2018-07-17_TableTestingNotebook_Clean

July 17, 2018

1 Initialise

In [1]:
```
#! /usr/bin/env python

'''
Trains 7D QuaLiKiz-NN with a single output (efeETG_GB)
'''

from __future__ import print_function

import keras
from keras.models import Sequential, Model
from keras.layers import Dense, Dropout, BatchNormalization
from keras.optimizers import RMSprop, adam, Adam
from keras.initializers import TruncatedNormal
from keras import regularizers
from keras import backend as K
import pandas
import numpy
import sys
import os

import matplotlib.pyplot as plt
from matplotlib.colors import LogNorm

from copy import deepcopy
from keras.models import load_model
```

```
Using TensorFlow backend.
```

In [2]:
```
# Define new Metric: rmse = Root Mean Square Error
def rmse(y_true, y_pred):
    return K.sqrt(K.mean(K.square( y_true-y_pred )))

# Gets the current file name. Useful for procedurally generating output/log files.
file_name =  os.path.basename(sys.argv[0][:-3])
```

```
# Define neural network parameters
batch_size = 10
#num_classes = 1
epochs = 100

# Load Data (which is in HDF5 or .h5 format)
store = pandas.HDFStore("../unstable_training_gen2_7D_nions0_flat_filter7.h5")
target_df = store['efeETG_GB'].to_frame()  # This one is relatively easy to train
input_df = store['input']
```

In [3]:
```
# Puts inputs and outputs in the same pandas dataframe.
# Also only keeps overlapping entries.
joined_dataFrame = target_df.join(input_df)

# Make a copy of joined_dataFrame for late use
joined_dataFrame_original = deepcopy(joined_dataFrame)

# Normalize data by standard deviation and mean-centering the data
joined_dataFrame['efeETG_GB'] = (joined_dataFrame['efeETG_GB'] - joined_dataFrame['efeET
joined_dataFrame['Ati'] = (joined_dataFrame['Ati'] - joined_dataFrame['Ati'].mean()) / j
joined_dataFrame['Ate'] = (joined_dataFrame['Ate'] - joined_dataFrame['Ate'].mean()) / j
joined_dataFrame['An'] = (joined_dataFrame['An'] - joined_dataFrame['An'].mean()) / join
joined_dataFrame['qx'] = (joined_dataFrame['qx'] - joined_dataFrame['qx'].mean()) / join
joined_dataFrame['smag'] = (joined_dataFrame['smag'] - joined_dataFrame['smag'].mean())
joined_dataFrame['x'] = (joined_dataFrame['x'] - joined_dataFrame['x'].mean()) / joined_
joined_dataFrame['Ti_Te'] = (joined_dataFrame['Ti_Te'] - joined_dataFrame['Ti_Te'].mean(

# Shuffles dataset
shuffled_joined_dataFrame = joined_dataFrame.reindex(numpy.random.permutation(
                                                    joined_dataFrame.index))

# Creates a pandas dataframe for the outputs
shuffled_clean_output_df = shuffled_joined_dataFrame['efeETG_GB']

# Creates a pandas dataframe for the inputs
shuffled_clean_input_df = shuffled_joined_dataFrame.drop('efeETG_GB', axis=1)

# Creates training dataset (90% of total data) for outputs
y_train = shuffled_clean_output_df.iloc[:int(
    numpy.round(len(shuffled_clean_output_df)*0.9))]

# Creates training dataset (90% of total data) for inputs
x_train = shuffled_clean_input_df.iloc[:int(
    numpy.round(len(shuffled_clean_input_df)*0.9))]

# Creates testing dataset (10% of total data) for outputs
y_test = shuffled_clean_output_df.iloc[int(
```

```
    numpy.round(len(shuffled_clean_output_df)*0.9)):]

# Creates testing dataset (10% of total data) for inputs
x_test = shuffled_clean_input_df.iloc[int(
    numpy.round(len(shuffled_clean_input_df)*0.9)):]

# Deletes pandas dataframes that are no longer needed
del target_df, input_df

# Closes the HDFStore. This is good practice.
store.close()
```

DEBUGGING

In [4]:
```
joined_dataFrame.describe(include='all')
```

Out[4]:
```
          efeETG_GB           Ati           Ate            An            qx  \
count  638880.000000  6.388800e+05  6.388800e+05  6.388800e+05  6.388800e+05
mean        0.000011  3.921781e-13 -3.483536e-13  5.831676e-14  4.388160e-14
std         1.000114  1.000000e+00  1.000000e+00  1.000000e+00  1.000000e+00
min        -1.136391 -1.848058e+00 -2.151526e+00 -4.177720e+00 -8.735150e-01
25%        -0.802423 -6.589535e-01 -6.642361e-01 -6.349019e-01 -6.749461e-01
50%        -0.301873 -1.493374e-01 -1.684729e-01 -4.443218e-02 -4.385544e-01
75%         0.557895  6.150866e-01  4.925446e-01  5.460375e-01  1.524246e-01
max         3.965936  2.908359e+00  1.814580e+00  2.317447e+00  2.516341e+00

               smag             x         Ti_Te
count  6.388800e+05  6.388800e+05  6.388800e+05
mean   2.038081e-13 -3.500088e-14  1.390060e-13
std    1.000000e+00  1.000000e+00  1.000000e+00
min   -1.264894e+00 -1.375825e+00 -1.666559e+00
25%   -3.146967e-01 -9.567490e-01 -9.260135e-01
50%   -5.555202e-02 -1.185980e-01 -6.698017e-02
75%    4.627374e-01  7.195530e-01  4.217800e-01
max    3.918000e+00  1.767242e+00  1.665897e+00
```

In [5]:
```
joined_dataFrame_original.describe(include='all')
```

Out[5]:
```
          efeETG_GB           Ati            Ate             An  \
count  638880.000000  6.388800e+05  638880.000000  638880.000000
mean       26.038572  5.439559e+00       8.509738       2.075249
std        22.899494  2.943392e+00       3.025638       1.693567
min         0.015788  1.000000e-14       2.000000      -5.000000
25%         7.663502  3.500000e+00       6.500000       1.000000
50%        19.125839  5.000000e+00       8.000000       2.000000
75%        38.814079  7.250000e+00      10.000000       3.000000
max       116.856499  1.400000e+01      14.000000       6.000000
```

```
                  qx           smag              x          Ti_Te
count  638880.000000  638880.000000  638880.000000  638880.000000
mean        4.355203       0.464310       0.483960       1.375224
std         4.230269       1.157654       0.286345       0.675178
min         0.660000      -1.000000       0.090000       0.250000
25%         1.500000       0.100000       0.210000       0.750000
50%         2.500000       0.400000       0.450000       1.330000
75%         5.000000       1.000000       0.690000       1.660000
max        15.000000       5.000000       0.990000       2.500000
```

2 Load Model

```
In [6]: new_model = load_model('../Saved-Networks/2018-07-16_7D_Run0050d.h5', custom_objects={'r
```

Model information

```
In [7]: new_model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 30)                240
_________________________________________________________________
dense_2 (Dense)              (None, 30)                930
_________________________________________________________________
dense_3 (Dense)              (None, 30)                930
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 31
=================================================================
Total params: 2,131
Trainable params: 2,131
Non-trainable params: 0
_________________________________________________________________
```

```
In [8]: new_model.get_weights()
```

```
Out[8]: [array([[-2.87334411e-03,  2.41862200e-02, -4.12066132e-02,
                 -1.34515027e-02,  9.28834919e-03,  1.93074974e-03,
                 -1.37387365e-02,  4.32844367e-03, -5.89680718e-03,
                  2.03450094e-03, -3.67641784e-02, -1.36629390e-02,
                  3.24408151e-03, -1.61009077e-02,  1.55328093e-02,
                 -5.37062017e-03,  2.79186517e-02,  1.23187387e-02,
                  5.63374674e-03,  1.52084474e-02,  5.09614870e-03,
                  2.81141233e-02,  8.71234760e-03,  7.75729641e-02,
                  2.03094129e-02, -3.85158360e-02,  1.47108687e-02,
                 -7.39949122e-02, -5.51870326e-03,  3.68888886e-03],
                [ 1.64472997e-01,  1.06675661e+00,  4.09337372e-01,
```

```
  8.65022659e-01,  1.29730806e-01, -7.61756837e-01,
 -9.31763232e-01,  1.19161308e+00,  1.50285649e+00,
 -7.73032486e-01,  6.38422847e-01, -6.03104711e-01,
  6.45129800e-01,  1.58113435e-01, -2.42174134e-01,
 -8.46612215e-01,  3.21146280e-01,  1.58368856e-01,
  1.67071044e-01,  1.47525787e+00,  7.64938593e-01,
  7.13677943e-01, -4.93443429e-01, -1.01022875e+00,
 -4.76346344e-01, -1.24295878e+00,  3.06886077e-01,
  6.21331036e-01,  1.07051027e+00, -5.47027647e-01],
[-1.26984343e-01,  2.56848097e-01,  4.81885880e-01,
  1.57591426e+00, -1.03540972e-01,  2.91106999e-01,
 -5.92427194e-01,  1.27917826e+00,  6.44989312e-02,
  9.25727010e-01, -8.36083233e-01, -7.87467599e-01,
 -9.92753729e-02, -8.13657269e-02, -9.42181349e-01,
  9.24308956e-01, -1.13008849e-01,  6.05998300e-02,
 -3.28860395e-02,  8.78981769e-01,  1.99197054e-01,
 -6.40956983e-02, -6.93166628e-02,  3.79543185e-01,
 -2.18214363e-01,  1.14177436e-01, -2.74927039e-02,
  5.89777231e-01, -2.79116035e-01, -1.06637247e-01],
[ 1.07973552e+00, -8.42819810e-02, -1.88043416e-01,
 -2.11450551e-02,  3.93051952e-02,  2.35682040e-01,
 -2.10233974e+00, -8.72754380e-02,  7.62421787e-01,
  2.21149647e-03, -1.36668772e-01,  1.77159071e-01,
  3.52035493e-01,  2.74182153e+00,  2.96959970e-02,
 -1.07304677e-02, -2.20328167e-01, -7.21435947e-03,
  4.38721991e+00,  4.66187000e-02,  1.16760099e+00,
  3.07604037e-02, -4.98672314e-02, -7.61934817e-01,
 -2.98055053e+00,  3.87072414e-02, -7.17803836e-02,
  6.58552229e-01, -1.91831887e+00,  5.65108806e-02],
[ 6.08073235e-01, -3.76589775e-01,  2.87275124e+00,
 -6.73020244e-01, -1.66408792e-01,  1.37351084e+00,
  5.25229275e-01, -2.23677731e+00, -8.19951415e-01,
  2.36129791e-01, -8.15693140e-02,  1.44496679e+00,
  2.67539477e+00,  8.03758353e-02, -9.93818566e-02,
 -4.15350758e-02, -1.82769012e+00, -3.96165419e+00,
 -1.80927590e-01, -1.30084157e+00, -2.35289550e+00,
 -8.17715347e-01,  1.19526160e+00,  1.30351052e-01,
  9.96745944e-01, -3.39319140e-01, -2.72491038e-01,
 -1.79533139e-01,  2.65228953e-02,  5.24463467e-02],
[ 1.14377797e+00,  2.85877943e-01,  1.77953333e-01,
  1.18974485e-01,  5.37990779e-02, -8.43612134e-01,
  8.94020647e-02,  4.97900903e-01, -7.80252814e-02,
  2.82550987e-04,  2.81605870e-01, -4.21142876e-01,
  7.70211071e-02, -3.42372954e-01, -1.49332479e-01,
  7.28331134e-02, -3.52074802e-02,  2.29843576e-02,
  4.73428965e-02,  1.89843565e-01,  1.07679296e+00,
 -6.45065248e-01, -3.96832526e-01,  3.06480646e-01,
 -3.19016688e-02, -3.34487021e-01, -3.07155419e-02,
```

```
         1.36558008e+00, -5.35130322e-01,  2.66407371e+00],
       [-7.72894472e-02,  1.41471922e-01,  2.96587348e-01,
        -5.52019954e-01, -2.56610203e+00, -2.35952303e-01,
        -5.52329957e-01, -8.95297885e-01,  8.43211770e-01,
        -9.04196203e-02,  3.44702363e-01, -3.70212466e-01,
         2.44947225e-01,  2.09329754e-01, -3.49108160e-01,
         4.74247448e-02,  1.77391648e-01,  1.95224911e-01,
         6.32827580e-02,  2.89560914e-01,  1.80957228e-01,
         6.61769137e-02, -1.29160464e+00, -3.71771395e-01,
         3.58647145e-02, -1.02815568e+00,  1.39717686e+00,
         4.81063813e-01,  1.07778706e-01, -1.57336902e-03]], dtype=float32),
array([ 0.34689942, -0.8294998 , -0.7786168 ,  2.6040158 , -3.9773817 ,
       -2.8039398 ,  0.98027277, -5.729468  , -3.0074127 , -0.24496965,
       -1.7469301 , -0.5725213 ,  0.45806736,  0.5340321 , -0.2777078 ,
       -1.954045  ,  0.6371471 , -0.6917116 ,  2.9314675 ,  1.9802775 ,
        2.2753966 , -1.1946698 , -1.9209228 ,  1.0282121 , -2.9778345 ,
        1.7863988 ,  0.94776016, -1.3170915 , -2.2026274 ,  3.6746933 ],
      dtype=float32),
array([[-9.56088826e-02, -8.90816301e-02,  8.21453780e-02,
         6.18751168e-01, -8.61653745e-01,  4.09828797e-02,
         2.60102344e+00,  9.59904969e-01,  1.02338552e-01,
        -1.46506935e-01,  3.26990128e-01, -5.70332170e-01,
         6.67673349e-01, -1.00302942e-01, -1.58234984e-01,
        -2.39643037e-01,  5.21264672e-01,  1.40756056e-01,
        -2.06611361e-02,  7.71015227e-01,  5.03138490e-02,
         7.17258394e-01,  1.47064674e+00, -4.19060469e-01,
         3.49942833e-01, -2.50777006e-01, -4.01767194e-01,
         1.49125898e+00, -6.21628106e-01, -3.38223320e-03],
       [ 1.30083775e-02, -3.64981331e-02,  8.57302368e-01,
         3.12383652e+00,  1.17775035e+00, -4.78834808e-01,
        -2.76651239e+00, -2.60876894e+00, -6.35414004e-01,
         6.23874903e-01, -7.68072724e-01, -7.98286032e-03,
         4.41038907e-01,  3.29350859e-01, -1.75757587e-01,
        -1.46167529e+00, -1.59989381e+00, -1.04848135e+00,
         8.19393992e-01,  6.91566646e-01,  6.43285751e-01,
         5.81680000e-01,  3.12398291e+00, -1.82766521e+00,
        -1.82343170e-01,  8.18798065e-01,  1.08804262e+00,
        -5.43746948e-01, -9.27818894e-01,  5.84346533e-01],
       [ 7.50721022e-02, -1.28632128e-01, -1.47556829e+00,
        -4.85712975e-01, -7.59585083e-01, -2.87975937e-01,
        -8.56798887e-01,  5.19002751e-02, -1.34575319e+00,
         1.78304088e+00,  2.75639266e-01,  3.27218533e-03,
         2.65250063e+00,  2.56955087e-01,  3.43152070e+00,
        -5.03013015e-01, -4.68117237e-01,  1.11873007e+00,
        -4.16526139e-01, -8.89447093e-01,  5.06492078e-01,
         1.57618141e+00,  1.48800588e+00,  8.74891654e-02,
        -4.39923376e-01,  3.57859403e-01,  1.23689163e+00,
         9.33784008e-01,  4.77428138e-01,  1.15548301e+00],
```

```
[-6.40349269e-01, -7.47442245e-02,  1.61256158e+00,
  4.73847948e-02,  3.25394225e+00, -3.52094650e-01,
  6.85847998e-01, -6.72870204e-02, -2.49217534e+00,
 -7.40516409e-02,  3.53569448e-01, -3.44449115e+00,
 -2.30822802e+00,  4.03165035e-02, -3.37012935e+00,
 -2.21393847e+00, -1.00807130e+00, -1.06027043e+00,
  8.81336987e-01,  4.68849212e-01, -1.16157055e+00,
 -1.61523795e+00,  4.00721461e-01, -4.71393764e-01,
  5.36516488e-01, -1.22482665e-01,  8.04770947e-01,
 -6.70740187e-01, -7.76790977e-01, -1.46550608e+00],
[-1.92035806e+00, -6.38330460e-01,  2.89369166e-01,
  1.79218781e+00, -2.81838685e-01, -4.66958553e-01,
  4.67858911e-01,  4.74771470e-01, -6.47370636e-01,
  8.68369818e-01, -1.11159849e+00, -8.92182589e-02,
 -1.99738100e-01, -6.18071675e-01, -2.08809137e+00,
  2.56751161e-02, -3.22313607e-01,  2.84474730e-01,
  2.68895340e+00,  1.00111353e+00, -1.86129129e+00,
 -2.46577573e+00, -4.88571435e-01, -6.23136401e-01,
  1.63017094e+00,  9.78072345e-01,  9.39182520e-01,
 -1.68938005e+00, -2.76712745e-01, -2.33608770e+00],
[-1.34651005e-01, -2.20332384e+00, -2.46430135e+00,
 -9.42645550e-01, -4.07170773e+00,  2.15569045e-02,
 -4.24010229e+00,  3.02933598e+00, -3.54516554e+00,
 -2.61719465e+00, -2.05363774e+00, -3.22237229e+00,
  2.63143444e+00, -3.14248264e-01, -1.16520166e+00,
 -1.68373013e+00,  5.83523035e-01,  5.93597591e-01,
  1.63863528e+00,  7.71881759e-01,  1.88549459e-01,
 -3.49248618e-01,  7.96904266e-01, -4.17949378e-01,
  1.31617332e+00, -1.35776651e+00, -9.93729115e-01,
  1.59795105e+00,  8.28111887e-01, -2.88734645e-01],
[ 3.71874534e-02,  2.27622017e-02, -9.67191830e-02,
 -2.70745568e-02, -3.93410015e+00, -7.80938789e-02,
 -8.91433835e-01,  7.22864866e-01, -1.99037150e-01,
 -1.51655829e+00,  3.79414707e-02,  6.90502971e-02,
  8.13090682e-01, -3.15564632e-01,  2.80015916e-01,
  1.36617129e-03, -8.48497450e-02,  6.39994323e-01,
  6.75271526e-02,  4.66475815e-01, -4.63004917e-01,
 -2.52495646e+00,  2.37781501e+00,  5.76625057e-02,
 -8.20251405e-02, -5.24489641e-01,  6.99236035e-01,
  1.61951792e+00,  2.41148338e-01,  3.57549787e-01],
[ 1.03836305e-01, -3.20913225e-01,  2.41835046e+00,
  4.70331699e-01, -8.16426203e-02,  2.97016148e-02,
 -9.69633043e-01,  4.66516346e-01,  3.42835076e-02,
 -3.52560252e-01, -5.29685318e-01,  3.29047024e-01,
 -2.31886834e-01, -1.27509117e+00, -1.72368979e+00,
  1.12560892e+00, -1.49966836e+00,  7.16897070e-01,
 -1.56129861e+00,  1.43887877e+00, -2.58361876e-01,
 -3.91844928e-01, -1.16439235e+00, -8.82139802e-01,
```

```
 -5.21308184e-01,  1.88114583e-01, -1.58978999e-01,
 -1.74156451e+00,  1.65574223e-01, -4.67103034e-01],
[ 4.10402596e-01,  3.46590519e-01,  1.82872340e-02,
  1.95124304e+00, -5.51869869e-01, -5.33378780e-01,
  2.99491906e+00, -7.30327666e-01,  2.64450282e-01,
 -4.01869822e+00,  1.44877940e-01, -8.61325935e-02,
  7.35875428e-01,  2.56053984e-01,  5.71089566e-01,
  2.70532697e-01, -4.87273604e-01, -1.21410981e-01,
 -1.34351838e+00, -1.20706701e+00, -5.75991631e-01,
 -5.62376440e-01, -1.01900208e+00,  8.89969394e-02,
 -1.41637135e+00,  7.39647269e-01,  2.12025619e+00,
  9.31608796e-01,  4.99353498e-01,  1.51020336e+00],
[-3.40118259e-01,  4.92769241e-01, -8.15064371e-01,
  6.31164536e-02,  1.29225284e-01, -8.89730871e-01,
  2.60759830e+00, -6.89973295e-01,  1.00891256e+00,
  8.51762772e-01, -2.98978657e-01, -3.15568089e-01,
  1.21392250e+00,  9.94270504e-01,  1.80865252e+00,
 -2.77786565e+00, -1.00268471e+00, -1.12629747e+00,
  1.91109943e+00,  3.33386600e-01,  6.63721859e-01,
 -2.19627452e+00, -1.11086679e+00, -7.81869814e-02,
  1.29811251e+00, -3.92254084e-01, -2.94842303e-01,
  8.38302076e-02,  1.67205024e+00, -2.65423525e-02],
[ 2.12919533e-01, -2.67393023e-01,  7.79523373e-01,
 -1.82084155e+00, -1.26492321e+00,  2.40282130e+00,
  8.15882921e-01, -5.93437314e-01,  1.08006358e-01,
 -6.35934532e-01,  1.56625640e+00,  1.19173720e-01,
 -3.02737147e-01, -6.19449973e-01, -5.08872211e-01,
  7.63627827e-01, -6.17812157e-01,  2.77546853e-01,
 -5.86510360e-01,  8.25828195e-01,  2.29618937e-01,
  2.60212350e+00,  1.62876070e+00,  1.21785390e+00,
 -3.40636700e-01, -8.98877323e-01,  6.91743255e-01,
 -9.28415716e-01, -5.33600688e-01, -7.44937479e-01],
[-1.07184388e-01,  4.65530939e-02, -3.66658300e-01,
 -1.54619738e-01,  3.18340182e-01, -4.25690621e-01,
  2.27895841e-01,  5.08025214e-02, -1.13380647e+00,
  3.64373541e+00, -9.13355708e-01,  1.20791113e+00,
  1.77263820e+00,  7.12797105e-01,  1.48756576e+00,
 -8.06765020e-01,  5.35336494e-01, -5.86311817e-01,
  2.44192053e-02,  4.12084907e-01,  1.36575317e+00,
 -1.72833896e+00,  9.42294359e-01,  1.06982946e+00,
  4.71235931e-01, -1.63828239e-01,  9.31946859e-02,
  1.28092742e+00,  4.15004849e-01,  4.58853900e-01],
[ 4.04715478e-01,  1.06932366e+00,  5.96683741e-01,
  1.46096721e-01, -4.18291509e-01, -1.96857795e-01,
 -1.35835320e-01,  9.02346849e-01, -6.32316694e-02,
  5.32167006e+00,  1.81943667e+00, -2.10746861e+00,
  4.32777926e-02, -4.15238827e-01,  5.33068717e-01,
  2.68371081e+00, -1.43701029e+00, -1.42918080e-01,
```

```
-1.59913766e+00,  2.09941149e+00,  1.82327223e+00,
  2.15869561e-01, -7.77506888e-01, -1.12193143e+00,
 -1.15194523e+00,  7.11646616e-01,  9.18790877e-01,
  1.67440593e+00, -8.95461798e-01, -3.12191963e-01],
[ 6.23133406e-02,  1.82104814e+00, -4.91203591e-02,
 -1.84864604e+00,  3.75799052e-02,  1.91773057e-01,
  2.98619461e+00,  1.18601000e+00, -7.27443159e-01,
 -1.61762142e+00, -1.41242985e-02, -3.49650651e-01,
 -6.99132979e-01, -6.49221838e-01,  2.72370696e-01,
  6.13563776e-01,  3.44816625e-01, -6.60827756e-01,
 -8.87079239e-02, -6.07659221e-01,  6.65341765e-02,
  9.80068028e-01, -4.44845104e+00,  1.50446147e-01,
 -3.80244493e-01,  2.37246051e-01, -1.04051960e+00,
  3.30946565e+00,  3.62434506e-01,  6.24094009e-01],
[ 1.01967521e-01,  7.08995759e-01, -1.34918368e+00,
  1.64548302e+00,  6.46066904e-01, -5.03161609e-01,
  8.68333817e-01,  7.19895482e-01,  9.83898759e-01,
 -2.18128943e+00, -5.06498218e-01,  2.58146793e-01,
  4.02658731e-01,  7.14624301e-02,  1.34867072e+00,
  1.06523848e+00, -1.23872757e+00,  8.55393987e-03,
 -4.91953567e-02, -5.65872014e-01,  7.82107353e-01,
 -2.58103430e-01, -1.60688293e+00, -6.07319176e-01,
 -9.72208738e-01,  1.56069174e-01, -1.90119799e-02,
 -7.59285092e-02,  1.48825347e+00,  3.79529476e-01],
[ 2.30039373e-01, -3.64814848e-01, -8.18728924e-01,
  1.05104113e+00,  8.30125093e-01, -2.11568165e+00,
 -5.81741631e-01, -4.46188974e+00, -2.89171748e-03,
 -9.26587820e-01, -4.05826950e+00,  1.47761536e+00,
 -5.43117106e-01,  2.62700510e+00,  3.80403072e-01,
 -1.61649048e+00,  1.74785066e+00, -5.69989145e-01,
  2.55688572e+00,  9.62636471e-01,  1.90704477e+00,
 -9.33011234e-01, -2.30251908e+00,  1.49005461e+00,
  8.91476631e-01, -5.28675556e-01,  4.49843943e-01,
 -1.02463138e+00,  1.39317071e+00, -9.20345247e-01],
[ 7.70952344e-01, -2.60747254e-01,  1.11595714e+00,
 -1.75069022e+00,  1.97260892e+00, -4.27928865e-01,
  9.34609652e-01, -9.46502388e-01, -1.32666421e+00,
  5.99972188e-01, -1.31333899e+00, -3.90489399e-01,
  1.28788829e-01,  4.56267685e-01,  1.22703600e+00,
  4.02187496e-01,  6.33369565e-01, -2.26795411e+00,
 -3.16043943e-01,  1.15912592e+00,  9.64605451e-01,
 -2.92316580e+00,  9.77210104e-02, -8.67976427e-01,
 -5.23234308e-01, -4.88911450e-01, -1.65673220e+00,
 -1.00079930e+00, -3.00528467e-01,  1.69107723e+00],
[-7.60209680e-01, -3.15145701e-01, -3.84923875e-01,
 -2.15068650e+00,  1.65906489e+00, -8.44059229e-01,
 -1.98834729e+00, -2.21376419e-01, -2.50372982e+00,
 -6.23817062e+00, -2.06766510e+00, -7.40499735e-01,
```

```
 -1.22274458e+00,  7.97955096e-01, -3.82861042e+00,
  1.43447554e+00,  8.91156912e-01, -1.20110855e-01,
  2.62804478e-01, -1.36760437e+00,  2.09439516e+00,
 -5.02227545e+00,  7.67590463e-01,  1.47230816e+00,
  1.32451379e+00, -1.87949586e+00,  2.15261722e+00,
 -1.28641427e+00,  2.21706414e+00, -2.21313047e+00],
[ 1.70659661e-01,  4.74177933e+00,  7.57772684e-01,
  1.81566083e+00, -5.65740407e-01,  3.75103831e-01,
  2.16959143e+00, -4.31772500e-01, -5.51272452e-01,
  1.29723978e+00, -1.04961658e+00, -5.90925932e-01,
 -2.72555619e-01, -2.69836307e-01, -3.86114448e-01,
  1.37047231e+00,  3.48808132e-02, -1.97661781e+00,
 -1.36136341e+00, -1.24041069e+00,  5.76286852e-01,
  5.78174639e+00, -3.23438287e+00, -6.90299451e-01,
 -6.38018310e-01,  2.32344294e+00, -7.89111629e-02,
  7.36015224e+00,  5.95712066e-01,  3.06096524e-01],
[ 3.08197707e-01,  1.82545885e-01, -4.45407152e-01,
 -1.39656210e+00, -2.45553657e-01,  8.48465025e-01,
 -1.64258748e-01,  8.75590801e-01,  1.92852899e-01,
  1.12042439e+00,  1.44132471e+00,  3.58710021e-01,
  1.35145962e-01, -7.56051242e-01,  1.00206864e+00,
 -4.32061338e+00, -1.35744244e-01,  4.28415209e-01,
  2.36899805e+00,  2.22379252e-01, -2.19687724e+00,
  5.15040874e-01,  3.63600791e-01, -2.58333039e+00,
  3.48416269e-01,  7.53074586e-01, -5.90055227e-01,
  6.96529925e-01, -4.07685459e-01,  4.67812657e-01],
[ 2.39492461e-01,  3.83538485e-01, -2.73956180e-01,
  8.24886322e-01,  4.55477208e-01, -7.74803162e-02,
 -2.38778615e+00, -1.49347925e+00, -9.64482844e-01,
  2.50153750e-01, -7.82087982e-01,  4.84110743e-01,
  2.68077135e-01,  1.40526056e-01, -1.23819184e+00,
 -3.47121149e-01, -7.79963315e-01,  4.00314808e-01,
 -4.81554726e-03,  1.30609512e+00, -1.71794772e-01,
 -3.95496774e+00,  2.32870150e+00,  3.45204701e-03,
 -4.25210781e-02,  1.14891934e+00, -4.26650465e-01,
  1.71035409e+00,  2.40541652e-01, -9.67137441e-02],
[-2.28169516e-01, -5.86061358e-01,  5.82860827e-01,
  1.41892731e-01,  5.37715137e-01,  4.77755606e-01,
 -1.18800688e+00,  1.30178499e+00, -3.02138835e-01,
 -2.44464445e+00, -2.14321747e-01, -2.24126186e-02,
 -6.47299409e-01, -4.11805451e-01,  4.11235169e-02,
  7.78244615e-01, -1.05806828e+00, -1.17954361e+00,
 -1.07538962e+00, -2.21818233e+00, -1.89120084e-01,
 -3.13134968e-01, -5.00645936e-01, -1.15030265e+00,
  6.36794090e-01,  8.19177330e-01,  8.05327058e-01,
  1.03423081e-01,  1.12951279e+00,  1.81257315e-02],
[-7.96185315e-01, -1.04567873e+00, -5.13373017e-01,
 -1.86142182e+00, -7.75779009e-01, -4.84131426e-01,
```

```
 -1.52014029e+00, -6.70105875e-01, -5.13140798e-01,
  1.09631217e+00, -3.00158828e-01, -6.08062744e-01,
  1.67787421e+00,  1.13150680e+00, -1.38165605e+00,
 -1.44258499e+00,  3.71838450e-01, -1.32554352e+00,
  9.04020429e-01, -1.63993835e+00, -3.07054281e-01,
 -8.38114321e-01, -4.63351980e-02,  6.43339813e-01,
  1.75945532e+00, -5.17202437e-01,  1.42636627e-01,
 -1.88168436e-01, -1.47819233e+00, -2.01526928e+00],
[-1.94290891e-01,  5.47139645e-01,  2.23354980e-01,
  1.06877673e+00,  5.78743637e-01, -2.58884758e-01,
  5.21882951e-01, -3.90851170e-01,  5.59217274e-01,
 -2.96984732e-01,  7.65249789e-01,  6.50841832e-01,
 -1.41352212e+00,  7.33723700e-01, -6.22559905e-01,
 -2.34141827e-01, -2.66722322e-01,  8.32008794e-02,
 -5.48922479e-01,  1.38990366e+00,  5.17808199e-01,
 -5.32640457e-01,  9.75996032e-02, -1.46569952e-01,
 -2.34594181e-01, -2.72881061e-01,  2.56298900e+00,
 -5.74550740e-02, -2.07463980e-01, -5.84076226e-01],
[-3.27699780e-01, -1.22060418e+00, -5.98929763e-01,
  5.36981523e-01, -1.39852417e+00,  9.35750678e-02,
 -3.10953307e+00, -1.34474647e+00,  5.92645526e-01,
  1.32024300e+00,  7.77152538e-01,  7.64985561e-01,
 -1.06752396e+00,  4.27711636e-01, -1.77270818e+00,
  1.53064275e+00,  1.40658140e+00, -1.12523496e+00,
 -1.45883572e+00, -1.25666964e+00,  7.04548597e-01,
 -1.96030033e+00, -3.11715817e+00,  3.13442260e-01,
 -3.85325015e-01, -1.06247830e+00, -2.05621719e-01,
 -3.37846613e+00,  4.90842134e-01,  1.70675188e-01],
[ 1.79861546e-01, -1.33600652e-01, -3.89672041e-01,
 -9.52542126e-01, -5.32156527e-02,  8.85752589e-03,
  6.95943534e-01, -9.86197472e-01,  9.15462077e-01,
  2.74414748e-01,  9.19051290e-01,  2.02467218e-01,
  1.26773402e-01, -1.29361168e-01, -3.84305298e-01,
  1.10087764e+00, -1.23705554e+00, -2.27398023e-01,
  5.86468160e-01, -6.39871418e-01,  2.55843550e-01,
  2.79940575e-01, -9.89333093e-02, -4.35305625e-01,
  3.51698190e-01, -3.22723389e-02, -2.09526062e-01,
  4.94070053e-01,  3.50631028e-01,  3.92606258e-01],
[ 2.47538710e+00, -2.29342729e-02, -2.78279841e-01,
  7.72828639e-01,  3.02084386e-01,  2.03692943e-01,
  2.38515353e+00, -9.07393098e-01, -1.84887826e+00,
  4.32323277e-01, -3.56564015e-01, -8.04459095e-01,
  6.22309923e-01, -1.26872802e+00,  2.26779953e-01,
  1.46652186e+00,  3.37635614e-02, -6.39442801e-01,
 -1.02566850e+00, -1.93759218e-01,  1.04638886e+00,
  7.11570203e-01, -1.00412154e+00,  1.25150710e-01,
 -1.18716192e+00,  1.32828426e+00, -2.51575172e-01,
  1.51566732e+00, -6.86308384e-01,  1.80933344e+00],
```

```
        [-7.79888220e-03,  4.20138508e-01, -2.02018574e-01,
          5.79533160e-01,  1.30674273e-01, -4.11021104e-03,
         -7.82849729e-01,  2.32158497e-01,  1.00247592e-01,
         -4.46199536e-01,  9.67945337e-01,  2.81087123e-02,
         -8.35324407e-01,  1.71313867e-01, -3.84957671e-01,
         -2.00092793e-01,  1.18132317e+00,  4.06033248e-01,
         -4.87721026e-01, -4.01466072e-01,  6.07978702e-02,
         -6.16312802e-01,  6.88161492e-01, -1.82104602e-01,
         -3.25706601e-01, -6.71487581e-03,  2.64730960e-01,
          4.11791652e-02,  6.25247061e-02,  2.37255678e-01],
        [-3.02535564e-01, -1.61347404e-01,  1.45913631e-01,
         -1.03410490e-01,  2.52265424e-01, -9.87298340e-02,
         -1.16733298e-01, -5.16295373e-01,  5.81672013e-01,
          5.94814658e-01,  6.24363363e-01,  2.20336542e-01,
         -2.16783360e-02, -1.44628823e-01, -1.14541459e+00,
         -2.50128984e-01,  2.62841612e-01, -3.66278946e-01,
         -2.23522514e-01, -3.80136728e-01, -1.58126980e-01,
         -5.83113730e-01, -2.16806626e+00, -1.19249374e-01,
          3.66219312e-01, -1.09952509e+00, -7.18302801e-02,
          7.78727591e-01, -1.31889824e-02, -5.57884648e-02],
        [ 1.48119733e-01, -6.04460649e-02, -1.86674386e-01,
         -4.05126780e-01,  1.23902154e+00, -2.21607909e-02,
          9.53316808e-01, -1.08492911e+00,  4.64152157e-01,
          7.39857852e-01,  1.48508966e+00,  9.28609967e-01,
         -2.85127640e-01, -4.77671027e-01, -4.05866176e-01,
         -3.93038958e-01, -1.34121883e+00,  6.31062686e-01,
         -2.56638885e-01, -2.26563573e+00, -3.27360511e-01,
          9.14779961e-01,  3.18116117e+00,  9.62874070e-02,
          7.46944547e-01,  5.44816732e-01,  6.84997082e-01,
          4.36594605e-01,  6.59842253e-01, -1.46086782e-01]], dtype=float32),
 array([ 0.08603439, -0.3097831 , -2.7533543 , -5.5542197 ,  0.51342434,
        -0.5735795 , -2.046294  , -1.319533  ,  2.9560566 ,  5.107563  ,
        -6.2915263 ,  1.4929398 ,  1.4985514 ,  4.025086  ,  1.7312226 ,
        -3.9450548 ,  4.553527  ,  3.7217822 ,  4.193882  , -4.70492   ,
         1.221635  ,  0.7254733 ,  0.5856059 ,  3.5372574 ,  3.7358053 ,
        -4.16473   ,  2.1288512 , -0.9938926 ,  3.2687972 , -2.4124732 ],
       dtype=float32),
 array([[-4.59673405e-01, -6.54037535e-01, -2.29213193e-01,
          7.83853352e-01,  1.66486311e+00, -1.51030064e+00,
          5.28936507e-03, -1.65034461e+00,  2.31212258e+00,
          8.39948595e-01, -6.66751564e-02, -2.48069167e-01,
         -4.67541590e-02, -1.88946283e+00, -1.28358960e+00,
          7.17815310e-02, -2.41940975e+00, -6.81465641e-02,
          1.07808232e+00, -7.76750669e-02, -8.75963196e-02,
          8.85499191e+00,  3.03462148e-02,  2.54217852e-02,
         -2.36630421e-02, -2.43488073e+00, -3.99612808e+00,
         -4.62234497e-01, -5.05486317e-02,  2.19241872e-01],
        [-1.25285435e+00, -1.12572432e+00, -9.75456476e-01,
```

```
  4.33948874e-01,  6.38818443e-01,  1.36393178e+00,
 -6.04893491e-02, -2.76325464e+00,  4.32748288e-01,
 -3.39337230e-01,  6.43776804e-02, -9.40576613e-01,
 -1.23236917e-01,  1.65593162e-01,  1.07888925e+00,
 -1.22977905e-02, -2.91216254e-01, -7.45915920e-02,
  4.97377068e-01, -2.55797505e-02, -8.94421265e-02,
 -1.12464559e+00,  2.22994778e-02, -7.02021196e-02,
  1.23600513e-01, -3.62612152e+00,  6.34078979e-01,
 -4.90670949e-01, -6.68483451e-02, -7.44461596e-01],
[ 3.23142074e-02,  8.20458889e-01,  1.83993924e+00,
  9.47062671e-01,  6.27624512e-01, -7.28026676e+00,
 -3.39780003e-02, -1.81217885e+00,  4.93368387e-01,
  4.17777598e-01,  1.97592638e-02,  4.64641064e-01,
 -5.65606318e-02,  1.40580550e-01,  2.74157941e-01,
  7.88301453e-02, -1.48610568e+00,  4.52459268e-02,
  2.86546081e-01, -4.90072854e-02, -3.22011076e-02,
 -3.04909658e+00,  4.02287394e-02, -7.21854717e-03,
 -9.31219663e-03,  1.34255934e+00, -1.09049106e+00,
 -1.60193563e+00,  1.12932129e-02,  6.56200826e-01],
[-2.74417073e-01,  1.20430791e+00, -1.97327566e+00,
  1.10758841e+00,  2.05924034e+00, -2.00917912e+00,
  1.12066716e-01,  1.30288351e+00,  4.56042439e-01,
  1.32389534e+00, -8.35717320e-02,  2.60036540e+00,
  3.59269716e-02, -6.68765008e-01, -2.69458818e+00,
 -2.69154646e-02,  1.80957496e-01, -9.13133547e-02,
  5.01264989e-01, -3.66435689e-03, -3.44935246e-02,
 -1.21295857e+00, -9.97429714e-02,  9.65489075e-02,
 -1.83167551e-02,  8.20981622e-01, -1.51372743e+00,
 -1.39583468e+00, -5.53198392e-03,  3.13509583e-01],
[-2.14987561e-01, -9.28157419e-02, -2.82754451e-02,
  2.88796973e+00, -3.05107348e-02,  1.61277074e-02,
 -4.51064855e-02, -6.06227573e-03,  6.49161279e-01,
 -6.35150194e-01, -3.77359204e-02,  1.16989148e+00,
  6.28505973e-03, -5.06632030e-01, -2.16129258e-01,
 -1.34366974e-02,  5.42497635e-01, -1.78687349e-02,
  1.71342105e-01,  2.78552994e-02, -5.73823377e-02,
 -3.37764502e-01, -9.51581635e-03, -4.01517041e-02,
 -8.13901145e-03, -5.58167212e-02,  1.01219796e-01,
 -7.72050619e-01,  6.55780360e-02, -1.27598608e+00],
[ 2.50521421e-01, -8.67724299e-01, -8.53468955e-01,
  6.10678077e-01,  3.85542780e-01,  1.09417073e-01,
 -6.83968142e-02,  6.44006193e-01,  5.70700645e-01,
  9.89671171e-01, -3.02203503e-02, -1.43093026e+00,
  2.50673108e-02, -6.69354081e-01, -1.47297636e-01,
 -1.07277296e-01,  3.46821547e+00,  1.07603490e-01,
  8.29106927e-01,  1.34063199e-01,  8.63130111e-03,
 -2.79103112e+00,  3.16866674e-02,  5.53745590e-02,
  4.24981304e-02,  3.92297328e-01,  1.80722821e+00,
```

```
 -6.94261312e-01, -3.01367287e-02, -5.33842504e-01],
[ 1.40566036e-01,  2.36746645e+00,  4.27397013e-01,
 -7.60884210e-02,  6.30335152e-01, -1.76663518e-01,
 -1.44613683e-02, -1.10651875e+00,  6.78432584e-01,
  2.49900907e-01,  7.02060992e-03,  5.92293501e-01,
  2.05632555e-03,  8.88014957e-02, -5.46075523e-01,
  1.93899255e-02, -6.39753103e-01,  1.85748674e-02,
  4.57974114e-02,  3.44654582e-02,  4.03563929e-04,
  8.24237525e-01, -3.00256852e-02,  8.22237134e-02,
  1.69851817e-02, -3.20611000e-01, -1.65109050e+00,
 -1.68003570e-02,  1.44935977e-02,  5.62664568e-01],
[ 8.21805969e-02,  5.51132917e-01, -6.24253392e-01,
 -1.60447881e-01, -5.13202906e-01, -1.10493608e-01,
  5.27389348e-02,  1.74418703e-01, -7.06856728e-01,
  1.18909642e-01, -2.02896483e-02, -1.41827738e+00,
  5.70003167e-02,  4.00282413e-01,  1.70145965e+00,
  1.32183731e-02, -2.63624728e-01, -6.13024086e-02,
  1.14378616e-01,  1.38742030e-02,  3.95545699e-02,
  1.07449150e+00,  1.57156177e-02, -4.33919504e-02,
 -5.06527238e-02,  2.11141363e-01,  1.45536900e-01,
 -1.02718127e+00,  1.19166998e-02,  2.13730484e-01],
[ 3.48001063e-01, -4.76047099e-01, -3.51602495e-01,
 -6.60171032e-01, -7.78117836e-01,  2.31217194e+00,
  3.20723802e-02,  9.52519029e-02, -1.48318994e+00,
 -1.61375970e-01, -4.78194281e-02,  6.36454344e-01,
 -4.12561260e-02, -8.49414110e-01, -3.34640622e-01,
  6.06650710e-02,  8.06942701e-01,  2.37528060e-04,
 -9.58501995e-02, -5.14872335e-02, -7.42011517e-02,
  1.74755728e+00, -3.00118458e-02,  2.04080511e-02,
  4.60211933e-02,  2.39731288e+00, -1.30286145e+00,
  1.51180756e+00, -2.38490384e-02,  1.38465151e-01],
[-1.84596330e-02,  4.39698994e-01,  4.93639082e-01,
  9.35006917e-01,  1.20573974e+00, -3.54527354e+00,
  6.21000379e-02, -3.13999891e+00,  1.67677367e+00,
 -9.14901316e-01,  2.79635470e-03,  2.20394421e+00,
  8.55490752e-03,  5.46698213e-01,  2.04429030e+00,
  4.52331752e-02,  3.28443122e+00, -2.75347643e-02,
  5.02370857e-02, -2.73011271e-02, -9.06747580e-03,
 -1.07226062e+00, -1.33653199e-02,  4.14420990e-03,
 -3.82005125e-02, -1.12327600e+00,  1.46538031e+00,
  6.86974764e-01, -1.92054864e-02, -7.78099746e-02],
[-6.66321695e-01,  5.76930046e-01,  3.19954187e-01,
  3.62568870e-02,  2.14860424e-01,  7.18049884e-01,
 -1.00815585e-02, -1.55247048e-01,  3.41020912e-01,
  6.34874821e-01, -1.87569316e-02,  3.58190447e-01,
 -3.97522822e-02, -6.85382783e-01, -2.95467114e+00,
 -2.20909379e-02, -2.20376238e-01,  6.92602247e-03,
  3.15780193e-01,  7.04548089e-03,  8.55563954e-03,
```

```
 -1.84376538e+00,  3.18118860e-03,  5.33651970e-02,
 -1.27527742e-02,  1.84018075e-01, -1.48961139e+00,
 -1.42126465e+00, -2.24684202e-03, -3.81311178e-01],
[ 3.75280917e-01,  4.03310776e-01, -4.95658815e-01,
 -3.00245464e-01, -6.36031389e-01,  1.44528091e+00,
  3.25018354e-02, -5.61079144e-01,  1.50202596e+00,
 -6.71877861e-01,  1.58922654e-02, -1.37509477e+00,
 -1.83230899e-02, -1.54924244e-01,  2.66642094e+00,
  3.14145163e-02,  8.21621180e-01, -4.57602069e-02,
 -4.00610780e-03, -2.62907352e-02,  3.03727891e-02,
  4.04023498e-01,  2.68674623e-02,  1.62764657e-02,
 -7.83260763e-02,  8.92726481e-01, -1.33494520e+00,
  1.10254943e+00,  5.02049690e-03,  4.08067435e-01],
[ 4.45128620e-01, -7.97597587e-01,  1.46990970e-01,
 -6.79995179e-01, -8.00864637e-01, -2.97396016e+00,
  7.55124586e-03,  2.23712265e-01, -1.06625283e+00,
 -9.02717039e-02,  1.70490686e-02, -8.66852224e-01,
 -3.25962156e-02,  3.03578675e-02, -9.01367217e-02,
  6.15010858e-02,  1.21032643e+00, -3.08804289e-02,
 -6.91427663e-02, -3.17846537e-02, -2.43249536e-02,
  2.15507364e+00, -9.31077171e-03,  2.94888765e-03,
 -3.95958452e-03, -2.38235164e+00, -3.38670939e-01,
  7.12548673e-01,  1.56136649e-02,  9.56590399e-02],
[ 4.80259627e-01, -1.04683205e-01,  3.16769809e-01,
 -8.51143524e-03, -5.84970832e-01, -9.54566419e-01,
 -4.13460992e-02,  1.45252728e+00, -1.05747628e+00,
  1.92932397e-01,  4.90570739e-02,  4.50485080e-01,
  2.06989404e-02,  1.18492651e+00,  7.79610217e-01,
 -6.05067909e-02, -3.37241888e-01,  3.42061780e-02,
 -3.00901622e-01,  9.52753797e-02,  4.25716378e-02,
  2.13583305e-01,  1.70147289e-02, -3.78271900e-02,
 -8.29808600e-03,  2.58236736e-01,  9.28548396e-01,
  4.68022525e-01,  4.43711467e-02,  3.45550925e-01],
[ 9.07137394e-01, -4.74406570e-01,  2.20176786e-01,
  6.15145639e-03,  1.83478773e-01, -2.31356764e+00,
 -7.96051249e-02, -5.06995618e-01, -1.18195152e+00,
 -5.82005084e-01,  6.21035555e-03,  1.05579996e+00,
 -1.58665664e-02,  9.86638367e-01,  3.87648761e-01,
 -3.99678433e-03,  5.87666094e-01,  1.29010439e-01,
 -3.69599871e-02,  5.74356131e-02, -3.35991457e-02,
 -1.03540707e+00,  9.56464335e-02, -9.11264718e-02,
  5.87170199e-02, -2.74113685e-01,  1.17248213e+00,
 -1.14658904e+00, -6.37442842e-02,  4.28282768e-01],
[-1.60590708e-01,  1.23504066e+00, -1.37706816e+00,
 -7.14426935e-01, -7.70941556e-01, -2.51486874e+00,
  1.32673606e-02, -8.46150041e-01, -6.46191776e-01,
  3.88464302e-01, -1.65337361e-02, -1.26572025e+00,
 -2.40613446e-02,  1.59728277e+00,  6.79417014e-01,
```

```
  3.14833187e-02, -6.13595128e-01,  1.04526551e-02,
  5.00962019e-01, -4.66561690e-02, -8.81525129e-02,
 -1.42226112e+00, -3.42466868e-02,  2.20459513e-02,
  6.47907332e-02,  6.52887046e-01, -8.38347256e-01,
 -1.61939704e+00, -1.02273980e-03, -1.73221183e+00],
[ 2.43798167e-01,  4.61631596e-01,  1.84785545e-01,
 -2.25744277e-01, -7.02175558e-01,  3.49180794e+00,
 -9.26340446e-02, -7.21047580e-01, -2.55707741e-01,
 -4.08423841e-01,  4.25307229e-02, -1.12484503e+00,
 -1.11218467e-01,  5.34531951e-01, -1.95180420e-02,
  2.85875779e-02,  1.10296655e+00,  2.86334120e-02,
  2.73416817e-01,  1.65628847e-02,  2.44537555e-02,
 -5.70235312e-01,  1.19750544e-01, -5.56826266e-03,
 -1.36122346e-01, -1.94863963e+00, -1.75808990e+00,
 -2.21085146e-01,  4.72070761e-02, -1.82778850e-01],
[ 6.18870437e-01, -4.46494937e-01,  9.46239650e-01,
 -1.18953049e+00,  9.18901384e-01, -2.02927518e+00,
 -3.39708664e-02,  9.33683634e-01, -4.62186903e-01,
 -1.08073318e+00,  1.36203095e-01, -4.34051692e-01,
  3.38841900e-02,  2.22826540e-01,  3.66048312e+00,
  8.52562860e-02,  9.52807069e-01, -4.79386933e-03,
 -4.92944598e-01, -2.60983352e-02,  8.63587186e-02,
  2.79038477e+00, -4.84783296e-03, -3.52163911e-02,
  6.41170070e-02, -1.99482119e+00,  1.81940687e+00,
  1.92503250e+00,  1.01362672e-02, -4.76260036e-01],
[ 6.53931499e-02, -6.49955571e-01,  1.33681560e+00,
  4.96534020e-01,  4.90655541e-01, -2.30977535e+00,
  1.22243436e-02,  2.30268955e+00,  9.90423411e-02,
 -4.01982933e-01, -1.78914014e-02,  6.81520477e-02,
 -4.54483703e-02, -1.26106811e+00,  6.87540650e-01,
  1.99001804e-02, -1.36513758e+00, -1.01157976e-02,
 -2.67670691e-01, -7.63736740e-02, -4.04898860e-02,
  9.48190808e-01, -2.46875696e-02,  5.74347638e-02,
  1.52240722e-02,  3.61768126e+00,  1.32714078e-01,
  1.05588055e+00,  7.49120209e-03, -1.08301926e+00],
[-2.25216761e-01,  6.29032969e-01,  9.13138594e-03,
  2.28741780e-01, -1.28787821e-02,  3.63361418e-01,
 -1.15756080e-01, -3.09692502e+00,  2.72977042e+00,
  9.35225964e-01,  6.49616569e-02,  2.87169248e-01,
 -7.54489675e-02, -2.54159540e-01, -1.04769528e+00,
  1.16711840e-01, -1.18651438e+00, -2.22298456e-03,
  2.15674445e-01,  1.65729527e-03, -3.15590650e-02,
 -2.16833425e+00,  1.35611370e-01, -8.13765004e-02,
  9.18022078e-03,  7.67188132e-01,  1.91119790e-01,
 -1.27206314e+00, -6.67762831e-02,  6.40604913e-01],
[ 6.16978467e-01,  1.72225046e+00, -8.85201618e-02,
 -1.96832865e-01, -2.02234000e-01, -6.27399147e-01,
  3.70378755e-02, -7.98539594e-02, -1.02798748e+00,
```

```
 -6.78186417e-01, -1.88273918e-02, -3.17474574e-01,
 -1.09705729e-02,  1.11698970e-01,  9.86799538e-01,
  1.59738325e-02,  1.72473714e-01, -4.00225259e-02,
 -1.80547923e-01, -7.73987323e-02,  5.25049225e-04,
 -1.69234008e-01,  2.77892984e-02, -6.48174584e-02,
 -2.60066334e-03, -6.24081314e-01, -7.37443194e-02,
  7.97151387e-01, -6.40420318e-02,  3.79423380e-01],
[-3.24391961e-01, -6.47847056e-01, -1.52175263e-01,
  2.84788489e-01,  7.72810355e-02,  1.34422386e+00,
 -1.27780410e-02, -4.27443713e-01, -2.73770422e-01,
  5.12028456e-01,  1.62014514e-02, -1.14826458e-02,
 -4.80439663e-02,  8.32796752e-01, -7.48256445e-01,
  5.20904027e-02,  1.55292854e-01,  1.45340711e-03,
  1.00503497e-01, -5.51163778e-02, -3.64956893e-02,
 -2.10766613e-01,  4.80566500e-03, -6.06307853e-03,
  5.19074313e-03,  6.89225674e-01,  1.74997652e+00,
 -7.62445688e-01,  1.37773817e-02, -1.56199574e-01],
[-1.57803088e-01, -7.56211430e-02,  2.76943613e-02,
  3.39336962e-01,  8.24765921e-01,  1.45415020e+00,
  1.25200823e-01, -1.99678791e+00, -1.68380868e+00,
  4.64872532e-02, -1.18586317e-01, -4.01447207e-01,
  2.13889480e-02, -1.72917128e-01,  3.09751302e-01,
 -9.20330733e-03, -3.82353455e-01, -7.10345991e-03,
 -3.17878239e-02, -2.93467212e-02, -7.50469789e-02,
 -9.92542028e-01, -2.22706720e-02, -4.50776517e-02,
  5.30875549e-02,  8.15628886e-01, -1.32370675e+00,
 -4.09513235e-01, -1.16940998e-01,  3.92551124e-01],
[ 2.54403114e-01, -4.36814994e-01, -8.22807848e-01,
  5.21905087e-02,  1.98714569e-01, -1.43597692e-01,
 -6.69597462e-02,  8.08064461e-01, -1.05224812e+00,
 -7.79520988e-01, -9.00441036e-03,  2.91694075e-01,
  2.84441840e-03,  5.84365547e-01, -2.49758184e-01,
 -4.01679389e-02, -5.06159484e-01,  4.82440107e-02,
 -2.64768004e-01,  1.07188709e-02, -5.73772984e-03,
  1.63753021e+00, -3.29522975e-02,  5.13964929e-02,
  8.36980529e-03,  4.01509106e-01,  1.97845411e+00,
 -4.24319059e-01,  7.41496533e-02,  2.64208734e-01],
[ 3.49920481e-01, -2.51371324e-01,  1.86756849e-01,
 -4.35704321e-01, -8.51697803e-01,  2.14221430e+00,
 -3.09704673e-02, -1.90922725e+00, -2.43407235e-01,
 -1.14775598e+00,  1.29647534e-02,  1.26978114e-01,
  5.93676278e-03,  1.78671610e-02,  2.69713569e+00,
  4.47295345e-02,  1.02945197e+00, -4.32216413e-02,
 -1.12859309e-01,  4.40916494e-02,  2.26235110e-02,
  1.32880163e+00,  2.71884222e-02, -1.46807674e-02,
 -4.23307270e-02, -1.07328188e+00, -1.52603477e-01,
  1.04838872e+00,  6.17234327e-04,  2.11343020e-01],
[-3.52853149e-01,  1.38718522e+00, -3.49788189e-01,
```

```
  1.13229297e-01,  6.92173466e-02, -2.36705089e+00,
  1.71964208e-03, -3.21136534e-01, -1.49155021e+00,
  5.67800820e-01, -9.19310749e-03,  4.87280726e-01,
  4.39777710e-02, -4.28988859e-02, -1.50416601e+00,
  2.91129425e-02, -1.40719160e-01, -1.02855973e-02,
  2.71053880e-01, -1.56487245e-02,  1.17879948e-02,
 -2.75240731e+00,  4.02605720e-02, -3.03773116e-02,
 -1.64339039e-02, -7.94540048e-01, -5.91772258e-01,
 -1.46924472e+00, -4.63805050e-02, -7.18538105e-01],
[ 3.08034092e-01, -7.37401322e-02, -3.74462247e-01,
 -1.98905662e-01,  7.70581126e-01, -3.88558805e-01,
 -3.70815881e-02, -1.43595532e-01, -1.28276587e+00,
  8.01239684e-02,  2.27188095e-02, -6.81865036e-01,
 -6.57725567e-03,  7.14557409e-01,  1.02130103e+00,
  2.37337570e-03,  4.69690144e-01, -7.82996044e-03,
  1.10302828e-01, -1.87638495e-02,  4.54581203e-03,
  3.90936911e-01,  5.58428727e-02,  4.64445306e-03,
 -4.38056625e-02, -4.43072766e-01,  2.07188869e+00,
  1.28666925e+00,  4.46881391e-02,  1.13478637e+00],
[-3.38254124e-02,  1.84237158e+00, -6.96639001e-01,
  3.42306286e-01, -8.35664868e-02,  4.23246574e+00,
  5.95746748e-02,  1.22058237e+00, -1.61769938e+00,
  1.12840223e+00, -1.22274198e-02, -6.83897510e-02,
  6.60105720e-02,  1.08710080e-01,  5.85694849e-01,
 -2.67298315e-02,  1.52834833e-01, -1.69515777e-02,
  4.34023321e-01, -3.21768932e-02,  4.34057266e-02,
  6.73382103e-01,  4.28010616e-03, -2.01083273e-02,
 -1.71470027e-02,  2.89101839e-01,  2.45886624e-01,
 -1.98310185e-02,  1.00298040e-03, -6.03562951e-01],
[ 7.20714271e-01, -2.09795251e-01, -8.47141743e-02,
 -2.82501578e-01,  1.23694554e-01,  1.48467433e+00,
  1.03159295e-02, -2.25504923e+00, -7.25931525e-01,
 -1.23919487e+00, -6.58563618e-03, -5.42851448e-01,
 -1.86458770e-02,  4.79575396e-01,  6.06736958e-01,
  6.06722431e-04, -1.32022774e+00,  4.51993085e-02,
 -1.94127440e-01,  3.95609550e-02, -2.03085989e-02,
  8.47139239e-01,  6.53969720e-02,  2.33218465e-02,
  2.67347936e-02, -1.63809991e+00, -2.78170514e+00,
  6.83619022e-01, -7.10887909e-02,  1.14285922e+00],
[-5.96429169e-01,  6.39205217e-01, -6.58662081e-01,
  1.17396601e-01,  2.23911703e-01, -2.23790979e+00,
  6.85726777e-02,  3.42500061e-01,  6.81941509e-01,
  2.34271741e+00,  2.45838854e-02,  2.41839468e-01,
  4.37811650e-02, -1.27655312e-01,  3.70694369e-01,
  5.06829210e-02, -2.12187052e-01, -8.53935704e-02,
  3.30974877e-01, -1.71830729e-02,  3.85299250e-02,
  6.31709769e-02,  1.86524559e-02, -2.85049137e-02,
 -8.86198133e-03, -2.07771659e-01, -2.94297755e-01,
```

```
        -1.34284186e+00, -3.86409052e-02, -2.85674661e-01]], dtype=float32),
array([ 1.8225375 , -2.4948206 ,  2.0260894 , -2.5787468 , -2.837735  ,
        1.6059943 ,  0.07396222, -1.9054754 , -1.9716415 , -3.121214  ,
       -0.02978724, -2.1414735 ,  0.06670253,  0.2886331 ,  2.3389769 ,
        0.00541568, -0.2198047 , -0.01787575, -1.0289241 ,  0.00486385,
        0.00845327,  0.91894853, -0.08716204,  0.06282846, -0.04619095,
        0.9383229 ,  0.4835131 ,  2.0221548 ,  0.04195892,  0.1033548 ],
      dtype=float32),
array([[-3.4717968e-01],
       [-2.6527724e-01],
       [ 3.6765152e-01],
       [ 2.1935374e-01],
       [ 1.8445426e-01],
       [ 6.5788761e-02],
       [-5.0895205e-03],
       [-8.3981164e-02],
       [ 9.6544378e-02],
       [ 2.4194971e-01],
       [ 1.9571478e-03],
       [-9.2007920e-02],
       [-1.5618777e-03],
       [-1.6256732e-01],
       [-2.5405091e-01],
       [-2.8014879e-03],
       [ 1.8248655e-01],
       [ 1.8508438e-03],
       [ 4.7898191e-01],
       [ 2.4726149e-04],
       [ 2.9022794e-04],
       [ 1.9850802e-01],
       [ 2.9635595e-03],
       [ 4.0588146e-03],
       [-4.7782147e-03],
       [-7.6923266e-02],
       [ 5.0785702e-02],
       [-1.5145859e-01],
       [ 4.7635529e-03],
       [-9.9846974e-02]], dtype=float32),
array([-0.26492894], dtype=float32)]
```

```
In [9]: new_model.optimizer
```

```
Out[9]: <keras.optimizers.Adam at 0x223800cbb70>
```

3 Predictions (Global)

"Global" in this sense means that I've fed model.predict() with all the possible parameters. This gives a good overview of the overall performance of the network. This can be useful to spot largescale phenomena like network overfitting but is not so great at looking at say individual data slices. This is done later.

Nota Bene: all data here is NORMALIZED at this stage.

```
In [10]: y_test_np_array = y_test.values
         print(y_test_np_array)
```

```
[-0.02230498  0.55885508 -0.65827708 ...  3.20071806  0.9469378
 -0.74114729]
```

```
In [11]: x_test_np_array = x_test.values
         print(x_test_np_array)
```

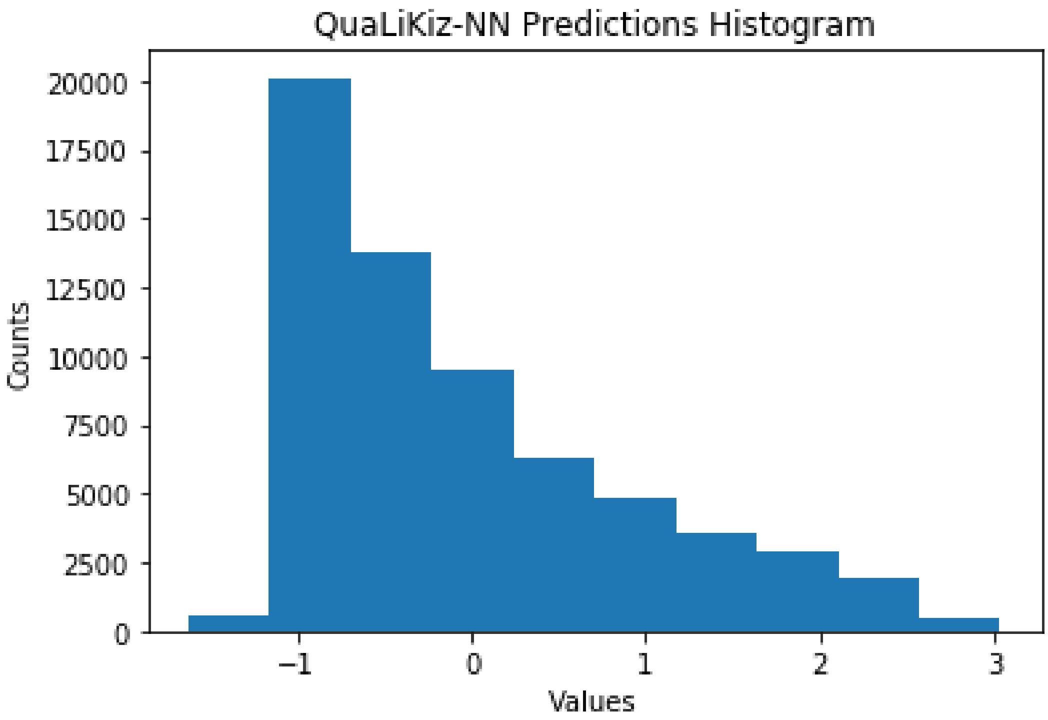
```
[[ 0.3602786  -0.41635451 -0.63490191 ...  0.20359267 -0.53767346
  -0.9260135 ]
 [-0.40414545 -0.41635451  0.25080268 ... -1.26489399  0.30047748
   0.42178003]
 [-0.91376148 -0.41635451  0.25080268 ... -1.26489399  0.30047748
   1.66589724]
 ...
 [-1.1685695   0.49254462  0.54603754 ...  0.89464524 -1.37582451
   1.66589724]
 [-0.14933743  1.81457971 -2.99678081 ... -0.05555202 -1.37582451
  -1.29628647]
 [ 0.10547058  1.81457971 -2.99678081 ... -1.26489399 -0.53767346
   0.42178003]]
```

```
In [12]: predictions_global = new_model.predict(x_test_np_array, batch_size = 10, verbose=0)
         print(type(predictions_global))

         predictions_global = predictions_global.flatten()
         print(predictions_global.shape)
         print(type(predictions_global))
```

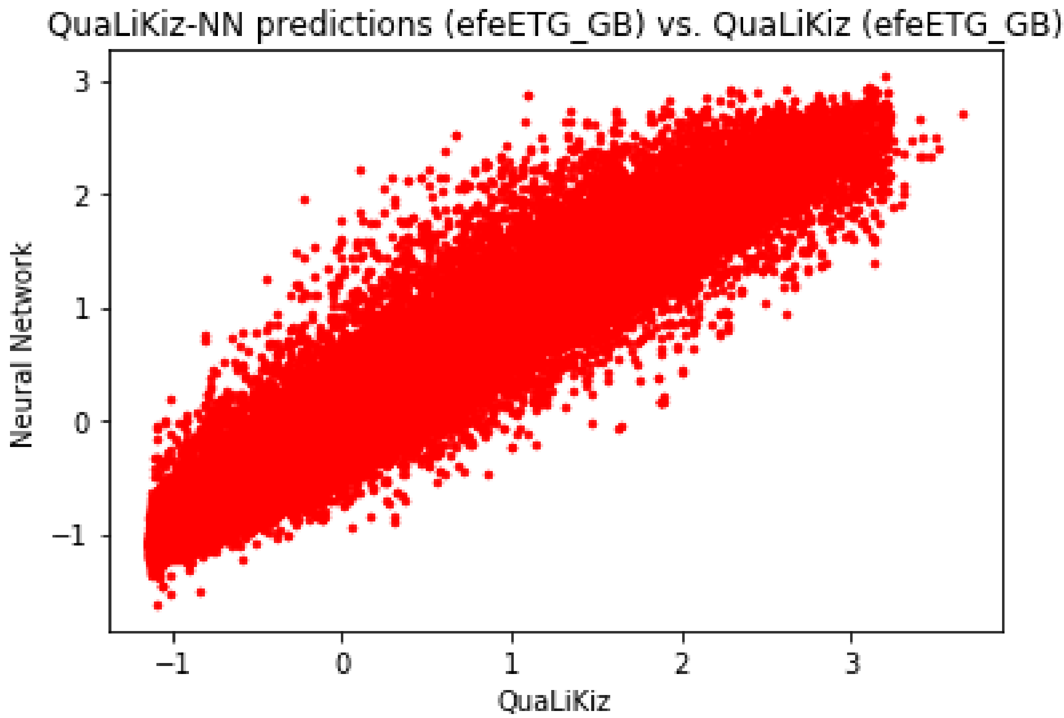
```
<class 'numpy.ndarray'>
(63888,)
<class 'numpy.ndarray'>
```

3.0.1 Predictions (Global) review - Normalized

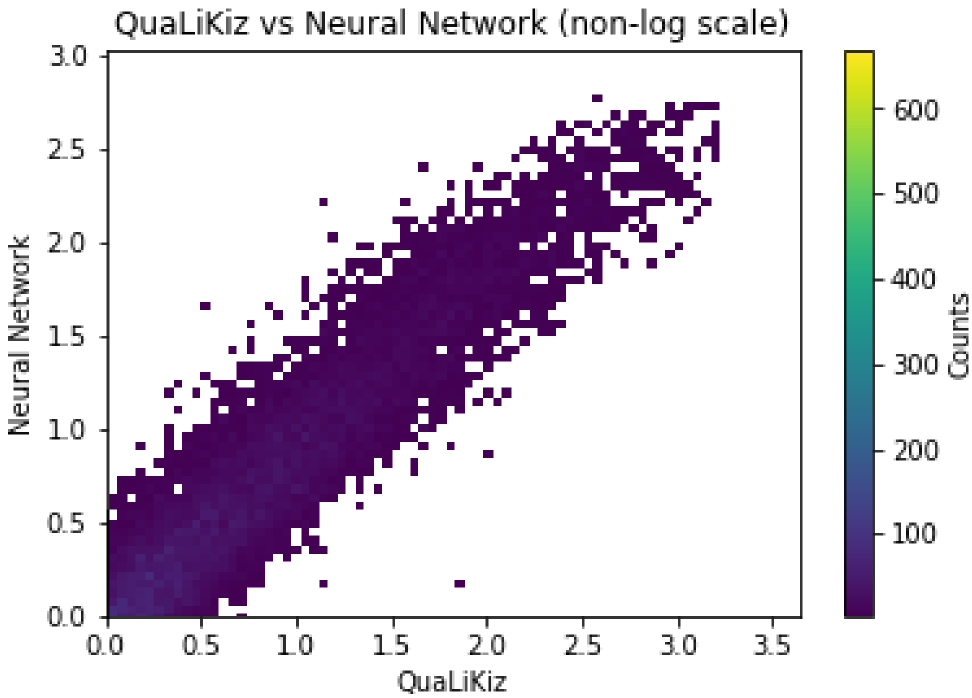
```
In [13]: plt.hist(predictions_global)
         plt.title('QuaLiKiz-NN Predictions Histogram')
         plt.xlabel('Values')
         plt.ylabel('Counts')
         # plt.savefig('./2018-07-10_Run0050b_DataSlicerPlot/NN_Predictions.png', dpi = 100)
         plt.show()
```



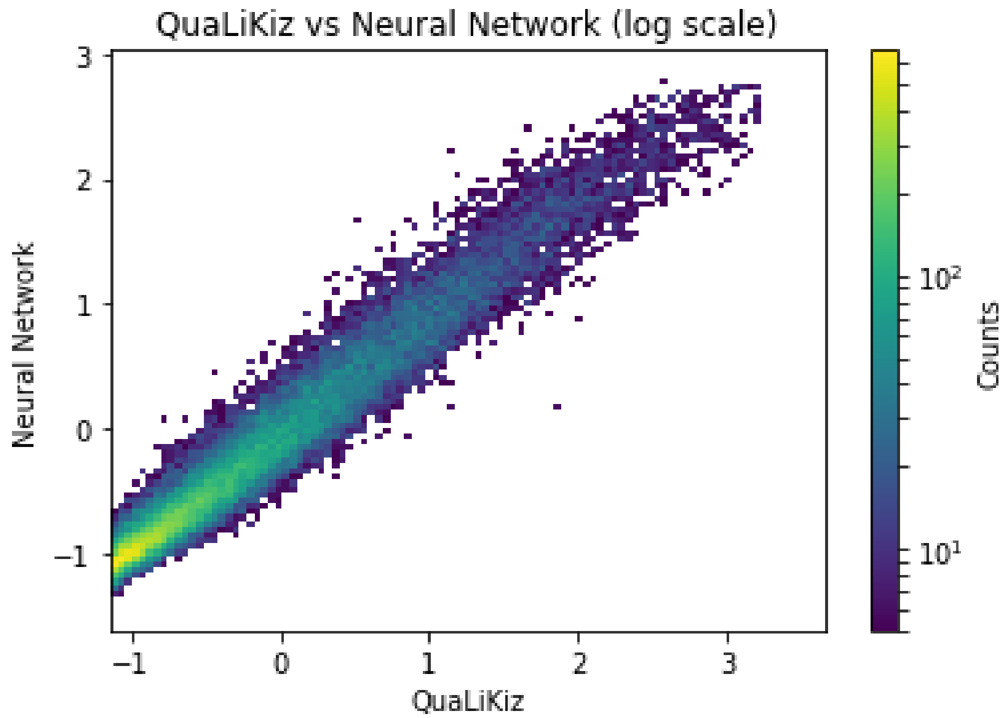

```
In [14]: plt.plot(y_test_np_array, predictions_global, 'r.', ms = 5, label = 'QuaLiKiz-NN')
         plt.title('QuaLiKiz-NN predictions (efeETG_GB) vs. QuaLiKiz (efeETG_GB)')
         plt.xlabel('QuaLiKiz')
         plt.ylabel('Neural Network')
         plt.show()
```




```
In [15]: plt.hist2d(y_test_np_array, predictions_global, bins=100, cmin=5)
         # plt.plot( [0,1],[0,1] )
         plt.title('QuaLiKiz vs Neural Network (non-log scale)')
         plt.xlabel('QuaLiKiz')
         plt.ylabel('Neural Network')
         plt.ylim(0)
         plt.xlim(0)
         cbar = plt.colorbar()
         cbar.ax.set_ylabel('Counts')
         # plt.savefig('./2018-07-10_Run0050b_DataSlicerPlot/QuaLiKiz-vs-NN_nonLogScale_bins100.
         plt.show()
```



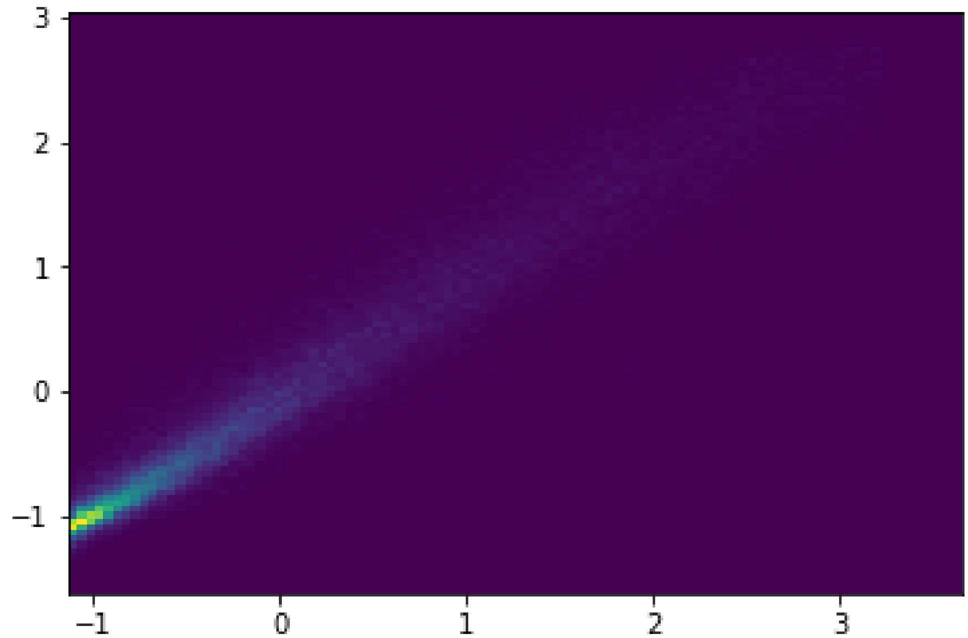

```
In [16]: plt.hist2d(y_test_np_array, predictions_global, bins=100, norm=LogNorm(), cmin=5)
         # plt.plot( [0,1],[0,1] )
         plt.title('QuaLiKiz vs Neural Network (log scale)')
         plt.xlabel('QuaLiKiz')
         plt.ylabel('Neural Network')
         # plt.ylim(0)
         # plt.xlim(0)
         cbar = plt.colorbar()
         cbar.ax.set_ylabel('Counts')
         plt.show()

C:\Users\danie\Anaconda3\lib\site-packages\matplotlib\colors.py:1031: RuntimeWarning: invalid va
  mask |= resdat <= 0
```



3.0.2 Sigmas (Normalized)

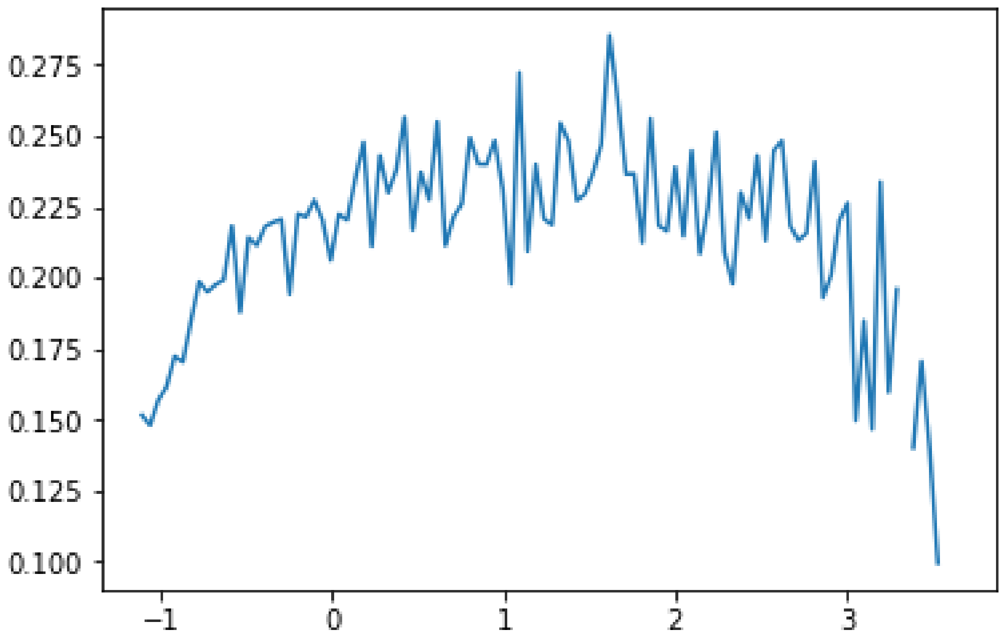
These plots show how spread out the datapoints in the earlier Neural Network vs. QuaLiKiz plots are.

```
In [17]: h = plt.hist2d(y_test_np_array, predictions_global,bins = 100)
```

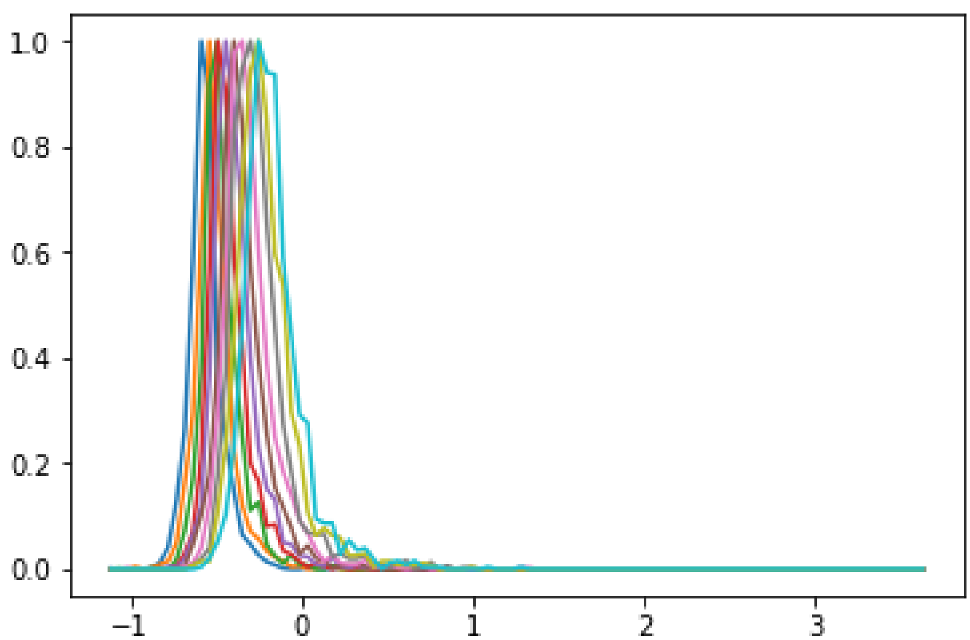



```
In [18]: sigmas = []
         Qualikiz = y_test_np_array
         for i in range(len(h[0])):
             sigmas.append(numpy.std(h[0][i] / numpy.max(h[0][i])))

C:\Users\danie\Anaconda3\lib\site-packages\ipykernel_launcher.py:4: RuntimeWarning: invalid valu
  after removing the cwd from sys.path.
```

```
In [19]: plt.plot((numpy.array(h[1][1:]) + numpy.array(h[1][:-1]))/2.,sigmas)
         plt.show()
```




```
In [20]: for i in range(10):
             plt.plot((h[1][1:] + h[1][:-1])/2., h[0][i]/numpy.max(h[0][i]) )
```

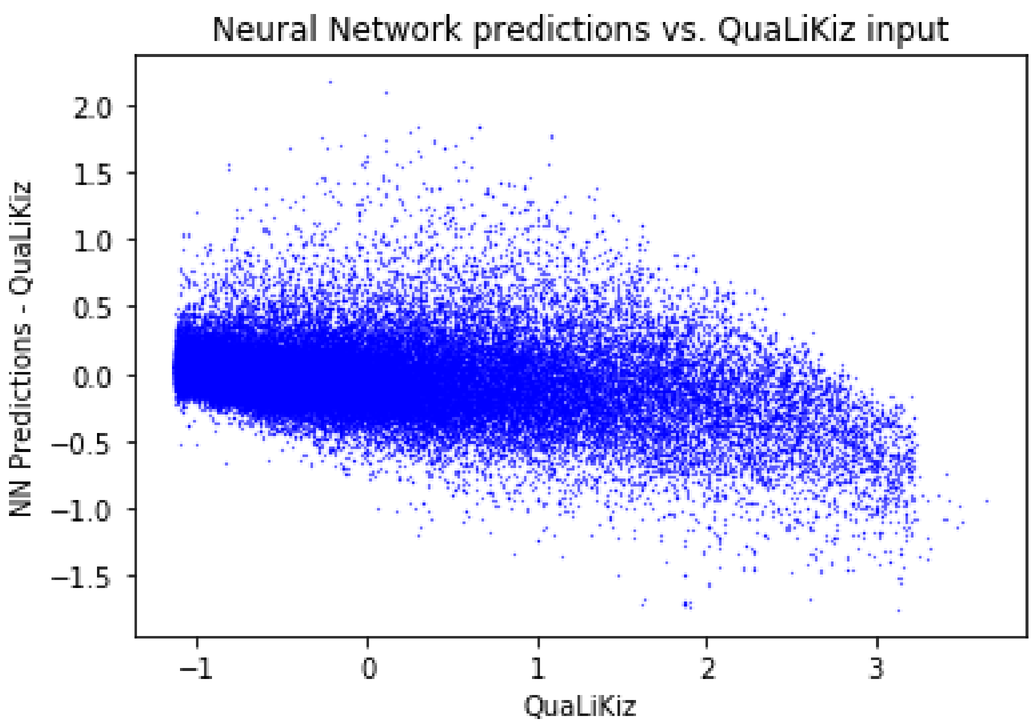


3.0.3 Exploring the spread of the predictions vs. original QuaLiKiz inputs

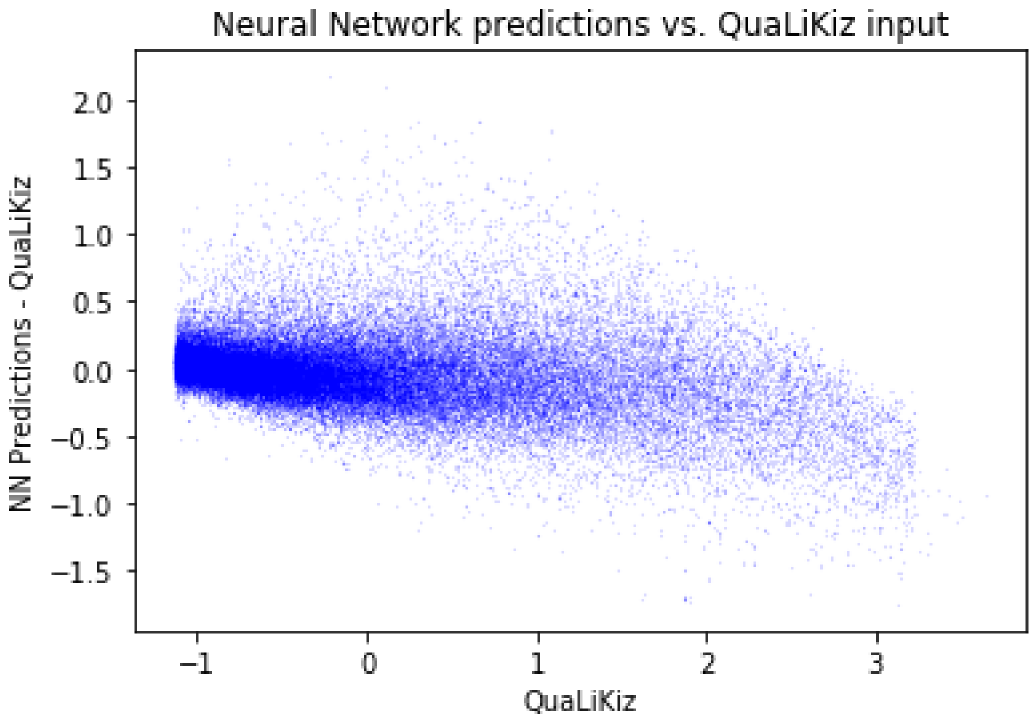
Here we explore the difference between our neural network predictions (of efeETG_GB) vs. the values of the original dataset (unstable_training_gen2_7D_nions0_flat_filter7.h5). The two graphs merely show different dot sizes.

In a perfect case we would have a single line at y = 0 signaling zero disagreement.

In [21]:
```
plt.plot(y_test_np_array, predictions_global - y_test_np_array, 'b.', ms = 0.5)
plt.title('Neural Network predictions vs. QuaLiKiz input')
plt.xlabel('QuaLiKiz')
plt.ylabel('NN Predictions - QuaLiKiz')
plt.show()
```



In [22]:
```
plt.plot(y_test_np_array, predictions_global - y_test_np_array, 'b.', ms = 0.1)
plt.title('Neural Network predictions vs. QuaLiKiz input')
plt.xlabel('QuaLiKiz')
plt.ylabel('NN Predictions - QuaLiKiz')
plt.show()
```



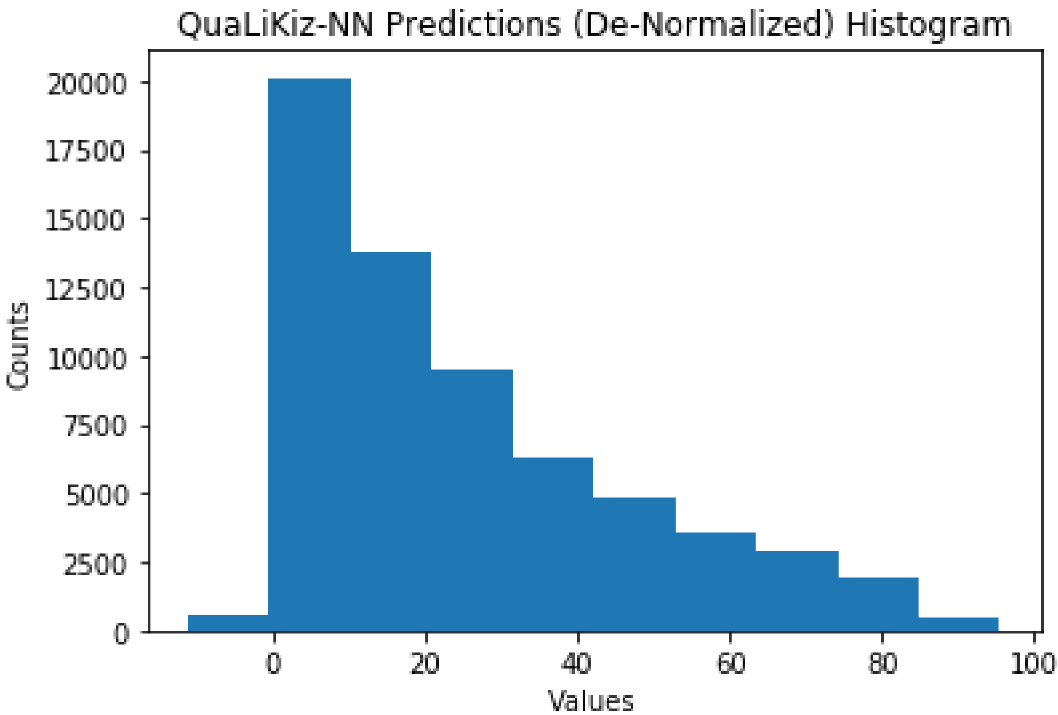
3.0.4 Predictions (Global) review - De-Normalized

```
In [23]: predictions_global_deNormalized = (predictions_global * joined_dataFrame_original['efeE
```

```
In [24]: y_test_deNormalized = (y_test * joined_dataFrame_original['efeETG_GB'].std()) + joined_
```

```
In [25]: y_test_deNormalized_np_array = y_test_deNormalized.values
```

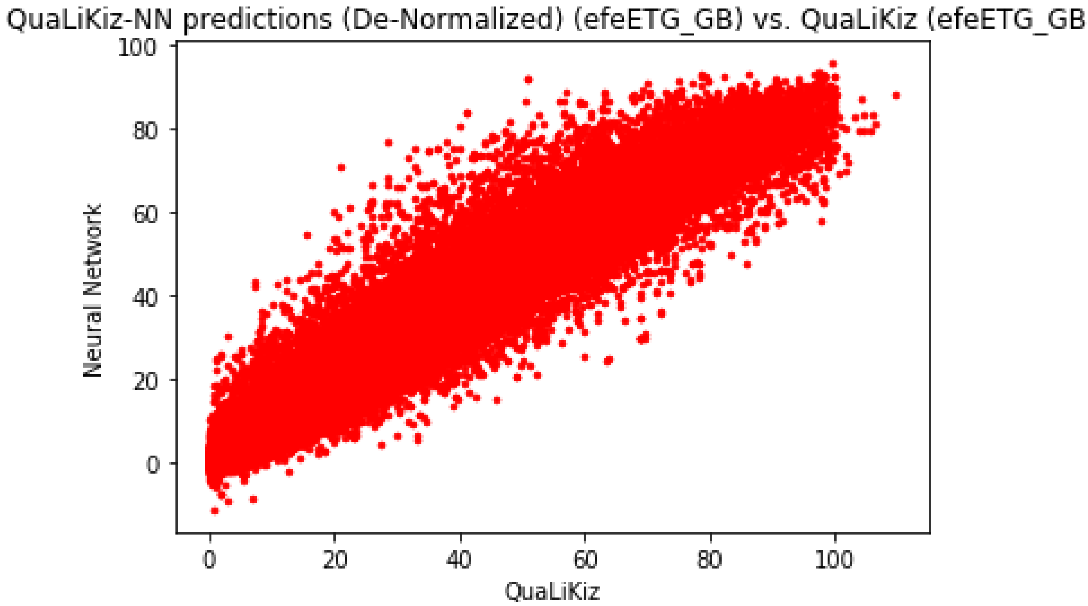
```
In [26]: plt.hist(predictions_global_deNormalized)
         plt.title('QuaLiKiz-NN Predictions (De-Normalized) Histogram')
         plt.xlabel('Values')
         plt.ylabel('Counts')
         # plt.savefig('./2018-07-10_Run0050b_DataSlicerPlot/NN_Predictions.png', dpi = 100)
         plt.show()
```



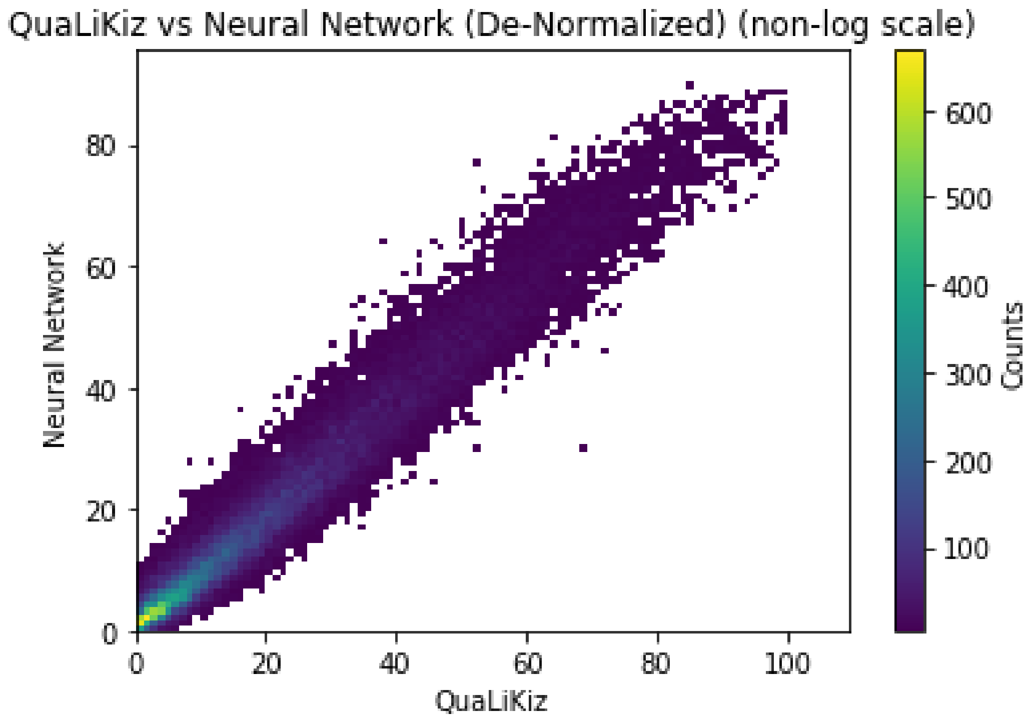

```
In [27]: print(predictions_global.shape)
```

```
(63888,)
```

```
In [28]: plt.plot(y_test_deNormalized_np_array, predictions_global_deNormalized, 'r.', ms = 5, l
         plt.title('QuaLiKiz-NN predictions (De-Normalized) (efeETG_GB) vs. QuaLiKiz (efeETG_GB)
         plt.xlabel('QuaLiKiz')
         plt.ylabel('Neural Network')
         plt.show()
```

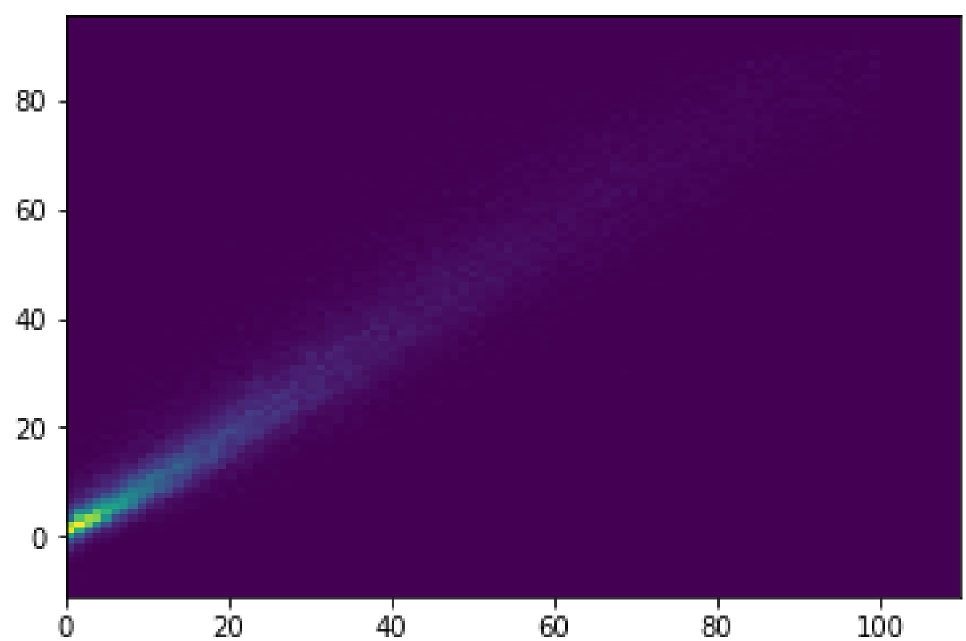



```
In [29]: plt.hist2d(y_test_deNormalized_np_array, predictions_global_deNormalized, bins=100, cmi
         # plt.plot( [0,1],[0,1] )
         plt.title('QuaLiKiz vs Neural Network (De-Normalized) (non-log scale)')
         plt.xlabel('QuaLiKiz')
         plt.ylabel('Neural Network')
         plt.ylim(0)
         plt.xlim(0)
         cbar = plt.colorbar()
         cbar.ax.set_ylabel('Counts')
         # plt.savefig('./2018-07-10_Run0050b_DataSlicerPlot/QuaLiKiz-vs-NN_nonLogScale_bins100.
         plt.show()
```



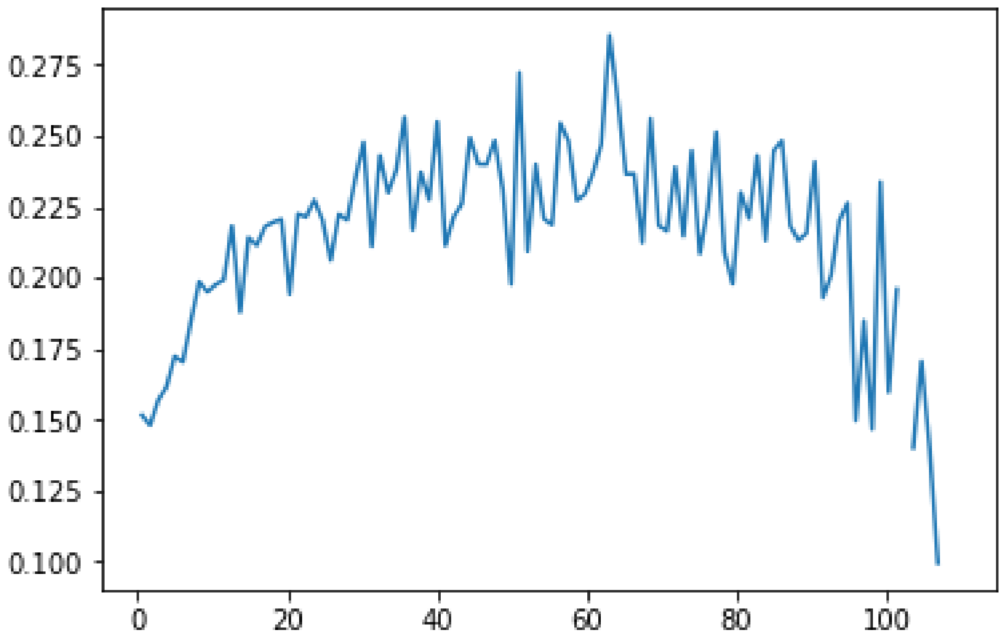
3.0.5 Sigmas (De-Normalized)

```
In [30]: h_deNormalized = plt.hist2d(y_test_deNormalized_np_array, predictions_global_deNormaliz
```

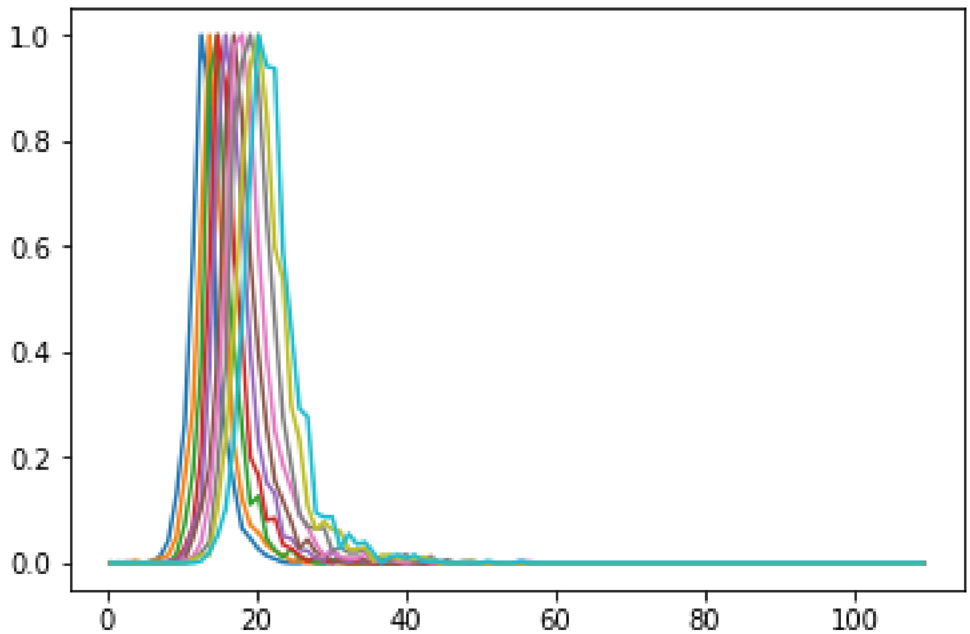


```
In [31]: sigmas_deNormalized = []
         Qualikiz_deNormalized = y_test_deNormalized_np_array
         for i in range(len(h[0])):
             sigmas_deNormalized.append(numpy.std(h_deNormalized[0][i] / numpy.max(h_deNormalize

C:\Users\danie\Anaconda3\lib\site-packages\ipykernel_launcher.py:4: RuntimeWarning: invalid valu
  after removing the cwd from sys.path.
```

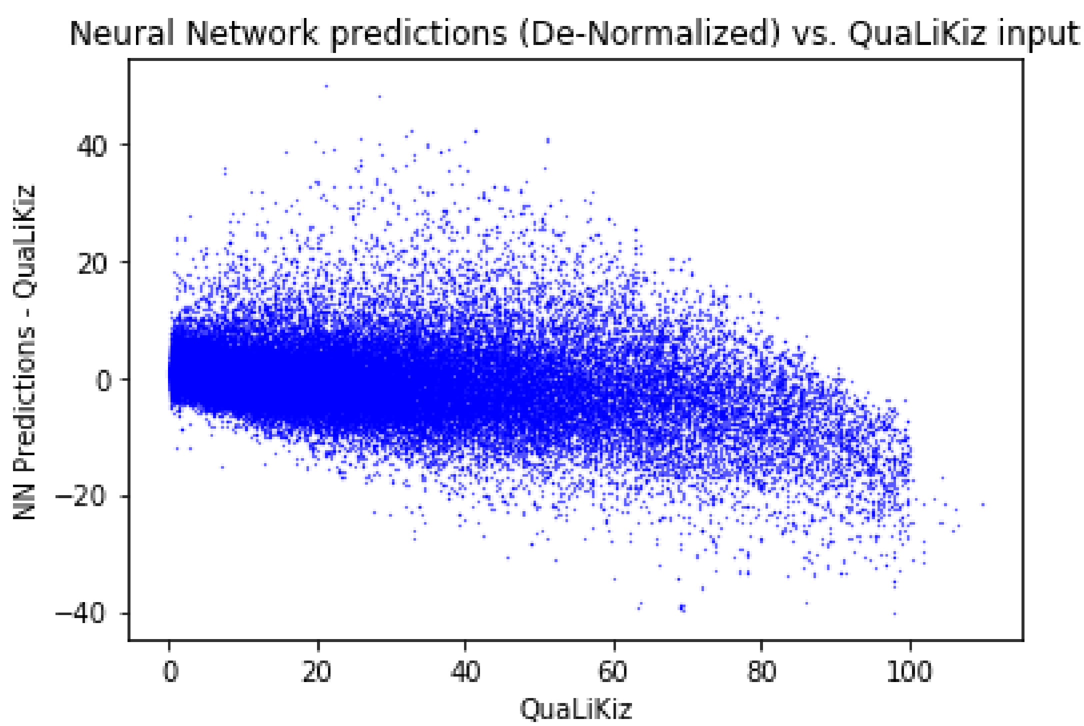
```
In [32]: plt.plot((numpy.array(h_deNormalized[1][1:]) + numpy.array(h_deNormalized[1][:-1]))/2.,
         plt.show()
```



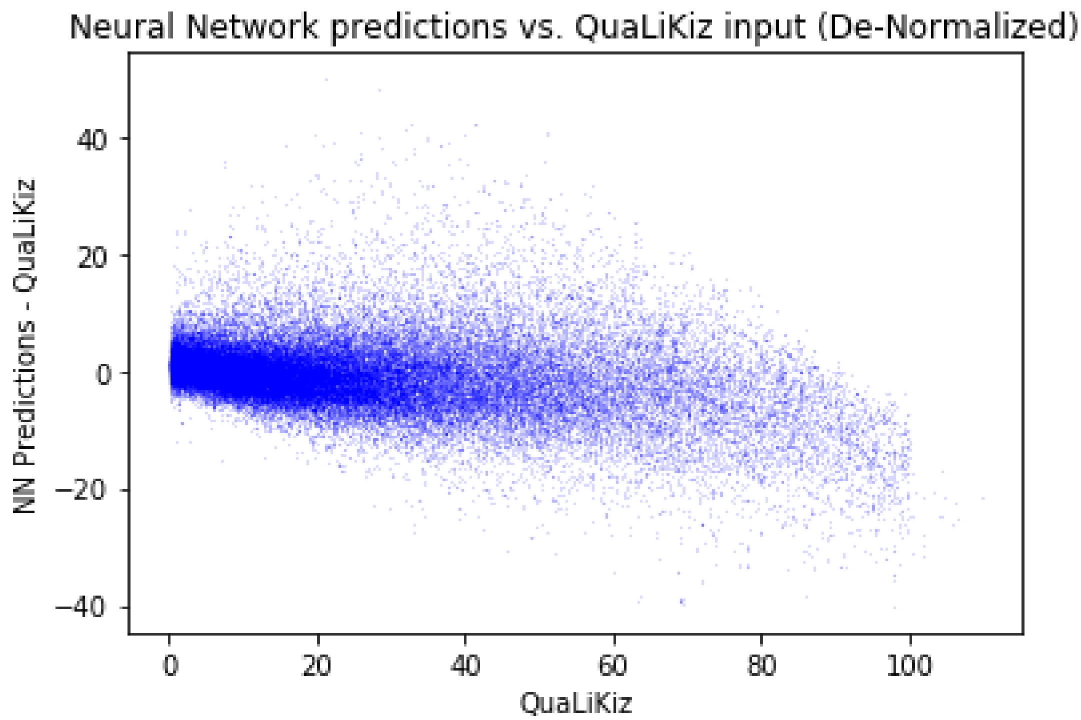

```
In [33]: for i in range(10):
             plt.plot((h_deNormalized[1][1:] + h_deNormalized[1][:-1])/2., h_deNormalized[0][i]/
```




```
In [34]: plt.plot(y_test_deNormalized_np_array, predictions_global_deNormalized - y_test_deNorma
         plt.title('Neural Network predictions (De-Normalized) vs. QuaLiKiz input')
         plt.xlabel('QuaLiKiz')
         plt.ylabel('NN Predictions - QuaLiKiz')
         plt.show()
```



```
In [35]: plt.plot(y_test_deNormalized_np_array, predictions_global_deNormalized - y_test_deNorma
         plt.title('Neural Network predictions vs. QuaLiKiz input (De-Normalized)')
         plt.xlabel('QuaLiKiz')
         plt.ylabel('NN Predictions - QuaLiKiz')
         plt.show()
```



4 Predictions (single Data Slice)

4.0.1 Predictions - Initialise table to feed predictions

```
In [36]: # table (Original values from input file unstable_training_gen2_7D_nions0_flat_filter7.
         # Hard-coded variables for the data slice

         table = numpy.zeros((200,7))

         table[:,0] = 5.75     # Ati
         table[:,1] = numpy.linspace(2,14,200)   # Ate
         table[:,2] = 3        # An
         table[:,3] = 3        # qx
         table[:,4] = 0.7      # smag
```

```
table[:,5] = 0.45    # x
table[:,6] = 1.33    # Ti_Te

table
```

```
Out[36]: array([[ 5.75      ,  2.        ,  3.        , ...,  0.7       ,
                  0.45      ,  1.33      ],
                [ 5.75      ,  2.06030151,  3.        , ...,  0.7       ,
                  0.45      ,  1.33      ],
                [ 5.75      ,  2.12060302,  3.        , ...,  0.7       ,
                  0.45      ,  1.33      ],
                ...,
                [ 5.75      , 13.87939698,  3.        , ...,  0.7       ,
                  0.45      ,  1.33      ],
                [ 5.75      , 13.93969849,  3.        , ...,  0.7       ,
                  0.45      ,  1.33      ],
                [ 5.75      , 14.        ,  3.        , ...,  0.7       ,
                  0.45      ,  1.33      ]])
```

```
In [37]: # Normalized table (inputs for model.predict())
         table_normalized = numpy.zeros((200,7))

         DataSlice_Ati = 5.75
         DataSlice_Ate = numpy.linspace(2,14,200)
         DataSlice_An = 3
         DataSlice_qx = 3
         DataSlice_smag = 0.7
         DataSlice_x = 0.45
         DataSlice_Ti_Te = 1.33

         # Normalize data by standard deviation and mean-centering the data
         table_normalized[:,0] = (DataSlice_Ati - joined_dataFrame_original['Ati'].mean()) / joi
         table_normalized[:,1] = (DataSlice_Ate - joined_dataFrame_original['Ate'].mean()) / joi
         table_normalized[:,2] = (DataSlice_An - joined_dataFrame_original['An'].mean()) / joine
         table_normalized[:,3] = (DataSlice_qx - joined_dataFrame_original['qx'].mean()) / joine
         table_normalized[:,4] = (DataSlice_smag - joined_dataFrame_original['smag'].mean()) / j
         table_normalized[:,5] = (DataSlice_x - joined_dataFrame_original['x'].mean()) / joined_
         table_normalized[:,6] = (DataSlice_Ti_Te - joined_dataFrame_original['Ti_Te'].mean()) /
```

```
In [38]: table_normalized
```

```
Out[38]: array([[ 0.10547058, -2.15152557,  0.54603754, ...,  0.20359268,
                 -0.118598  , -0.06698023],
                [ 0.10547058, -2.13159539,  0.54603754, ...,  0.20359268,
                 -0.118598  , -0.06698023],
                [ 0.10547058, -2.11166521,  0.54603754, ...,  0.20359268,
                 -0.118598  , -0.06698023],
                ...,
                [ 0.10547058,  1.77471936,  0.54603754, ...,  0.20359268,
```

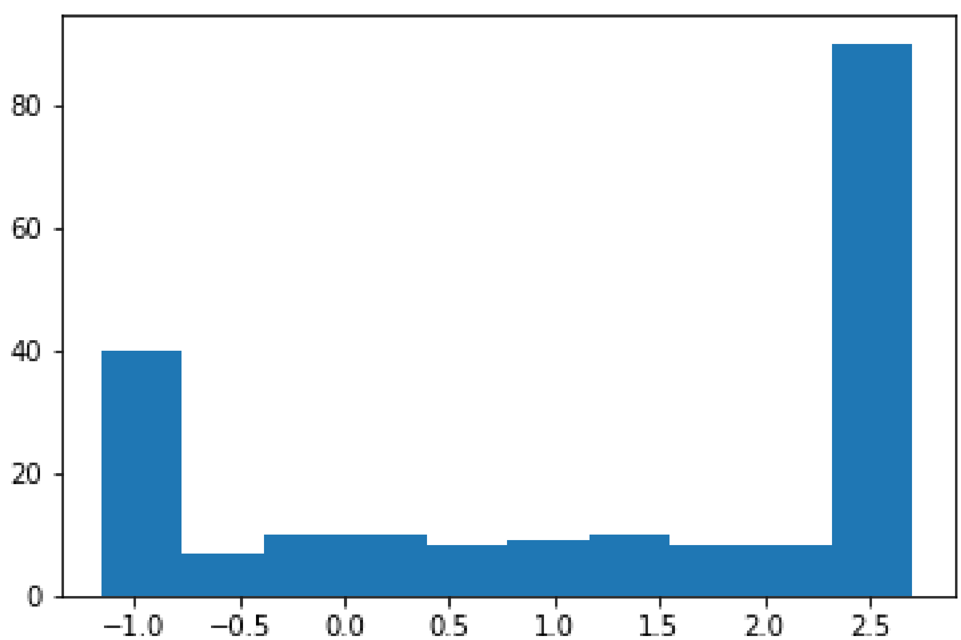
```
        -0.118598  , -0.06698023],
       [ 0.10547058,  1.79464954,  0.54603754, ...,  0.20359268,
        -0.118598  , -0.06698023],
       [ 0.10547058,  1.81457971,  0.54603754, ...,  0.20359268,
        -0.118598  , -0.06698023]])
```

In [39]:
```
# joined_dataFrame.to_csv('./out_joined_dataFrame.csv', encoding='utf-8')
```

In [40]:
```
predictions = new_model.predict(table_normalized, batch_size = 10, verbose=0)
print(type(predictions))

predictions = predictions.flatten()
print(predictions.shape)
print(type(predictions))
```

```
<class 'numpy.ndarray'>
(200,)
<class 'numpy.ndarray'>
```

In [41]:
```
plt.hist(predictions)
plt.show()
```



5 Masks

These are used to rapidly select which parameters to look at (for a given data slice).

Ate

```
In [42]: Ate_mask1 = joined_dataFrame_original.Ate == 2
         Ate_mask2 = joined_dataFrame_original.Ate == 2.75
         Ate_mask3 = joined_dataFrame_original.Ate == 3.5
         Ate_mask4 = joined_dataFrame_original.Ate == 4.25
         Ate_mask5 = joined_dataFrame_original.Ate == 5
         Ate_mask6 = joined_dataFrame_original.Ate == 5.75
         Ate_mask7 = joined_dataFrame_original.Ate == 6.5
         Ate_mask8 = joined_dataFrame_original.Ate == 7.25
         Ate_mask9 = joined_dataFrame_original.Ate == 8
         Ate_mask10 = joined_dataFrame_original.Ate == 10
         Ate_mask11 = joined_dataFrame_original.Ate == 14

         Ate_anti_mask1 = joined_dataFrame_original.Ate != 2
         Ate_anti_mask2 = joined_dataFrame_original.Ate != 2.75
         Ate_anti_mask3 = joined_dataFrame_original.Ate != 3.5
         Ate_anti_mask4 = joined_dataFrame_original.Ate != 4.25
         Ate_anti_mask5 = joined_dataFrame_original.Ate != 5
         Ate_anti_mask6 = joined_dataFrame_original.Ate != 5.75
         Ate_anti_mask7 = joined_dataFrame_original.Ate != 6.5
         Ate_anti_mask8 = joined_dataFrame_original.Ate != 7.25
         Ate_anti_mask9 = joined_dataFrame_original.Ate != 8
         Ate_anti_mask10 = joined_dataFrame_original.Ate != 10
         Ate_anti_mask11 = joined_dataFrame_original.Ate != 14
```

An

```
In [43]: An_mask1 = joined_dataFrame_original.An == -5
         An_mask2 = joined_dataFrame_original.An == -3
         An_mask3 = joined_dataFrame_original.An == -1
         An_mask4 = numpy.array(joined_dataFrame_original.An <= 1.1e-14) * numpy.array(joined_da
         An_mask5 = joined_dataFrame_original.An == 0.5
         An_mask6 = joined_dataFrame_original.An == 1.0
         An_mask7 = joined_dataFrame_original.An == 1.5
         An_mask8 = joined_dataFrame_original.An == 2.0
         An_mask9 = joined_dataFrame_original.An == 2.5
         An_mask10 = joined_dataFrame_original.An == 3.0
         An_mask11 = joined_dataFrame_original.An == 4.0
         An_mask12 = joined_dataFrame_original.An == 6.0

         An_anti_mask1 = joined_dataFrame_original.An != -5
         An_anti_mask2 = joined_dataFrame_original.An != -3
         An_anti_mask3 = joined_dataFrame_original.An != -1
         An_anti_mask4 = numpy.array(joined_dataFrame_original.An >= 1.1e-14) * numpy.array(join
         An_anti_mask5 = joined_dataFrame_original.An != 0.5
         An_anti_mask6 = joined_dataFrame_original.An != 1.0
         An_anti_mask7 = joined_dataFrame_original.An != 1.5
```

```
An_anti_mask8 = joined_dataFrame_original.An != 2.0
An_anti_mask9 = joined_dataFrame_original.An != 2.5
An_anti_mask10 = joined_dataFrame_original.An != 3.0
An_anti_mask11 = joined_dataFrame_original.An != 4.0
An_anti_mask12 = joined_dataFrame_original.An != 6.0
```

Ati

```
In [44]: Ati_mask1 = numpy.array(joined_dataFrame_original.Ati <= 1.1e-14) * numpy.array(joined_
         Ati_mask2 = joined_dataFrame_original.Ati == 2
         Ati_mask3 = joined_dataFrame_original.Ati == 2.75
         Ati_mask4 = joined_dataFrame_original.Ati == 3.5
         Ati_mask5 = joined_dataFrame_original.Ati == 4.25
         Ati_mask6 = joined_dataFrame_original.Ati == 5
         Ati_mask7 = joined_dataFrame_original.Ati == 5.75
         Ati_mask8 = joined_dataFrame_original.Ati == 6.5
         Ati_mask9 = joined_dataFrame_original.Ati == 7.25
         Ati_mask10 = joined_dataFrame_original.Ati == 8
         Ati_mask11 = joined_dataFrame_original.Ati == 10
         Ati_mask12 = joined_dataFrame_original.Ati == 14
```

qx

```
In [45]: qx_mask0 = numpy.array(joined_dataFrame_original.qx <= 0.67) * numpy.array(joined_dataF
         qx_mask1 = joined_dataFrame_original.qx == 1.0
         qx_mask2 = joined_dataFrame_original.qx == 1.5
         qx_mask3 = joined_dataFrame_original.qx == 2.0
         qx_mask4 = joined_dataFrame_original.qx == 2.5
         qx_mask5 = joined_dataFrame_original.qx == 3.0
         qx_mask6 = joined_dataFrame_original.qx == 4.0
         qx_mask7 = joined_dataFrame_original.qx == 5.0
         qx_mask8 = joined_dataFrame_original.qx == 10.00
         qx_mask9 = joined_dataFrame_original.qx == 15.00

         qx_anti_mask0 = numpy.array(joined_dataFrame_original.qx >= 0.67) * numpy.array(joined_
         qx_anti_mask1 = joined_dataFrame_original.qx != 1.0
         qx_anti_mask2 = joined_dataFrame_original.qx != 1.5
         qx_anti_mask3 = joined_dataFrame_original.qx != 2.0
         qx_anti_mask4 = joined_dataFrame_original.qx != 2.5
         qx_anti_mask5 = joined_dataFrame_original.qx != 3.0
         qx_anti_mask6 = joined_dataFrame_original.qx != 4.0
         qx_anti_mask7 = joined_dataFrame_original.qx != 5.0
         qx_anti_mask8 = joined_dataFrame_original.qx != 10.00
         qx_anti_mask9 = joined_dataFrame_original.qx != 15.00
```

smag

```
In [46]: smag_mask1 = joined_dataFrame_original.smag == -1.0
         smag_mask2 = numpy.array(joined_dataFrame_original.smag <= 0.11) * numpy.array(joined_d
```

```
smag_mask3 = numpy.array(joined_dataFrame_original.smag <= 0.41) * numpy.array(joined_d
smag_mask4 = numpy.array(joined_dataFrame_original.smag <= 0.71) * numpy.array(joined_d
smag_mask5 = joined_dataFrame_original.smag == 1
smag_mask6 = joined_dataFrame_original.smag == 1.5
smag_mask7 = joined_dataFrame_original.smag == 2.0
smag_mask8 = joined_dataFrame_original.smag == 2.75
smag_mask9 = joined_dataFrame_original.smag == 3.5
smag_mask10 = joined_dataFrame_original.smag == 5.0

smag_anti_mask1 = joined_dataFrame_original.smag != -1.0
smag_anti_mask2 = numpy.array(joined_dataFrame_original.smag >= 0.11) * numpy.array(joi
smag_anti_mask3 = numpy.array(joined_dataFrame_original.smag >= 0.41) * numpy.array(joi
smag_anti_mask4 = numpy.array(joined_dataFrame_original.smag >= 0.71) * numpy.array(joi
smag_anti_mask5 = joined_dataFrame_original.smag != 1
smag_anti_mask6 = joined_dataFrame_original.smag != 1.5
smag_anti_mask7 = joined_dataFrame_original.smag != 2.0
smag_anti_mask8 = joined_dataFrame_original.smag != 2.75
smag_anti_mask9 = joined_dataFrame_original.smag != 3.5
smag_anti_mask10 = joined_dataFrame_original.smag != 5.0
```

x

```
In [47]: x_mask1 = numpy.array(joined_dataFrame_original.x <= 0.10) * numpy.array(joined_dataFra
         x_mask2 = numpy.array(joined_dataFrame_original.x <= 0.22) * numpy.array(joined_dataFra
         x_mask3 = numpy.array(joined_dataFrame_original.x <= 0.34) * numpy.array(joined_dataFra
         x_mask4 = numpy.array(joined_dataFrame_original.x <= 0.46) * numpy.array(joined_dataFra
         x_mask5 = numpy.array(joined_dataFrame_original.x <= 0.58) * numpy.array(joined_dataFra
         x_mask6 = numpy.array(joined_dataFrame_original.x <= 0.70) * numpy.array(joined_dataFra
         x_mask7 = numpy.array(joined_dataFrame_original.x <= 0.85) * numpy.array(joined_dataFra
         x_mask8 = numpy.array(joined_dataFrame_original.x <= 1.00) * numpy.array(joined_dataFra

         x_anti_mask1 = numpy.array(joined_dataFrame_original.x >= 0.10) * numpy.array(joined_da
         x_anti_mask2 = numpy.array(joined_dataFrame_original.x >= 0.22) * numpy.array(joined_da
         x_anti_mask3 = numpy.array(joined_dataFrame_original.x >= 0.34) * numpy.array(joined_da
         x_anti_mask4 = numpy.array(joined_dataFrame_original.x >= 0.46) * numpy.array(joined_da
         x_anti_mask5 = numpy.array(joined_dataFrame_original.x >= 0.58) * numpy.array(joined_da
         x_anti_mask6 = numpy.array(joined_dataFrame_original.x >= 0.70) * numpy.array(joined_da
         x_anti_mask7 = numpy.array(joined_dataFrame_original.x >= 0.85) * numpy.array(joined_da
         x_anti_mask8 = numpy.array(joined_dataFrame_original.x >= 1.00) * numpy.array(joined_da
```

Ti_Te

```
In [48]: Ti_Te_mask1 = joined_dataFrame_original.Ti_Te == 0.25
         Ti_Te_mask2 = joined_dataFrame_original.Ti_Te == 0.5
         Ti_Te_mask3 = joined_dataFrame_original.Ti_Te == 0.75
         Ti_Te_mask4 = joined_dataFrame_original.Ti_Te == 1
         Ti_Te_mask5 = numpy.array(joined_dataFrame_original.Ti_Te <= 1.34) * numpy.array(joined
         Ti_Te_mask6 = numpy.array(joined_dataFrame_original.Ti_Te <= 1.67) * numpy.array(joined
         Ti_Te_mask7 = joined_dataFrame_original.Ti_Te == 2.50
```

6 Plots

```
In [49]: '''
         {'An': 3,
         'Ati': 5.75,
         'Ti_Te': 1.33,
         'q': 3,
         'smag': 0.7,
         'x': 0.45}
         '''

         newDF = joined_dataFrame[An_mask10 & Ati_mask7 & qx_mask5 & smag_mask4 & x_mask4 & Ti_T
         newDF_Mk2 = joined_dataFrame_original[An_mask10 & Ati_mask7 & qx_mask5 & smag_mask4 & x
         print(newDF)
         print(type(newDF))

         print(newDF_Mk2)
         print(type(newDF_Mk2))
```

```
           efeETG_GB       Ati       Ate        An        qx      smag  \
30963559  -0.813345  0.105471 -1.407881  0.546038 -0.320359  0.203593
31366759  -0.556873  0.105471 -1.159999  0.546038 -0.320359  0.203593
32576359   1.644625  0.105471 -0.416355  0.546038 -0.320359  0.203593

                 x    Ti_Te
30963559 -0.118598 -0.06698
31366759 -0.118598 -0.06698
32576359 -0.118598 -0.06698
<class 'pandas.core.frame.DataFrame'>
          efeETG_GB   Ati   Ate   An   qx  smag     x  Ti_Te
30963559   7.413384  5.75  4.25  3.0  3.0   0.7  0.45   1.33
31366759  13.286460  5.75  5.00  3.0  3.0   0.7  0.45   1.33
32576359  63.699650  5.75  7.25  3.0  3.0   0.7  0.45   1.33
<class 'pandas.core.frame.DataFrame'>
```

```
In [50]: efeETG_DF = newDF['efeETG_GB']
         efeETG_DF_np_array = efeETG_DF.values
         print(efeETG_DF_np_array)

         Ate_DF = newDF['Ate']
         Ate_DF_np_array = Ate_DF.values
         print(Ate_DF_np_array)
```

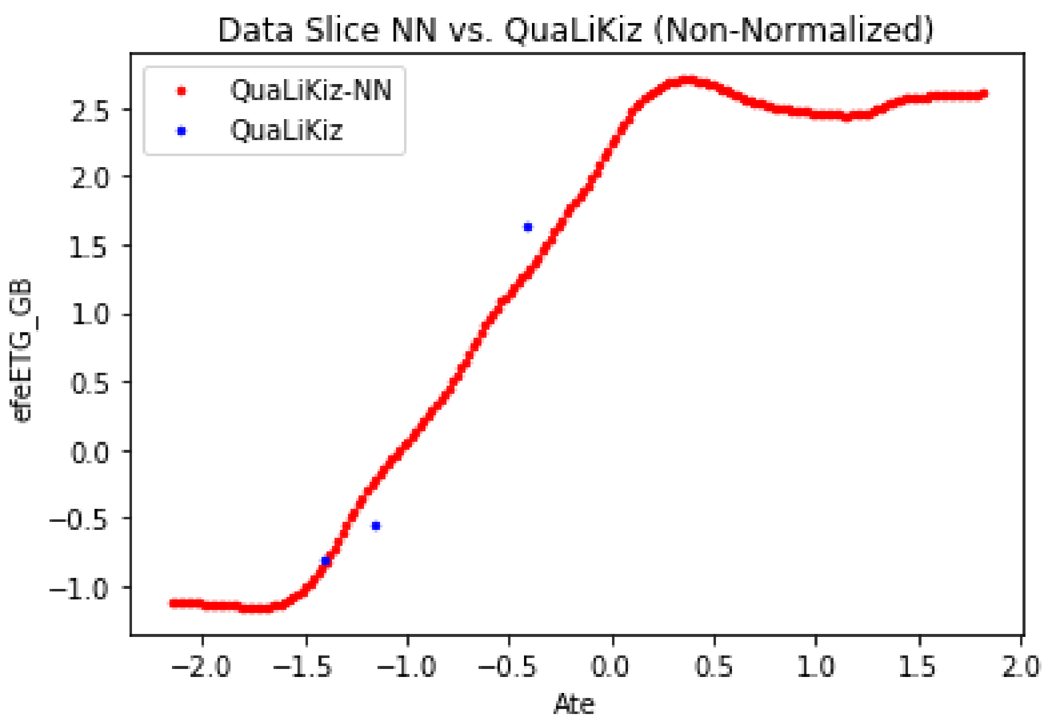
```
[-0.81334497 -0.5568731   1.64462486]
[-1.40788083 -1.15999925 -0.41635451]
```

```
In [51]: efeETG_DF_Mk2 = newDF_Mk2['efeETG_GB']
         efeETG_DF_Mk2_np_array = efeETG_DF_Mk2.values
```

```
print(efeETG_DF_Mk2_np_array)

Ate_DF_Mk2 = newDF_Mk2['Ate']
Ate_DF_Mk2_np_array = Ate_DF_Mk2.values
print(Ate_DF_Mk2_np_array)
```

```
[ 7.413384 13.28646  63.69965 ]
[4.25 5.   7.25]
```

```
In [52]: efeETG_DF = newDF['efeETG_GB']
         Ate_DF = newDF['Ate']
         plt.plot(table_normalized[:,1], predictions, 'r.', ms = 5, label = 'QuaLiKiz-NN')
         plt.plot(Ate_DF_np_array, efeETG_DF_np_array, 'b.', ms = 5, label = 'QuaLiKiz')
         plt.title('Data Slice NN vs. QuaLiKiz (Non-Normalized)')
         plt.xlabel('Ate')
         plt.ylabel('efeETG_GB')
         plt.legend()
         plt.show()
```




```
In [53]: predictions_deNormalized = (predictions * joined_dataFrame_original['efeETG_GB'].std())
         print(predictions_deNormalized.shape)
```

```
(200,)
```

In [54]:
```
plt.plot(table[:,1], predictions_deNormalized, 'r.', ms = 5, label = 'QuaLiKiz-NN')
plt.plot(Ate_DF_Mk2_np_array, efeETG_DF_Mk2_np_array, 'b.', ms = 5, label = 'QuaLiKiz')
plt.title('Data Slice NN vs. QuaLiKiz (De-Normalized)')
plt.xlabel('Ate')
plt.ylabel('efeETG_GB')
plt.legend()
# plt.savefig('./2018-07-17_TableTestingNotebook_Mk2/NN_Predictions.png', dpi = 100)
plt.show()
```