Project work

Balogh Nóra, Gyulai László, Vargha Noémi

In this project we have implemented 2 dimensional Ising model with Monte-Carlo simulation. We aimed to train neural networks based on the simulation results. The main parts of our work:

1. Implementing Monte-Carlo simulation code for the Ising model
2. Generating training data
3. Neural Network learns to calculate energy and magnetization
4. Neural Network learns to calculate coupling strength

Ideas:

[https://arxiv.org/pdf/1706.09779.pdf (https://arxiv.org/pdf/1706.09779.pdf)](https://arxiv.org/pdf/1706.09779.pdf)

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import csv
from numpy.random import rand
from numpy.random import randint
```

In [2]:
```
import random
```

In [3]:
```
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers.core import Dense
from keras.optimizers import SGD, Adam
from sklearn.preprocessing import MinMaxScaler
```

```
Using TensorFlow backend.
```

In [4]:
```
from sklearn.metrics import mean_squared_error
```

1. Monte-Carlo simulation code for the Ising model

We used a square lattice od size LxL=10x10. Each cell in the square lattice has spin $\sigma_i = \pm 1$ which is initialized randomly. The Hamiltonian of the system is:

$$H = -K \sum_{\langle i,j \rangle} \sigma_i \sigma_J - h \sum_i \sigma_i$$

where the summation runs over the nearest neighbors, K is coupling strength between the nearest neighbors and h is the external field.

assiged each lattice cell a random spin of ± 1 with the function below:

Parameters of the 2D Ising model:

- hi: interaction with external field
- Kij: interactions between neighbours

In [5]:
```
def initialize(L):
    ''' generates a random spin (+1 or -1) configuration for initial conditio
n'''
    state = 2*np.random.randint(2, size=(L,L))-1
    return state #square lattice, cells filled randomly with +1 or -1
```

Magnetization is the average value of the spin. (Source: [https://en.wikipedia.org/wiki/Ising_model (https://en.wikipedia.org/wiki/Ising_model)](https://en.wikipedia.org/wiki/Ising_model)). The energy of a configuration is calculated from the Hamiltonian.

In [6]:
```
def magnetization(config):
    '''Magnetization of a given configuration'''
    mag = np.sum(config)/(config.shape[0]*config.shape[1])
    return mag

def Energy(config, h, K):
    '''Calculates energy of a given configuration'''
    energy = 0
    L=len(config)
    for i in range(L):
        for j in range(L):
            S = config[i,j]
            energy-=h*S
            #neighbors with periodic boundary conditions
            neighbors = config[(i+1)%L, j] + config[i,(j+1)%L] + config[(i-1)%
L, j] + config[i,(j-1)%L]
            energy -= K*neighbors*S
    return energy
```

One Monte Carlo timestep consists of LxL elementary step in which a spin is chosen randomly and flipped according to the Metropolis probabilities.

In each elementary step we randomly choose a lattice point with spin s. If the spin is flipped, the cost is $\Delta E = E_{flipped} - E_{original}$. From the Hamiltonian:

$$\Delta E = 2hs - 2Ks \cdot nb$$

where nb is the sum of the spins of the nearest neighbors of s. The function "mcmove" below calculates this cost and flips the spin if $\Delta E < 0$ and flips this spin with probability $p = \exp(\beta \cdot \Delta E)$ if $E > 0$. ($\beta = \frac{1}{k_b T}$)

In [7]:
```
def mcmove(config, beta, h, K):
    '''Monte Carlo move using Metropolis algorithm '''
    L=len(config)
    for i in range(L):
        for j in range(L):
                a = np.random.randint(0, L)
                b = np.random.randint(0, L)
                s =  config[a, b]
                neighbors = config[(a+1)%L,b] + config[a,(b+1)%L] + config[(a-
1)%L,b] + config[a,(b-1)%L]
                #cost=difference between energies
                cost = 2*h*s + 2*K*neighbors*s
                if cost < 0:
                    s *= -1
                elif np.random.rand() < np.exp(-cost*beta):
                    s *= -1
                config[a, b] = s
    return config
```

Use Monte carlo step as: config=mcmove(config, beta, h, K)

2. Training data generation

2.1 Generation of lattices for energy and magnetization prediction

In [8]:
```
K=1
beta=0.2
L=10
h=5
timesteps = 500
iterations = 20
config_data=[]


for i in range(iterations):
    config=initialize(L)
    E,M=np.zeros(timesteps), np.zeros(timesteps)
    M0=np.zeros(timesteps)
    for t in range(timesteps):
        E[t]=Energy(config, h, K)
        M[t]=magnetization(config)
        row1 = [E[t]]+[M[t]]+list(config.flatten())
        config_data.append(row1)
        config=mcmove(config, beta, h, K)
```

In [9]:
```
len(config_data)
```

Out[9]:
```
10000
```

We will store in a .csv file the following features of each generated lattice:

- energy
- magnetization
- config: the lattice configuration (array of +1, -1 values)

In [10]:
```
with open('training_data.csv', 'w+', newline='') as writeFile:
    writer = csv.writer(writeFile)
    writer.writerows(config_data)
writeFile.close()
```

2.2. Training data generation for coupling strength prediction

In each iteration we use different coupling strength and step 10 Monte-Carlo steps (timesteps) with it. Temperature and external field values are fixed.

In [11]:
```
K=1
beta=0.2
L=10
h=5
timesteps = 10
iterations = 5000
config_data=[]

E,M=np.zeros(iterations), np.zeros(iterations)
#Klist=[j/1000 for j in range(iterations)]
for i in range(iterations):
    K=np.random.normal()+1  # k is a random value between 0 and 2
    config=initialize(L)
    for t in range(timesteps):
        config=mcmove(config, beta, h, K)
    E[i]=Energy(config, h, K)
    M[i]=magnetization(config)
    row1 = [K]+[E[i]]+[M[i]]+list(config.flatten())
    config_data.append(row1)
```

In [12]:
```
len(config_data)
```

Out[12]: 5000

In [13]:
```
with open('training_data2.csv', 'w+', newline='') as writeFile:
    writer = csv.writer(writeFile)
    writer.writerows(config_data)
writeFile.close()
```

3. Neural Network learns to calculate energy and magnetization

3.1. Loading the data

In [14]:
```
traindata = pd.read_csv("training_data.csv", names=["E"]+["M"]+[i for i in ran
ge(L*L)])
```

In [15]:
```
trainlabelsE=traindata["E"].values
trainlabelsM=traindata["M"].values
```

In [16]:
```
train_attrs=traindata.drop(columns=["E", "M"])
```

3.2. Magnetization prediction

Normalization of training labels:

In [17]:
```
max(trainlabelsM)
```

Out[17]:
```
1.0
```

In [18]:
```
min(trainlabelsM)
```

Out[18]:
```
-0.22
```

In [19]:
```
trainlabelsM
```

Out[19]:
```
array([-0.04,  0.44,  0.72, ...,  0.92,  0.96,  0.92])
```

In [20]:
```
#trainlabelsM=(trainlabelsM-(-1))/(1-(-1))
trainlabelsM=(trainlabelsM-min(trainlabelsM))/(max(trainlabelsM)-min(trainlabelsM))
```

Split train and test values:

In [21]:
```
X_train, X_test, Y_train, Y_test = train_test_split(train_attrs, trainlabelsM, test_size=0.33, random_state=0)
```

In [22]:
```
X_train.shape, Y_train.shape, X_test.shape, Y_test.shape
```

Out[22]:
```
((6700, 100), (6700,), (3300, 100), (3300,))
```

Definition of the model:

In [23]:
```
model = Sequential()
model.add(Dense(100, input_dim=100, activation='relu'))
model.add(Dense(1, activation='relu'))
```

```
WARNING:tensorflow:From C:\ProgramData\Anaconda3\lib\site-packages\tensorflow\python\framework\op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
```

In [24]:
```
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 100)               10100
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 101
=================================================================
Total params: 10,201
Trainable params: 10,201
Non-trainable params: 0
_________________________________________________________________
```

In [25]:
```
model.compile(loss='mean_squared_error', optimizer='adam')
```

In [26]:
```
history = model.fit(x=X_train, y=Y_train, batch_size=32, epochs=100, validatio
n_data=(X_test, Y_test))
```

```
WARNING:tensorflow:From C:\ProgramData\Anaconda3\lib\site-packages\tensorflow
\python\ops\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops)
is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 6700 samples, validate on 3300 samples
Epoch 1/100
6700/6700 [==============================] - 0s 43us/step - loss: 0.0463 - va
l_loss: 0.0083
Epoch 2/100
6700/6700 [==============================] - 0s 21us/step - loss: 0.0039 - va
l_loss: 0.0020
Epoch 3/100
6700/6700 [==============================] - 0s 22us/step - loss: 0.0014 - va
l_loss: 0.0018
Epoch 4/100
6700/6700 [==============================] - 0s 21us/step - loss: 7.8930e-04
- val_loss: 8.3094e-04
Epoch 5/100
6700/6700 [==============================] - 0s 21us/step - loss: 5.0175e-04
- val_loss: 6.3748e-04
Epoch 6/100
6700/6700 [==============================] - 0s 22us/step - loss: 3.5736e-04
- val_loss: 0.0010
Epoch 7/100
6700/6700 [==============================] - 0s 22us/step - loss: 2.9739e-04
- val_loss: 5.2830e-04
Epoch 8/100
6700/6700 [==============================] - 0s 21us/step - loss: 3.0750e-04
- val_loss: 8.8820e-04
Epoch 9/100
6700/6700 [==============================] - 0s 21us/step - loss: 3.2129e-04
- val_loss: 3.5737e-04
Epoch 10/100
6700/6700 [==============================] - 0s 21us/step - loss: 2.7184e-04
- val_loss: 2.7840e-04
Epoch 11/100
6700/6700 [==============================] - 0s 21us/step - loss: 2.3572e-04
- val_loss: 3.6024e-04
Epoch 12/100
6700/6700 [==============================] - 0s 21us/step - loss: 3.4369e-04
- val_loss: 2.7300e-04
Epoch 13/100
6700/6700 [==============================] - 0s 21us/step - loss: 2.3496e-04
- val_loss: 2.7163e-04
Epoch 14/100
6700/6700 [==============================] - 0s 20us/step - loss: 2.9052e-04
- val_loss: 2.6775e-04
Epoch 15/100
6700/6700 [==============================] - 0s 20us/step - loss: 2.4227e-04
- val_loss: 3.2567e-04
Epoch 16/100
6700/6700 [==============================] - 0s 20us/step - loss: 2.2727e-04
- val_loss: 2.9877e-04
Epoch 17/100
6700/6700 [==============================] - 0s 20us/step - loss: 2.5215e-04
- val_loss: 3.4135e-04
```

```
Epoch 18/100
6700/6700 [==============================] - 0s 20us/step - loss: 2.8019e-04
- val_loss: 3.7209e-04
Epoch 19/100
6700/6700 [==============================] - 0s 21us/step - loss: 3.1938e-04
- val_loss: 9.5445e-04
Epoch 20/100
6700/6700 [==============================] - 0s 21us/step - loss: 4.5812e-04
- val_loss: 0.0035
Epoch 21/100
6700/6700 [==============================] - 0s 20us/step - loss: 4.5160e-04
- val_loss: 0.0010
Epoch 22/100
6700/6700 [==============================] - 0s 21us/step - loss: 3.8592e-04
- val_loss: 7.8077e-04
Epoch 23/100
6700/6700 [==============================] - 0s 21us/step - loss: 3.2499e-04
- val_loss: 2.8076e-04
Epoch 24/100
6700/6700 [==============================] - 0s 20us/step - loss: 1.8243e-04
- val_loss: 2.0966e-04
Epoch 25/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.9117e-04
- val_loss: 2.1833e-04
Epoch 26/100
6700/6700 [==============================] - 0s 21us/step - loss: 2.4516e-04
- val_loss: 1.3320e-04
Epoch 27/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.9525e-04
- val_loss: 1.2842e-04
Epoch 28/100
6700/6700 [==============================] - 0s 20us/step - loss: 2.7944e-04
- val_loss: 5.5456e-04
Epoch 29/100
6700/6700 [==============================] - 0s 23us/step - loss: 1.9831e-04
- val_loss: 1.2281e-04
Epoch 30/100
6700/6700 [==============================] - 0s 26us/step - loss: 3.0323e-04
- val_loss: 0.0010
Epoch 31/100
6700/6700 [==============================] - 0s 22us/step - loss: 6.8180e-04
- val_loss: 1.0396e-04
Epoch 32/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.7372e-04
- val_loss: 1.1056e-04
Epoch 33/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.4584e-04
- val_loss: 8.8586e-05
Epoch 34/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.5432e-04
- val_loss: 2.1574e-04
Epoch 35/100
6700/6700 [==============================] - 0s 20us/step - loss: 2.9761e-04
- val_loss: 3.1065e-04
Epoch 36/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.8416e-04
- val_loss: 8.4911e-05
```

```
Epoch 37/100
6700/6700 [==============================] - 0s 20us/step - loss: 1.8578e-04
- val_loss: 1.6141e-04
Epoch 38/100
6700/6700 [==============================] - 0s 20us/step - loss: 2.8653e-04
- val_loss: 1.0694e-04
Epoch 39/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.6468e-04
- val_loss: 8.9866e-05
Epoch 40/100
6700/6700 [==============================] - 0s 21us/step - loss: 2.0375e-04
- val_loss: 2.1837e-04
Epoch 41/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.6581e-04
- val_loss: 8.2248e-05
Epoch 42/100
6700/6700 [==============================] - 0s 21us/step - loss: 2.1573e-04
- val_loss: 3.7585e-04
Epoch 43/100
6700/6700 [==============================] - 0s 20us/step - loss: 1.4919e-04
- val_loss: 8.1330e-05
Epoch 44/100
6700/6700 [==============================] - 0s 20us/step - loss: 2.3774e-04
- val_loss: 4.7765e-04
Epoch 45/100
6700/6700 [==============================] - 0s 21us/step - loss: 2.0250e-04
- val_loss: 1.1913e-04
Epoch 46/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.4944e-04
- val_loss: 6.2397e-04
Epoch 47/100
6700/6700 [==============================] - 0s 21us/step - loss: 9.1264e-04
- val_loss: 7.4391e-05
Epoch 48/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.0002e-04
- val_loss: 9.8653e-05
Epoch 49/100
6700/6700 [==============================] - 0s 20us/step - loss: 9.7960e-05
- val_loss: 8.2893e-05
Epoch 50/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.0362e-04
- val_loss: 7.7520e-05
Epoch 51/100
6700/6700 [==============================] - 0s 20us/step - loss: 1.1123e-04
- val_loss: 2.7200e-04
Epoch 52/100
6700/6700 [==============================] - 0s 20us/step - loss: 1.6272e-04
- val_loss: 7.5300e-05
Epoch 53/100
6700/6700 [==============================] - 0s 20us/step - loss: 1.6597e-04
- val_loss: 8.5988e-05
Epoch 54/100
6700/6700 [==============================] - 0s 20us/step - loss: 1.6370e-04
- val_loss: 7.7600e-05
Epoch 55/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.6647e-04
- val_loss: 1.3906e-04
```

```
Epoch 56/100
6700/6700 [==============================] - 0s 25us/step - loss: 1.8304e-04
- val_loss: 2.1045e-04
Epoch 57/100
6700/6700 [==============================] - 0s 23us/step - loss: 1.7962e-04
- val_loss: 1.5767e-04
Epoch 58/100
6700/6700 [==============================] - 0s 22us/step - loss: 1.4457e-04
- val_loss: 7.4169e-05
Epoch 59/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.6310e-04
- val_loss: 1.3674e-04
Epoch 60/100
6700/6700 [==============================] - 0s 20us/step - loss: 1.8363e-04
- val_loss: 1.8015e-04
Epoch 61/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.8325e-04
- val_loss: 1.3435e-04
Epoch 62/100
6700/6700 [==============================] - 0s 20us/step - loss: 1.5545e-04
- val_loss: 1.3860e-04
Epoch 63/100
6700/6700 [==============================] - 0s 20us/step - loss: 1.1472e-04
- val_loss: 1.2414e-04
Epoch 64/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.5042e-04
- val_loss: 7.2797e-05
Epoch 65/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.4844e-04
- val_loss: 1.9304e-04
Epoch 66/100
6700/6700 [==============================] - 0s 20us/step - loss: 1.6275e-04
- val_loss: 6.9435e-05
Epoch 67/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.4293e-04
- val_loss: 1.7793e-04
Epoch 68/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.3565e-04
- val_loss: 7.3113e-05
Epoch 69/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.6522e-04
- val_loss: 1.0222e-04
Epoch 70/100
6700/6700 [==============================] - 0s 22us/step - loss: 1.8436e-04
- val_loss: 6.0167e-05
Epoch 71/100
6700/6700 [==============================] - 0s 23us/step - loss: 1.1472e-04
- val_loss: 6.3931e-05
Epoch 72/100
6700/6700 [==============================] - 0s 22us/step - loss: 1.9015e-04
- val_loss: 1.4502e-04
Epoch 73/100
6700/6700 [==============================] - 0s 25us/step - loss: 1.1350e-04
- val_loss: 5.6661e-05
Epoch 74/100
6700/6700 [==============================] - 0s 25us/step - loss: 1.2967e-04
- val_loss: 1.1325e-04
```

```
Epoch 75/100
6700/6700 [==============================] - 0s 23us/step - loss: 1.3005e-04
- val_loss: 3.5763e-04
Epoch 76/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.9418e-04
- val_loss: 6.2119e-05
Epoch 77/100
6700/6700 [==============================] - 0s 20us/step - loss: 1.1604e-04
- val_loss: 5.2118e-04
Epoch 78/100
6700/6700 [==============================] - 0s 22us/step - loss: 1.4130e-04
- val_loss: 6.4923e-05
Epoch 79/100
6700/6700 [==============================] - 0s 24us/step - loss: 1.5197e-04
- val_loss: 1.6206e-04
Epoch 80/100
6700/6700 [==============================] - 0s 22us/step - loss: 1.8274e-04
- val_loss: 1.6595e-04
Epoch 81/100
6700/6700 [==============================] - 0s 25us/step - loss: 1.0378e-04
- val_loss: 6.4206e-05
Epoch 82/100
6700/6700 [==============================] - 0s 22us/step - loss: 1.3996e-04
- val_loss: 1.1748e-04
Epoch 83/100
6700/6700 [==============================] - 0s 22us/step - loss: 1.6299e-04
- val_loss: 1.1567e-04
Epoch 84/100
6700/6700 [==============================] - 0s 22us/step - loss: 1.1758e-04
- val_loss: 9.3142e-05
Epoch 85/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.5212e-04
- val_loss: 6.7238e-05
Epoch 86/100
6700/6700 [==============================] - 0s 24us/step - loss: 1.2903e-04
- val_loss: 8.1630e-05
Epoch 87/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.4384e-04
- val_loss: 5.5154e-05
Epoch 88/100
6700/6700 [==============================] - 0s 22us/step - loss: 1.2623e-04
- val_loss: 1.6352e-04
Epoch 89/100
6700/6700 [==============================] - 0s 22us/step - loss: 1.2890e-04
- val_loss: 6.9255e-05
Epoch 90/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.3332e-04
- val_loss: 9.3086e-05
Epoch 91/100
6700/6700 [==============================] - 0s 23us/step - loss: 1.7272e-04
- val_loss: 5.8617e-05
Epoch 92/100
6700/6700 [==============================] - 0s 22us/step - loss: 1.0922e-04
- val_loss: 1.0116e-04
Epoch 93/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.3937e-04
- val_loss: 7.9088e-05
```

```
Epoch 94/100
6700/6700 [==============================] - 0s 23us/step - loss: 1.4913e-04
- val_loss: 4.3698e-04
Epoch 95/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.3673e-04
- val_loss: 9.0687e-05
Epoch 96/100
6700/6700 [==============================] - 0s 24us/step - loss: 1.1646e-04
- val_loss: 6.7394e-05
Epoch 97/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.3907e-04
- val_loss: 6.4052e-05
Epoch 98/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.2415e-04
- val_loss: 2.2615e-04
Epoch 99/100
6700/6700 [==============================] - 0s 23us/step - loss: 1.3774e-04
- val_loss: 6.1892e-05
Epoch 100/100
6700/6700 [==============================] - 0s 21us/step - loss: 1.3721e-04
- val_loss: 3.0640e-04
```

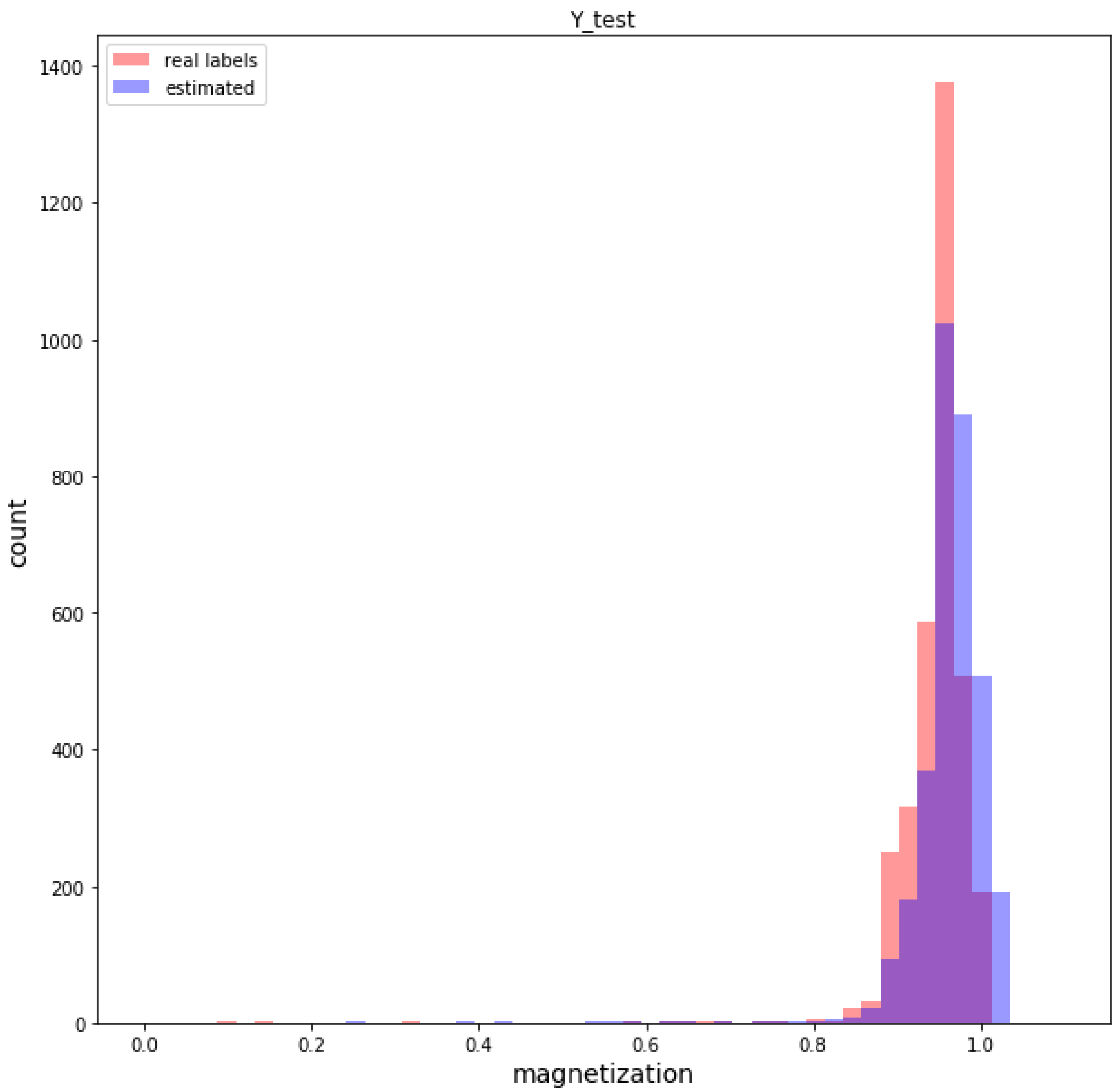
Making predictions and evaluation:

In [27]:
```
preds = model.predict(X_test)
print('MSE : ', mean_squared_error(preds, Y_test))
```

```
MSE :  0.0003063977349365151
```

In [28]:
```
plt.figure(figsize=(10,10))
Y_hist = plt.hist(Y_test, 50, (0,1.1),histtype="stepfilled",alpha= 0.4, color
='r')
plt.title("Y_test")
pred_hist = plt.hist(preds, 50, (0,1.1),histtype="stepfilled",alpha= 0.4, colo
r ='b')
plt.legend(["real labels", "estimated"], loc='upper left')
plt.xlabel("magnetization", fontsize = 14)
plt.ylabel("count",  fontsize = 14)
```

Out[28]:
```
Text(0, 0.5, 'count')
```



3.3 Energy prediction

Normalization of the labels:

In [29]:
```
max(trainlabelsE)
```
Out[29]:
```
78.0
```
In [30]:
```
min(trainlabelsE)
```
Out[30]:
```
-900.0
```
In [31]:
```
trainlabelsE=(trainlabelsE-min(trainlabelsE))/(max(trainlabelsE)-min(trainlabelsE))
```
In [32]:
```
min(trainlabelsE)
```
Out[32]:
```
0.0
```
In [33]:
```
X_trainE, X_testE, Y_trainE, Y_testE = train_test_split(train_attrs, trainlabelsE, test_size=0.33, random_state=0)
```

Definition of the model:

In [34]:
```
model2 = Sequential()
model2.add(Dense(100, input_dim=100, activation='relu'))
model2.add(Dense(50, activation='relu'))
model2.add(Dense(30, activation='relu'))
model2.add(Dense(1, activation='relu'))
```
In [35]:
```
model2.compile(loss='mean_squared_error', optimizer='adam')
```
In [36]:
```
model2.summary()
```
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_3 (Dense)              (None, 100)               10100
_________________________________________________________________
dense_4 (Dense)              (None, 50)                5050
_________________________________________________________________
dense_5 (Dense)              (None, 30)                1530
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 31
=================================================================
Total params: 16,711
Trainable params: 16,711
Non-trainable params: 0
_________________________________________________________________
```

In [37]:
```
history = model2.fit(x=X_trainE, y=Y_trainE, batch_size=64, epochs=100, valida
tion_data=(X_testE, Y_testE))
```

```
Train on 6700 samples, validate on 3300 samples
Epoch 1/100
6700/6700 [==============================] - 0s 46us/step - loss: 0.0095 - va
l_loss: 0.0086
Epoch 2/100
6700/6700 [==============================] - 0s 15us/step - loss: 0.0096 - va
l_loss: 0.0086
Epoch 3/100
6700/6700 [==============================] - 0s 15us/step - loss: 0.0086 - va
l_loss: 0.0057
Epoch 4/100
6700/6700 [==============================] - 0s 17us/step - loss: 0.0026 - va
l_loss: 9.0791e-04
Epoch 5/100
6700/6700 [==============================] - 0s 16us/step - loss: 7.2906e-04
- val_loss: 6.6341e-04
Epoch 6/100
6700/6700 [==============================] - 0s 15us/step - loss: 5.1134e-04
- val_loss: 6.1371e-04
Epoch 7/100
6700/6700 [==============================] - 0s 14us/step - loss: 3.7253e-04
- val_loss: 4.8657e-04
Epoch 8/100
6700/6700 [==============================] - 0s 15us/step - loss: 3.2988e-04
- val_loss: 4.3452e-04
Epoch 9/100
6700/6700 [==============================] - 0s 15us/step - loss: 3.0934e-04
- val_loss: 4.2270e-04
Epoch 10/100
6700/6700 [==============================] - 0s 17us/step - loss: 2.5403e-04
- val_loss: 4.6692e-04
Epoch 11/100
6700/6700 [==============================] - 0s 15us/step - loss: 2.5111e-04
- val_loss: 3.8298e-04
Epoch 12/100
6700/6700 [==============================] - 0s 14us/step - loss: 2.3678e-04
- val_loss: 3.5835e-04
Epoch 13/100
6700/6700 [==============================] - 0s 14us/step - loss: 2.5118e-04
- val_loss: 3.5113e-04
Epoch 14/100
6700/6700 [==============================] - 0s 15us/step - loss: 2.3888e-04
- val_loss: 3.3910e-04
Epoch 15/100
6700/6700 [==============================] - 0s 15us/step - loss: 2.1836e-04
- val_loss: 3.9200e-04
Epoch 16/100
6700/6700 [==============================] - 0s 15us/step - loss: 2.2053e-04
- val_loss: 3.2168e-04
Epoch 17/100
6700/6700 [==============================] - 0s 14us/step - loss: 2.0928e-04
- val_loss: 3.0969e-04
Epoch 18/100
6700/6700 [==============================] - 0s 14us/step - loss: 1.8621e-04
- val_loss: 2.9223e-04
Epoch 19/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.7350e-04
```

```
- val_loss: 3.2263e-04
Epoch 20/100
6700/6700 [==============================] - 0s 17us/step - loss: 1.7778e-04
- val_loss: 2.6235e-04
Epoch 21/100
6700/6700 [==============================] - 0s 16us/step - loss: 1.6923e-04
- val_loss: 3.0144e-04
Epoch 22/100
6700/6700 [==============================] - 0s 14us/step - loss: 2.0779e-04
- val_loss: 2.7780e-04
Epoch 23/100
6700/6700 [==============================] - 0s 14us/step - loss: 2.0759e-04
- val_loss: 3.7397e-04
Epoch 24/100
6700/6700 [==============================] - 0s 14us/step - loss: 2.0104e-04
- val_loss: 2.4348e-04
Epoch 25/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.4547e-04
- val_loss: 2.6907e-04
Epoch 26/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.4899e-04
- val_loss: 4.1676e-04
Epoch 27/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.6044e-04
- val_loss: 2.8890e-04
Epoch 28/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.6294e-04
- val_loss: 3.2142e-04
Epoch 29/100
6700/6700 [==============================] - 0s 14us/step - loss: 2.1283e-04
- val_loss: 3.5773e-04
Epoch 30/100
6700/6700 [==============================] - 0s 14us/step - loss: 2.1528e-04
- val_loss: 2.8416e-04
Epoch 31/100
6700/6700 [==============================] - 0s 14us/step - loss: 1.8784e-04
- val_loss: 3.2071e-04
Epoch 32/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.7837e-04
- val_loss: 2.8808e-04
Epoch 33/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.8815e-04
- val_loss: 2.7028e-04
Epoch 34/100
6700/6700 [==============================] - 0s 17us/step - loss: 1.6249e-04
- val_loss: 2.2668e-04
Epoch 35/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.4013e-04
- val_loss: 2.1050e-04
Epoch 36/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.6698e-04
- val_loss: 2.2421e-04
Epoch 37/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.6054e-04
- val_loss: 2.7347e-04
Epoch 38/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.1196e-04
```

```
- val_loss: 2.5605e-04
Epoch 39/100
6700/6700 [==============================] - 0s 14us/step - loss: 1.0148e-04
- val_loss: 2.6875e-04
Epoch 40/100
6700/6700 [==============================] - 0s 14us/step - loss: 1.2641e-04
- val_loss: 2.4072e-04
Epoch 41/100
6700/6700 [==============================] - 0s 14us/step - loss: 1.1043e-04
- val_loss: 2.4538e-04
Epoch 42/100
6700/6700 [==============================] - 0s 14us/step - loss: 8.5789e-05
- val_loss: 2.0870e-04
Epoch 43/100
6700/6700 [==============================] - 0s 14us/step - loss: 7.9371e-05
- val_loss: 2.3521e-04
Epoch 44/100
6700/6700 [==============================] - 0s 15us/step - loss: 9.8144e-05
- val_loss: 2.4827e-04
Epoch 45/100
6700/6700 [==============================] - 0s 14us/step - loss: 6.4969e-05
- val_loss: 2.4456e-04
Epoch 46/100
6700/6700 [==============================] - 0s 15us/step - loss: 8.8783e-05
- val_loss: 2.4241e-04
Epoch 47/100
6700/6700 [==============================] - 0s 17us/step - loss: 9.5675e-05
- val_loss: 2.8629e-04
Epoch 48/100
6700/6700 [==============================] - 0s 17us/step - loss: 9.2001e-05
- val_loss: 2.5288e-04
Epoch 49/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.3692e-04
- val_loss: 2.8992e-04
Epoch 50/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.3048e-04
- val_loss: 2.3019e-04
Epoch 51/100
6700/6700 [==============================] - 0s 17us/step - loss: 1.2334e-04
- val_loss: 2.0251e-04
Epoch 52/100
6700/6700 [==============================] - 0s 17us/step - loss: 1.2470e-04
- val_loss: 2.4465e-04
Epoch 53/100
6700/6700 [==============================] - 0s 15us/step - loss: 8.6297e-05
- val_loss: 1.8108e-04
Epoch 54/100
6700/6700 [==============================] - 0s 14us/step - loss: 1.0387e-04
- val_loss: 1.7673e-04
Epoch 55/100
6700/6700 [==============================] - 0s 14us/step - loss: 1.3145e-04
- val_loss: 2.2125e-04
Epoch 56/100
6700/6700 [==============================] - 0s 14us/step - loss: 9.8194e-05
- val_loss: 2.1349e-04
Epoch 57/100
6700/6700 [==============================] - 0s 14us/step - loss: 8.7307e-05
```

```
- val_loss: 2.1779e-04
Epoch 58/100
6700/6700 [==============================] - 0s 17us/step - loss: 8.1872e-05
- val_loss: 2.3609e-04
Epoch 59/100
6700/6700 [==============================] - 0s 17us/step - loss: 6.5726e-05
- val_loss: 2.0276e-04
Epoch 60/100
6700/6700 [==============================] - 0s 14us/step - loss: 7.5456e-05
- val_loss: 2.0403e-04
Epoch 61/100
6700/6700 [==============================] - 0s 14us/step - loss: 6.1114e-05
- val_loss: 2.2222e-04
Epoch 62/100
6700/6700 [==============================] - 0s 14us/step - loss: 5.4757e-05
- val_loss: 1.6872e-04
Epoch 63/100
6700/6700 [==============================] - 0s 14us/step - loss: 5.9361e-05
- val_loss: 2.5263e-04
Epoch 64/100
6700/6700 [==============================] - ETA: 0s - loss: 4.6570e-0 - 0s 1
5us/step - loss: 5.3771e-05 - val_loss: 1.8171e-04
Epoch 65/100
6700/6700 [==============================] - 0s 17us/step - loss: 5.6788e-05
- val_loss: 3.0584e-04
Epoch 66/100
6700/6700 [==============================] - 0s 16us/step - loss: 6.5091e-05
- val_loss: 1.4948e-04
Epoch 67/100
6700/6700 [==============================] - 0s 14us/step - loss: 5.7455e-05
- val_loss: 1.6258e-04
Epoch 68/100
6700/6700 [==============================] - 0s 14us/step - loss: 6.9598e-05
- val_loss: 2.0332e-04
Epoch 69/100
6700/6700 [==============================] - 0s 15us/step - loss: 6.8590e-05
- val_loss: 2.0118e-04
Epoch 70/100
6700/6700 [==============================] - 0s 18us/step - loss: 7.0734e-05
- val_loss: 2.3669e-04
Epoch 71/100
6700/6700 [==============================] - 0s 16us/step - loss: 1.1239e-04
- val_loss: 1.6155e-04
Epoch 72/100
6700/6700 [==============================] - 0s 15us/step - loss: 6.8656e-05
- val_loss: 1.6386e-04
Epoch 73/100
6700/6700 [==============================] - 0s 14us/step - loss: 6.5449e-05
- val_loss: 1.6285e-04
Epoch 74/100
6700/6700 [==============================] - 0s 15us/step - loss: 1.2916e-04
- val_loss: 1.5346e-04
Epoch 75/100
6700/6700 [==============================] - 0s 14us/step - loss: 1.4664e-04
- val_loss: 2.6980e-04
Epoch 76/100
6700/6700 [==============================] - 0s 14us/step - loss: 1.3508e-04
```

```
- val_loss: 1.6402e-04
Epoch 77/100
6700/6700 [==============================] - 0s 14us/step - loss: 6.2690e-05
- val_loss: 1.5513e-04
Epoch 78/100
6700/6700 [==============================] - 0s 14us/step - loss: 4.4216e-05
- val_loss: 1.3988e-04
Epoch 79/100
6700/6700 [==============================] - 0s 14us/step - loss: 4.4806e-05
- val_loss: 1.4517e-04
Epoch 80/100
6700/6700 [==============================] - 0s 14us/step - loss: 4.2969e-05
- val_loss: 1.3328e-04
Epoch 81/100
6700/6700 [==============================] - 0s 15us/step - loss: 4.0551e-05
- val_loss: 2.0161e-04
Epoch 82/100
6700/6700 [==============================] - 0s 14us/step - loss: 4.0087e-05
- val_loss: 1.3500e-04
Epoch 83/100
6700/6700 [==============================] - 0s 14us/step - loss: 3.7657e-05
- val_loss: 1.2472e-04
Epoch 84/100
6700/6700 [==============================] - 0s 14us/step - loss: 2.9260e-05
- val_loss: 1.4157e-04
Epoch 85/100
6700/6700 [==============================] - 0s 14us/step - loss: 2.8124e-05
- val_loss: 1.2345e-04
Epoch 86/100
6700/6700 [==============================] - 0s 14us/step - loss: 3.0886e-05
- val_loss: 1.2925e-04
Epoch 87/100
6700/6700 [==============================] - 0s 14us/step - loss: 3.7691e-05
- val_loss: 1.4433e-04
Epoch 88/100
6700/6700 [==============================] - 0s 14us/step - loss: 4.6137e-05
- val_loss: 1.4279e-04
Epoch 89/100
6700/6700 [==============================] - 0s 14us/step - loss: 4.6925e-05
- val_loss: 1.2765e-04
Epoch 90/100
6700/6700 [==============================] - 0s 14us/step - loss: 3.1655e-05
- val_loss: 1.3989e-04
Epoch 91/100
6700/6700 [==============================] - 0s 14us/step - loss: 3.5668e-05
- val_loss: 1.1718e-04
Epoch 92/100
6700/6700 [==============================] - 0s 15us/step - loss: 5.4470e-05
- val_loss: 1.4867e-04
Epoch 93/100
6700/6700 [==============================] - 0s 14us/step - loss: 5.4770e-05
- val_loss: 1.3947e-04
Epoch 94/100
6700/6700 [==============================] - 0s 14us/step - loss: 7.8101e-05
- val_loss: 1.2246e-04
Epoch 95/100
6700/6700 [==============================] - 0s 14us/step - loss: 7.8687e-05
```

```
- val_loss: 1.4130e-04
Epoch 96/100
6700/6700 [==============================] - 0s 14us/step - loss: 1.0096e-04
- val_loss: 1.1429e-04
Epoch 97/100
6700/6700 [==============================] - 0s 15us/step - loss: 5.1945e-05
- val_loss: 1.2295e-04
Epoch 98/100
6700/6700 [==============================] - 0s 14us/step - loss: 4.3305e-05
- val_loss: 1.1678e-04
Epoch 99/100
6700/6700 [==============================] - 0s 14us/step - loss: 2.7169e-05
- val_loss: 1.1534e-04
Epoch 100/100
6700/6700 [==============================] - 0s 14us/step - loss: 2.1146e-05
- val_loss: 1.1049e-04
```

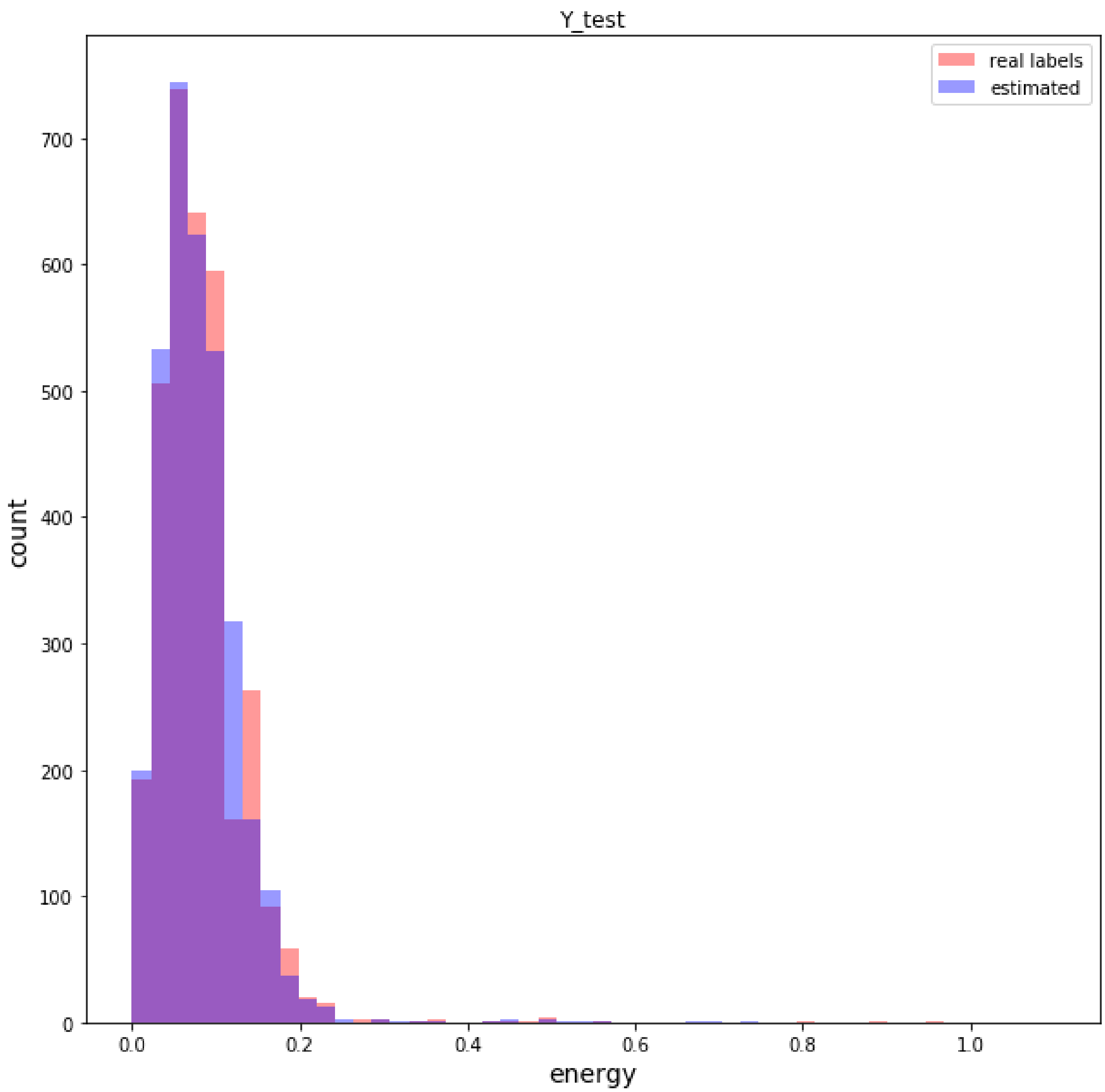
In [38]:

```python
predsE = model2.predict(X_testE)
print('MSE : ', mean_squared_error(predsE, Y_testE))
```

```
MSE :  0.00011048531808883348
```

In [39]:
```
plt.figure(figsize=(10,10))
Y_hist = plt.hist(Y_testE, 50, (0,1.1),histtype="stepfilled",alpha= 0.4, color
='r')
plt.title("Y_test")
pred_hist = plt.hist(predsE, 50, (0,1.1),histtype="stepfilled",alpha= 0.4, col
or ='b')
plt.legend(["real labels", "estimated"], loc='upper right')
plt.xlabel("energy", fontsize = 14)
plt.ylabel("count",  fontsize = 14)
```

Out[39]:
```
Text(0, 0.5, 'count')
```



4. Neural Network learns to calculate coupling strength

In [40]:
```
traindata2 = pd.read_csv("training_data2.csv", names=["K"]+["E"]+["M"]+[i for
i in range(L*L)])
```

In [41]:
```
traindata2.head()
```

Out[41]:

	K	E	M	0	1	2	3	4	5	6	...	90	91	92	93	94	95	96	97	98
0	0.981545	-866.913386	0.98	1	1	1	1	1	1	1	...	1	1	1	1	1	1	1	1	1
1	0.244548	-472.604335	0.82	1	1	1	1	1	1	1	...	1	1	1	1	1	1	1	1	1
2	1.479373	-1058.079211	0.98	1	1	1	1	1	1	1	...	1	1	1	1	1	1	1	1	1
3	3.178758	-1649.783027	0.96	1	1	1	-1	1	1	1	...	1	1	1	1	1	1	1	1	1
4	0.148842	-438.103564	0.80	1	1	1	1	1	1	1	...	1	1	1	1	1	1	-1	1	1

5 rows × 103 columns

In [42]:
```
traindata2["E"]=(traindata2["E"]-min(traindata2["E"]))/(max(traindata2["E"])-m
in(traindata2["E"]))
```

In [43]:
```
traindata2["M"]=(traindata2["M"]-min(traindata2["M"]))/(max(traindata2["M"])-m
in(traindata2["M"]))
```

In [44]:
```
trainlabelsK=traindata2["K"].values
```

In [45]:
```
train_attrs2=traindata2.drop(columns=["K"])
```

In [46]:
```
trainlabelsK=(trainlabelsK-min(trainlabelsK))/(max(trainlabelsK)-min(trainlabe
lsK))
```

In [47]:
```
min(trainlabelsK)
```

Out[47]: 0.0

In [48]:
```
X_trainK, X_testK, Y_trainK, Y_testK = train_test_split(train_attrs2, trainlab
elsK, test_size=0.33, random_state=0)
```

In [49]:
```
X_trainK.shape, Y_trainK.shape, X_testK.shape, Y_testK.shape
```

Out[49]: ((3350, 102), (3350,), (1650, 102), (1650,))

In [50]:
```
model3 = Sequential()
model3.add(Dense(102, input_dim=102, activation='relu'))
model3.add(Dense(100, activation='relu'))
model3.add(Dense(100, activation='relu'))
model3.add(Dense(40, activation='relu'))
model3.add(Dense(1, activation='relu'))
```

In [51]:
```
model3.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_7 (Dense)              (None, 102)               10506     
_________________________________________________________________
dense_8 (Dense)              (None, 100)               10300     
_________________________________________________________________
dense_9 (Dense)              (None, 100)               10100     
_________________________________________________________________
dense_10 (Dense)             (None, 40)                4040      
_________________________________________________________________
dense_11 (Dense)             (None, 1)                 41        
=================================================================
Total params: 34,987
Trainable params: 34,987
Non-trainable params: 0
_________________________________________________________________
```

In [52]:
```
model3.compile(loss='mean_squared_error', optimizer='adam')
```

In [53]:
```
history = model3.fit(x=X_trainK, y=Y_trainK, batch_size=64, epochs=100, valida
tion_data=(X_testK, Y_testK))
```

```
Train on 3350 samples, validate on 1650 samples
Epoch 1/100
3350/3350 [==============================] - 0s 102us/step - loss: 0.0122 - v
al_loss: 0.0084
Epoch 2/100
3350/3350 [==============================] - 0s 21us/step - loss: 0.0058 - va
l_loss: 0.0054
Epoch 3/100
3350/3350 [==============================] - 0s 21us/step - loss: 0.0046 - va
l_loss: 0.0052
Epoch 4/100
3350/3350 [==============================] - 0s 20us/step - loss: 0.0039 - va
l_loss: 0.0048
Epoch 5/100
3350/3350 [==============================] - 0s 20us/step - loss: 0.0036 - va
l_loss: 0.0056
Epoch 6/100
3350/3350 [==============================] - 0s 20us/step - loss: 0.0035 - va
l_loss: 0.0040
Epoch 7/100
3350/3350 [==============================] - 0s 19us/step - loss: 0.0032 - va
l_loss: 0.0040
Epoch 8/100
3350/3350 [==============================] - 0s 21us/step - loss: 0.0027 - va
l_loss: 0.0036
Epoch 9/100
3350/3350 [==============================] - 0s 19us/step - loss: 0.0022 - va
l_loss: 0.0031
Epoch 10/100
3350/3350 [==============================] - 0s 20us/step - loss: 0.0020 - va
l_loss: 0.0038
Epoch 11/100
3350/3350 [==============================] - 0s 21us/step - loss: 0.0018 - va
l_loss: 0.0027
Epoch 12/100
3350/3350 [==============================] - 0s 21us/step - loss: 0.0011 - va
l_loss: 0.0027
Epoch 13/100
3350/3350 [==============================] - 0s 20us/step - loss: 0.0012 - va
l_loss: 0.0027
Epoch 14/100
3350/3350 [==============================] - 0s 21us/step - loss: 8.7859e-04
- val_loss: 0.0025
Epoch 15/100
3350/3350 [==============================] - 0s 20us/step - loss: 7.5626e-04
- val_loss: 0.0024
Epoch 16/100
3350/3350 [==============================] - 0s 20us/step - loss: 7.2821e-04
- val_loss: 0.0024
Epoch 17/100
3350/3350 [==============================] - 0s 20us/step - loss: 5.5854e-04
- val_loss: 0.0022
Epoch 18/100
3350/3350 [==============================] - 0s 20us/step - loss: 5.8436e-04
- val_loss: 0.0024
Epoch 19/100
3350/3350 [==============================] - 0s 20us/step - loss: 5.9514e-04
```

```
- val_loss: 0.0021
Epoch 20/100
3350/3350 [==============================] - 0s 20us/step - loss: 5.7663e-04
- val_loss: 0.0025
Epoch 21/100
3350/3350 [==============================] - 0s 20us/step - loss: 4.6344e-04
- val_loss: 0.0023
Epoch 22/100
3350/3350 [==============================] - 0s 20us/step - loss: 4.9757e-04
- val_loss: 0.0033
Epoch 23/100
3350/3350 [==============================] - 0s 20us/step - loss: 5.0893e-04
- val_loss: 0.0021
Epoch 24/100
3350/3350 [==============================] - 0s 20us/step - loss: 3.1786e-04
- val_loss: 0.0020
Epoch 25/100
3350/3350 [==============================] - 0s 20us/step - loss: 3.8621e-04
- val_loss: 0.0020
Epoch 26/100
3350/3350 [==============================] - 0s 20us/step - loss: 3.1208e-04
- val_loss: 0.0029
Epoch 27/100
3350/3350 [==============================] - 0s 20us/step - loss: 4.6536e-04
- val_loss: 0.0023
Epoch 28/100
3350/3350 [==============================] - 0s 20us/step - loss: 5.7816e-04
- val_loss: 0.0020
Epoch 29/100
3350/3350 [==============================] - 0s 20us/step - loss: 2.7729e-04
- val_loss: 0.0019
Epoch 30/100
3350/3350 [==============================] - 0s 20us/step - loss: 1.8378e-04
- val_loss: 0.0019
Epoch 31/100
3350/3350 [==============================] - 0s 20us/step - loss: 1.7033e-04
- val_loss: 0.0018
Epoch 32/100
3350/3350 [==============================] - 0s 20us/step - loss: 2.0217e-04
- val_loss: 0.0021
Epoch 33/100
3350/3350 [==============================] - 0s 20us/step - loss: 1.7423e-04
- val_loss: 0.0019
Epoch 34/100
3350/3350 [==============================] - 0s 20us/step - loss: 1.9935e-04
- val_loss: 0.0019
Epoch 35/100
3350/3350 [==============================] - 0s 20us/step - loss: 1.8724e-04
- val_loss: 0.0023
Epoch 36/100
3350/3350 [==============================] - 0s 20us/step - loss: 2.1344e-04
- val_loss: 0.0019
Epoch 37/100
3350/3350 [==============================] - 0s 21us/step - loss: 1.4211e-04
- val_loss: 0.0019
Epoch 38/100
3350/3350 [==============================] - 0s 20us/step - loss: 2.3550e-04
```

```
- val_loss: 0.0018
Epoch 39/100
3350/3350 [==============================] - 0s 20us/step - loss: 3.8074e-04
- val_loss: 0.0018
Epoch 40/100
3350/3350 [==============================] - 0s 21us/step - loss: 4.3072e-04
- val_loss: 0.0019
Epoch 41/100
3350/3350 [==============================] - 0s 20us/step - loss: 2.6892e-04
- val_loss: 0.0023
Epoch 42/100
3350/3350 [==============================] - 0s 20us/step - loss: 3.5764e-04
- val_loss: 0.0018
Epoch 43/100
3350/3350 [==============================] - 0s 20us/step - loss: 1.5433e-04
- val_loss: 0.0018
Epoch 44/100
3350/3350 [==============================] - 0s 20us/step - loss: 1.2722e-04
- val_loss: 0.0017
Epoch 45/100
3350/3350 [==============================] - 0s 20us/step - loss: 1.2453e-04
- val_loss: 0.0017
Epoch 46/100
3350/3350 [==============================] - 0s 20us/step - loss: 8.1918e-05
- val_loss: 0.0020
Epoch 47/100
3350/3350 [==============================] - 0s 20us/step - loss: 8.7455e-05
- val_loss: 0.0017
Epoch 48/100
3350/3350 [==============================] - 0s 20us/step - loss: 8.7114e-05
- val_loss: 0.0017
Epoch 49/100
3350/3350 [==============================] - 0s 20us/step - loss: 1.3941e-04
- val_loss: 0.0017
Epoch 50/100
3350/3350 [==============================] - 0s 20us/step - loss: 2.1987e-04
- val_loss: 0.0018
Epoch 51/100
3350/3350 [==============================] - 0s 20us/step - loss: 1.1079e-04
- val_loss: 0.0018
Epoch 52/100
3350/3350 [==============================] - 0s 19us/step - loss: 4.2140e-04
- val_loss: 0.0019
Epoch 53/100
3350/3350 [==============================] - 0s 20us/step - loss: 2.6078e-04
- val_loss: 0.0018
Epoch 54/100
3350/3350 [==============================] - 0s 19us/step - loss: 1.8966e-04
- val_loss: 0.0018
Epoch 55/100
3350/3350 [==============================] - 0s 18us/step - loss: 1.1278e-04
- val_loss: 0.0017
Epoch 56/100
3350/3350 [==============================] - 0s 18us/step - loss: 8.1299e-05
- val_loss: 0.0017
Epoch 57/100
3350/3350 [==============================] - 0s 19us/step - loss: 2.0992e-04
```

```
- val_loss: 0.0016
Epoch 58/100
3350/3350 [==============================] - 0s 19us/step - loss: 1.4064e-04
- val_loss: 0.0019
Epoch 59/100
3350/3350 [==============================] - 0s 20us/step - loss: 2.2128e-04
- val_loss: 0.0017
Epoch 60/100
3350/3350 [==============================] - 0s 18us/step - loss: 1.3053e-04
- val_loss: 0.0016
Epoch 61/100
3350/3350 [==============================] - 0s 19us/step - loss: 3.5315e-04
- val_loss: 0.0017
Epoch 62/100
3350/3350 [==============================] - 0s 18us/step - loss: 2.2724e-04
- val_loss: 0.0018
Epoch 63/100
3350/3350 [==============================] - 0s 19us/step - loss: 2.6776e-04
- val_loss: 0.0017
Epoch 64/100
3350/3350 [==============================] - 0s 19us/step - loss: 1.5492e-04
- val_loss: 0.0018
Epoch 65/100
3350/3350 [==============================] - 0s 18us/step - loss: 2.4177e-04
- val_loss: 0.0025
Epoch 66/100
3350/3350 [==============================] - 0s 19us/step - loss: 2.6040e-04
- val_loss: 0.0017
Epoch 67/100
3350/3350 [==============================] - 0s 18us/step - loss: 1.3431e-04
- val_loss: 0.0016
Epoch 68/100
3350/3350 [==============================] - 0s 19us/step - loss: 8.9730e-05
- val_loss: 0.0016
Epoch 69/100
3350/3350 [==============================] - 0s 19us/step - loss: 7.0689e-05
- val_loss: 0.0016
Epoch 70/100
3350/3350 [==============================] - 0s 19us/step - loss: 7.1202e-05
- val_loss: 0.0016
Epoch 71/100
3350/3350 [==============================] - 0s 18us/step - loss: 8.7717e-05
- val_loss: 0.0015
Epoch 72/100
3350/3350 [==============================] - 0s 19us/step - loss: 7.6731e-05
- val_loss: 0.0016
Epoch 73/100
3350/3350 [==============================] - 0s 18us/step - loss: 1.0921e-04
- val_loss: 0.0015
Epoch 74/100
3350/3350 [==============================] - 0s 19us/step - loss: 6.8771e-05
- val_loss: 0.0015
Epoch 75/100
3350/3350 [==============================] - 0s 19us/step - loss: 1.3634e-04
- val_loss: 0.0018
Epoch 76/100
3350/3350 [==============================] - 0s 18us/step - loss: 1.4562e-04
```

```
- val_loss: 0.0016
Epoch 77/100
3350/3350 [==============================] - 0s 19us/step - loss: 8.4579e-05
- val_loss: 0.0016
Epoch 78/100
3350/3350 [==============================] - 0s 19us/step - loss: 8.4325e-05
- val_loss: 0.0015
Epoch 79/100
3350/3350 [==============================] - 0s 19us/step - loss: 9.9866e-05
- val_loss: 0.0015
Epoch 80/100
3350/3350 [==============================] - 0s 18us/step - loss: 3.6033e-04
- val_loss: 0.0016
Epoch 81/100
3350/3350 [==============================] - 0s 18us/step - loss: 2.0272e-04
- val_loss: 0.0019
Epoch 82/100
3350/3350 [==============================] - 0s 18us/step - loss: 2.7051e-04
- val_loss: 0.0015
Epoch 83/100
3350/3350 [==============================] - 0s 19us/step - loss: 2.2900e-04
- val_loss: 0.0015
Epoch 84/100
3350/3350 [==============================] - 0s 18us/step - loss: 9.1139e-05
- val_loss: 0.0014
Epoch 85/100
3350/3350 [==============================] - 0s 19us/step - loss: 6.7970e-05
- val_loss: 0.0014
Epoch 86/100
3350/3350 [==============================] - 0s 20us/step - loss: 1.2413e-04
- val_loss: 0.0015
Epoch 87/100
3350/3350 [==============================] - 0s 19us/step - loss: 1.9477e-04
- val_loss: 0.0014
Epoch 88/100
3350/3350 [==============================] - 0s 20us/step - loss: 7.0105e-05
- val_loss: 0.0014
Epoch 89/100
3350/3350 [==============================] - 0s 20us/step - loss: 8.1347e-05
- val_loss: 0.0014
Epoch 90/100
3350/3350 [==============================] - 0s 19us/step - loss: 7.0587e-05
- val_loss: 0.0014
Epoch 91/100
3350/3350 [==============================] - 0s 19us/step - loss: 6.6642e-05
- val_loss: 0.0014
Epoch 92/100
3350/3350 [==============================] - 0s 18us/step - loss: 7.6651e-05
- val_loss: 0.0015
Epoch 93/100
3350/3350 [==============================] - 0s 19us/step - loss: 1.9916e-04
- val_loss: 0.0014
Epoch 94/100
3350/3350 [==============================] - 0s 18us/step - loss: 7.7098e-05
- val_loss: 0.0014
Epoch 95/100
3350/3350 [==============================] - 0s 18us/step - loss: 8.0648e-05
```

```
- val_loss: 0.0014
Epoch 96/100
3350/3350 [==============================] - 0s 20us/step - loss: 8.8086e-05
- val_loss: 0.0015
Epoch 97/100
3350/3350 [==============================] - 0s 18us/step - loss: 1.4877e-04
- val_loss: 0.0019
Epoch 98/100
3350/3350 [==============================] - 0s 20us/step - loss: 1.8819e-04
- val_loss: 0.0015
Epoch 99/100
3350/3350 [==============================] - 0s 19us/step - loss: 1.7048e-04
- val_loss: 0.0015
Epoch 100/100
3350/3350 [==============================] - 0s 21us/step - loss: 1.1905e-04
- val_loss: 0.0014
```

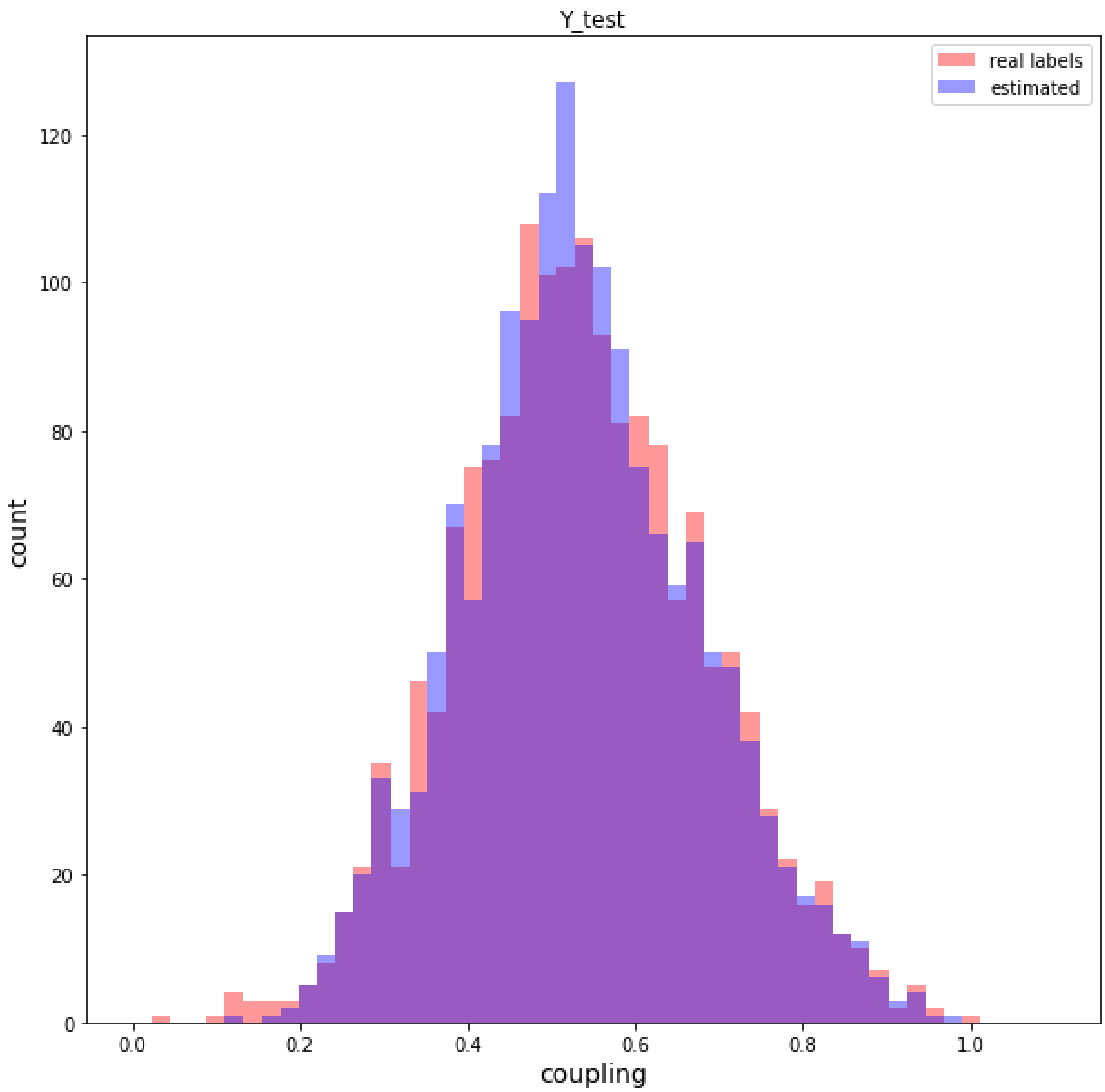
In [54]:

```python
predsK = model3.predict(X_testK)
print('MSE : ', mean_squared_error(predsK, Y_testK))
```

```
MSE :  0.001403084344643398
```

In [55]:
```
plt.figure(figsize=(10,10))
Y_hist = plt.hist(Y_testK, 50, (0,1.1),histtype="stepfilled",alpha= 0.4, color
='r')
plt.title("Y_test")
pred_hist = plt.hist(predsK, 50, (0,1.1),histtype="stepfilled",alpha= 0.4, col
or ='b')
plt.legend(["real labels", "estimated"], loc='upper right')
plt.xlabel("coupling", fontsize = 14)
plt.ylabel("count",  fontsize = 14)
```

Out[55]:
```
Text(0, 0.5, 'count')
```



In [ ]: