```

```python
# split into train, test sets--------------------------------------------------#
train_ratio = 0.80
validation_ratio = 0.10
test_ratio = 0.10
trainX, testX, trainY, testY = train_test_split(X, y, test_size= 1 - train_ratio)
valX, testX, valY, testY = train_test_split(testX, testY, test_size=test_ratio/(test_ratio + validation_ratio))
```

```
#Model-----------------------------------------------------------------------#
    #define the keras model---------------------------------------------------#
model = Sequential()
model.add(Dense(10, input_dim=(7),  kernel_initializer='uniform', activation='relu'))
model.add(Dense(6,  kernel_initializer='uniform', activation='relu'))
model.add(Dense(1,  kernel_initializer='uniform', activation='sigmoid'))

model.summary()

    #compile the keras model-------------------------------------------------#
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

    #fit the keras model on the dataset-------------------------------------#
history=model.fit(trainX, trainY, epochs=900, batch_size=50,validation_data=(valX, valY))
```

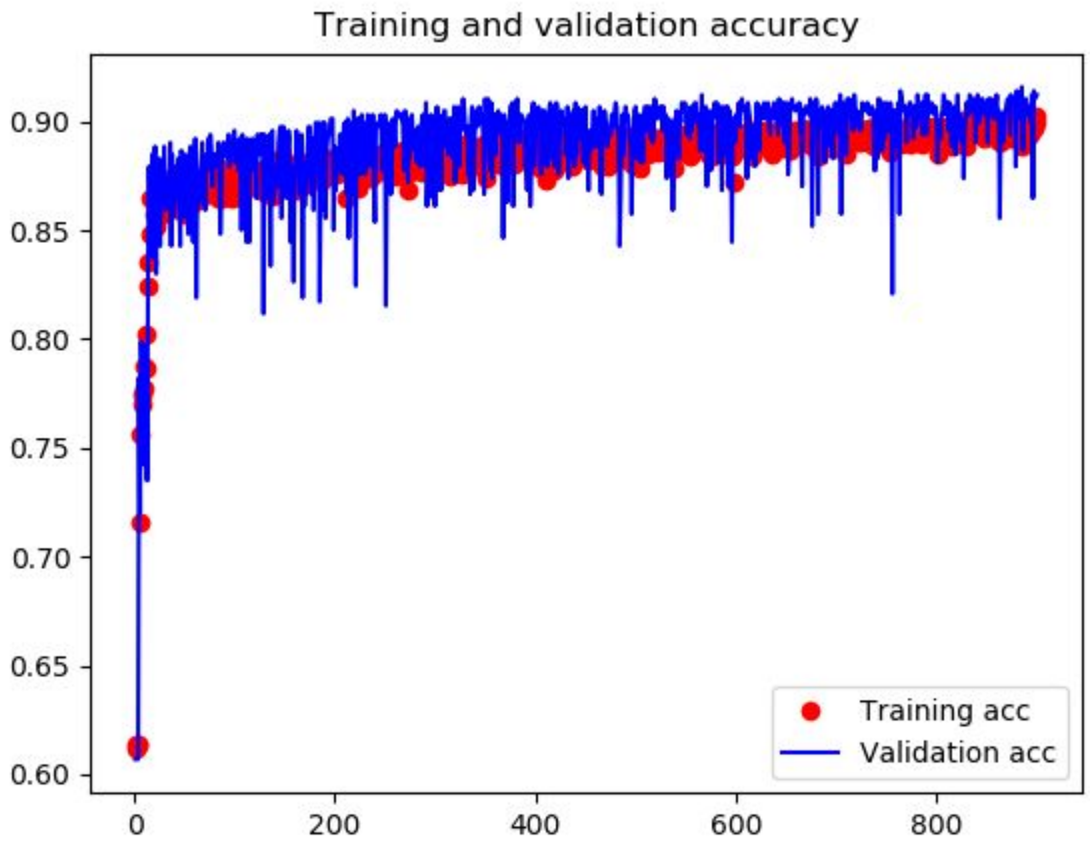
```
#Obtaining the best model values-----------------------------------#
hist_stars=pd.DataFrame(history.history)
hist_stars['epoch'] = hist_stars.index + 1
cols = list(hist_stars.columns)
cols = [cols[-1]] + cols[:-1]
hist_stars= hist_stars[cols]
#hist_stars.to_csv('/' + 'history_stars_' + model_name + '.csv')
hist_stars.head()

values_of_best_model = hist_stars[hist_stars.val_loss == hist_stars.val_loss.min()]
print('Best model', values_of_best_model)
```

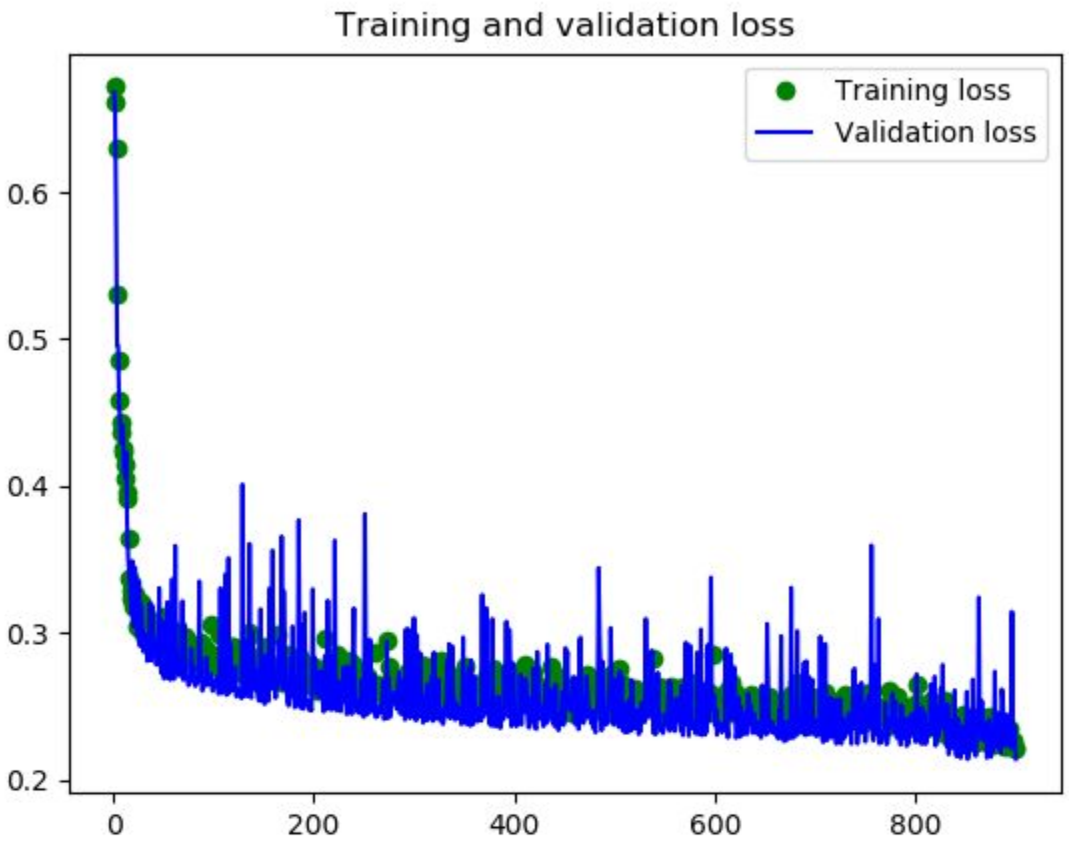
```python
#Validation------------------------------------------------------------#
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(1, len(acc) + 1)

plt.plot(epochs, acc, 'ro', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()
plt.show()


plt.plot(epochs, loss, 'go', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()

plt.show()
```

```python
#Model testing--------------------------------------------------------------#
test_loss, test_acc = model.evaluate(testX, testY)
print()
print('Test Accuracy:', test_acc)
```

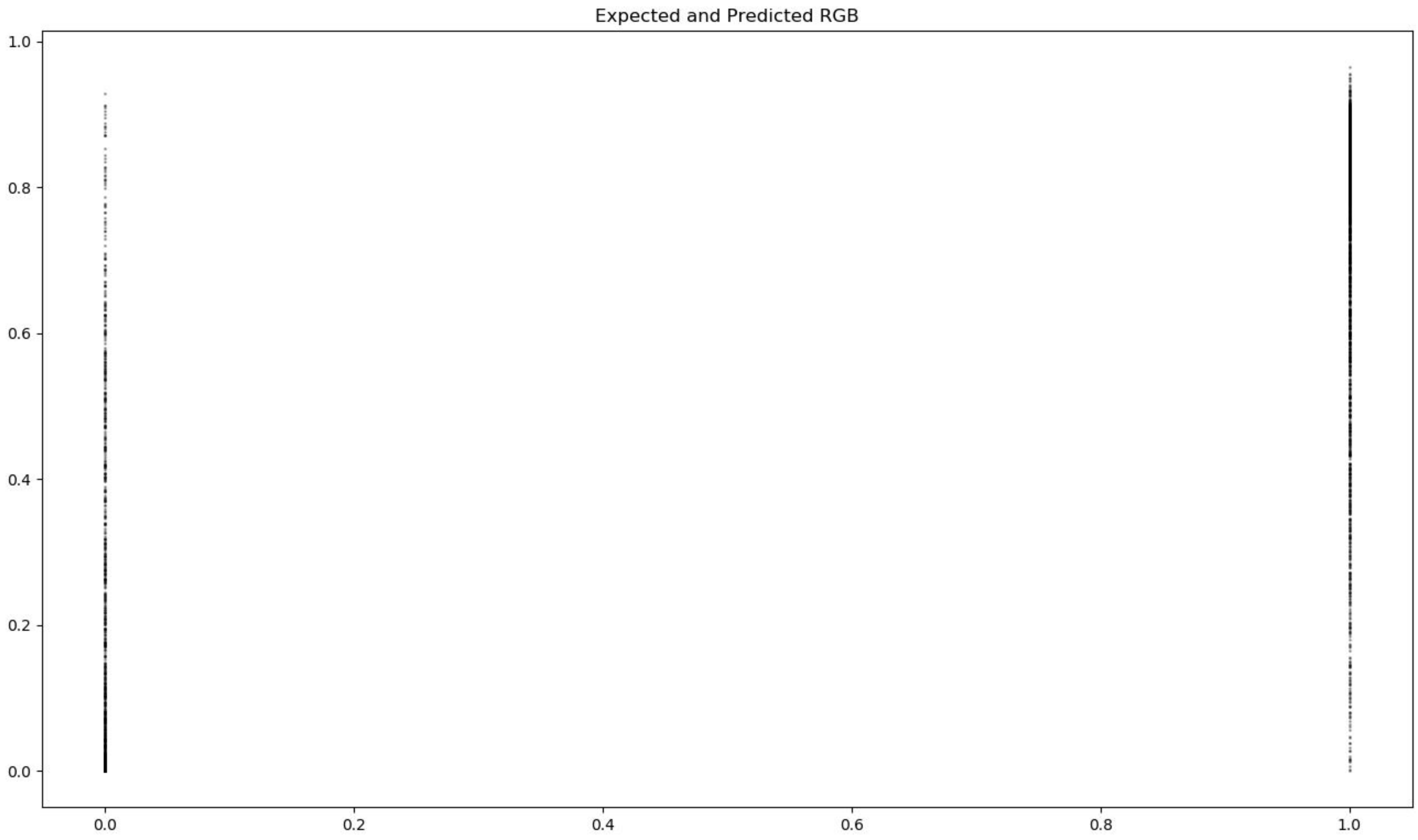
```python
#Predictions---------------------------------------------------------------#
y_pred = model.predict(X)
#print(y_pred)

plt.scatter(y,y_pred, c='k', s=1, alpha=0.3)
plt.title('Expected and Predicted RGB')
plt.show()


# get the end time---------------------------------------------------------#
et = time.time()

# get the execution time---------------------------------------------------#
elapsed_time = et - st
print('Execution time:', elapsed_time, 'seconds')
```

Results/Conclusion





```
Epoch 900/900
88/88 [==============================] - 0s 792us/step - loss: 0.2219 - accuracy: 0.8999 - val_loss: 0.2139 - val_accuracy: 0.9122
Best model      epoch      loss  accuracy  val_loss  val_accuracy
851    852  0.238022   0.89719  0.213854       0.91042
18/18 [==============================] - 0s 794us/step - loss: 0.2108 - accuracy: 0.8996

Test Accuracy: 0.8996350169181824
18/18 [==============================] - 0s 733us/step
Execution time: 259.887921333313 seconds
maria@maria-Creator-M16-A12UC:~/Documents/TAAD/Projeto$
```


Expected and Predicted RGB
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

The End
+
Questions