CENTRO UNIVERSITÁRIO FEI

Avenida Humberto de Alencar Castelo Branco, 3972, CEP: 09850-901 São Bernardo do Campo
Telefone: (011) 4353-2900 Fax (011) 4109-5994
Curso de Ciências da Computação

Inteligência Artificial & Robótica

Atividade #05:

RNA Aproximador de funções

Autores: José Joaquim da Silva Neto 22.221.013-0 Lucas Guglielmi Pereira 22.121.013-1 Gabriel Martinho S de Moraes 22.120.039-7 William Yang 22.121.043-8	Data de emissão: 07/04/2024

centro universitário FEI

CENTRO UNIVERSITÁRIO FEI

Avenida Humberto de Alencar Castelo Branco, 3972, CEP: 09850-901 São Bernardo do Campo
Telefone: (011) 4353-2900 Fax (011) 4109-5994

Curso de Ciências da Computação

1 Atividade Laboratório #5

- Baseado no arquivo de exemplo, monte os aproximadores de função para os testes de 2 a 5 (disponíveis na atividade!)
- Faça pelo menos 3 simulações de arquitetura (Camadas e Qtdd de neurônios) para cada problema
    - Cada simulação deve ser executada pelo menos 10x (apresente a média e o Desvio padrão do erro final)
- Incluir no relatório:
    - Link do Git com o Código-fonte
    - Gráficos com o melhor resultado das 3 simulações

1.1 RNA.py

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.neural_network import MLPRegressor

best_losses = []
for _ in range(10):
    print('Carregando Arquivo de teste')
    arquivo = np.load('teste5.npy')
    x = arquivo[0]
    y = np.ravel(arquivo[1]) #trocar arquivo aqui

    regr = MLPRegressor(hidden_layer_sizes=(2,2),
                        max_iter=1000,
                         activation='relu', #{'identity','logistic','tanh','relu'},
                        solver='adam',
                        learning_rate = 'adaptive',
                        n_iter_no_change=500)
    print('Treinando RNA')
    regr = regr.fit(x,y)
    print('Preditor')
    y_est = regr.predict(x)
    plt.figure(figsize=[14,7])

    #plot curso original
    plt.subplot(1,3,1)
    plt.plot(x,y)

    #plot aprendizagem
    plt.subplot(1,3,2)
    plt.plot(regr.loss_curve_)

    #plot regressor
    plt.subplot(1,3,3)
    plt.plot(x,y,linewidth=1,color='yellow')
    plt.plot(x,y_est,linewidth=2)
    plt.show()

    best_losses.append(regr.best_loss_)
    print("Best Loss:", regr.best_loss_)

print(" ")
# Calculatar media e desvio padrao
mean_best_loss = np.mean(best_losses)
std_best_loss = np.std(best_losses)
```

```
print("Media do Best Losses:", mean_best_loss)
print("Desvio padrao do Best Losses:", std_best_loss)
```

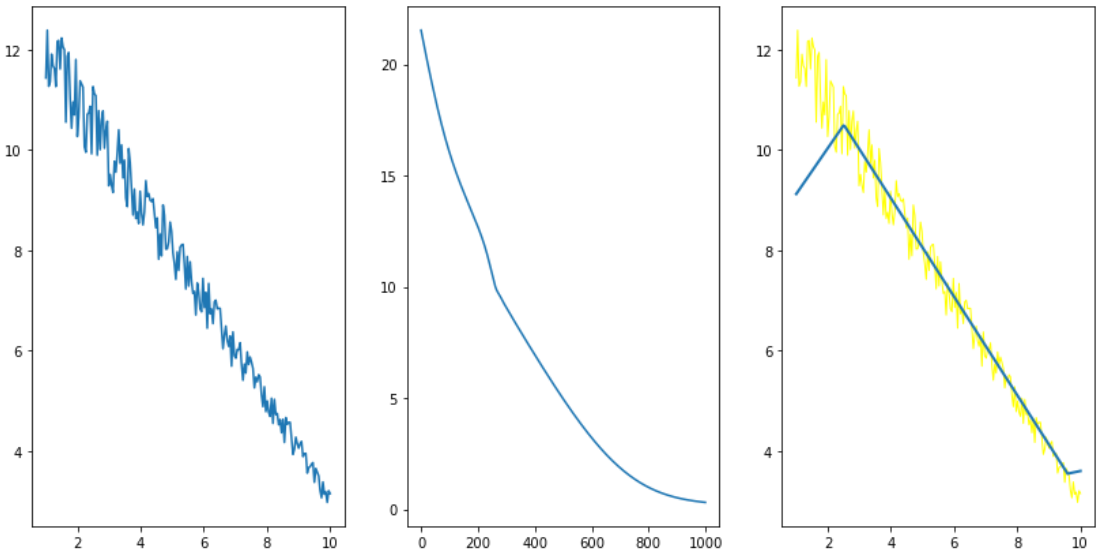
OBS: MAX_iter e n_iter_no_change não foram alterado entre os testes

Link do github: https://github.com/PurgamentumSolis/INTELIGENCIA-ARTIFICIAL-E-ROBOTICA/tree/main/Atividade%205-%20RNA%20aproximador%20de%20%20fun%C3%A7%C3%B5es

2 Melhores simulações para teste 2

2.1 Simulação 2(a)

Hidden_layer_sizes = (2,2)

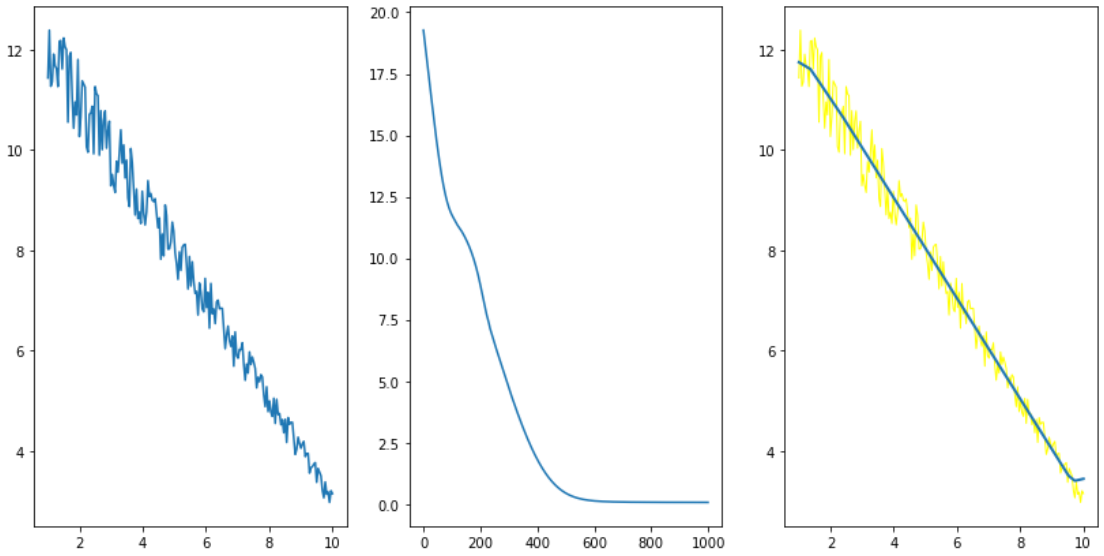



```
Best Loss: 0.3070744816876135

Media do Best Losses: 10.324397717690314
Desvio padrao do Best Losses: 9.870256907112761
```

2.2 Simulação 2(b)

Hidden_layer_sizes = (10,10)

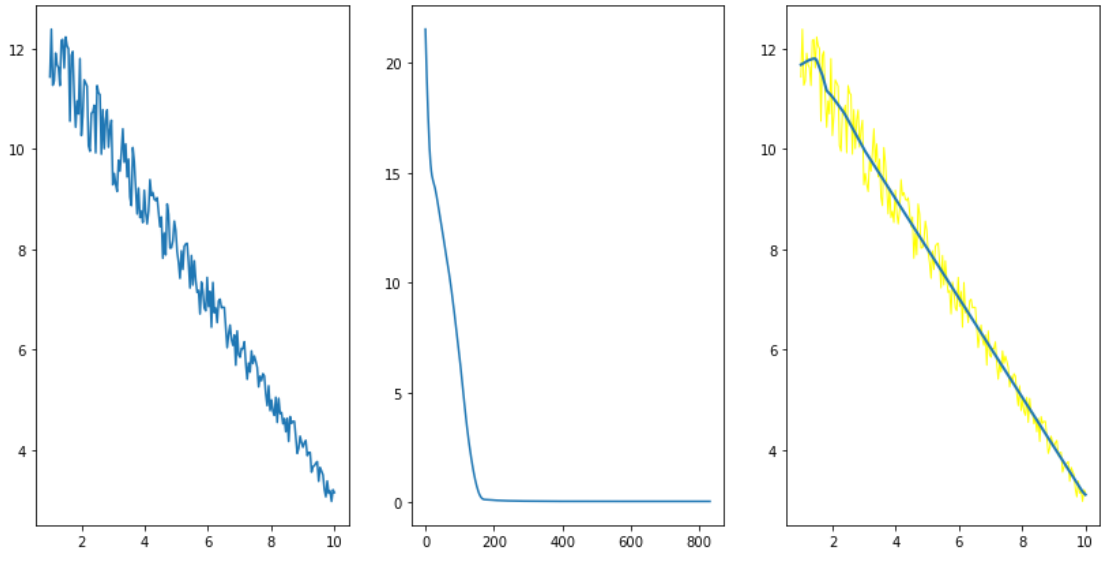


```
Best Loss: 0.06218488610241304
```

```
Media do Best Losses: 0.27988501137680866
Desvio padrao do Best Losses: 0.4758919962706571
```

2.3 Simulação 2(c)

Hidden_layer_sizes = (50,50)



0)

```
Best Loss: 0.0582530405677915
```

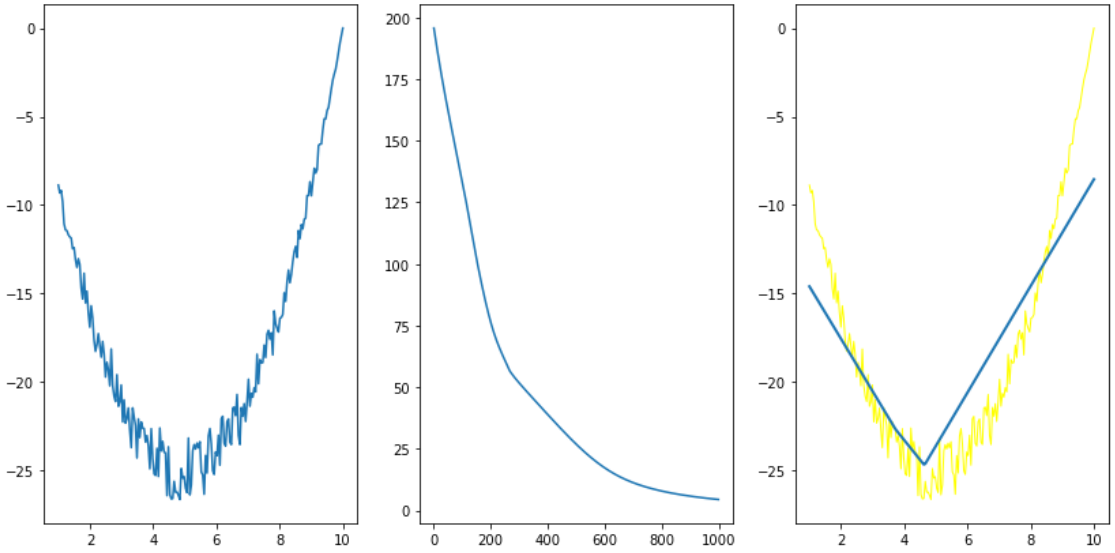
```
Media do Best Losses: 0.05953356967051056
```

```
Desvio padrao do Best Losses: 0.0009799817960286252
```

3 Melhores simulações para teste 3

3.1 Simulação 3(a)

Hidden_layer_sizes = (10,10)

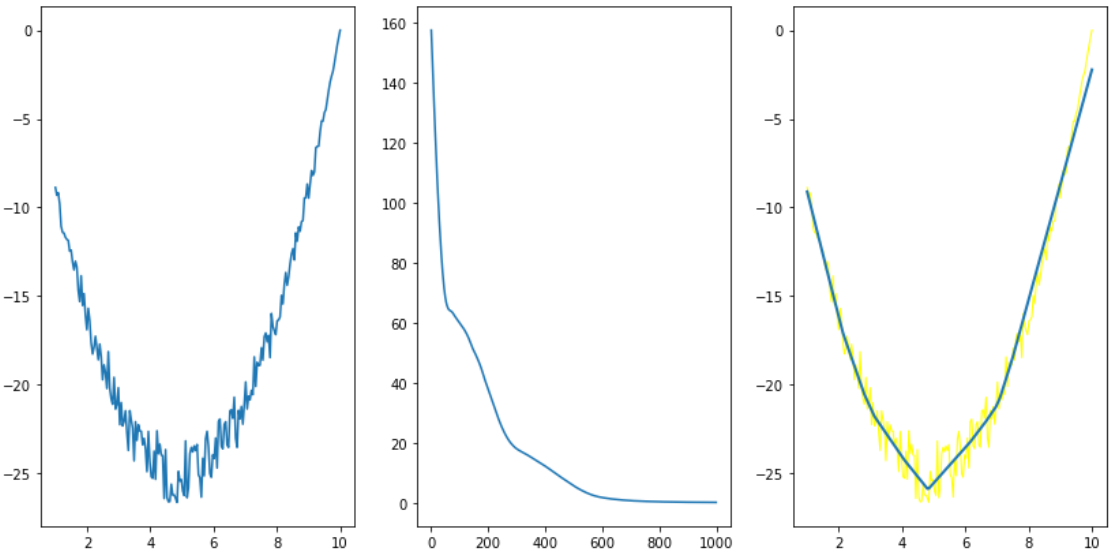


```
Best Loss: 4.458978624467434

Media do Best Losses: 20.330695822639907
Desvio padrao do Best Losses: 9.79821293189992
```

3.2 Simulação 3(b)

Hidden_layer_sizes = (50,50)

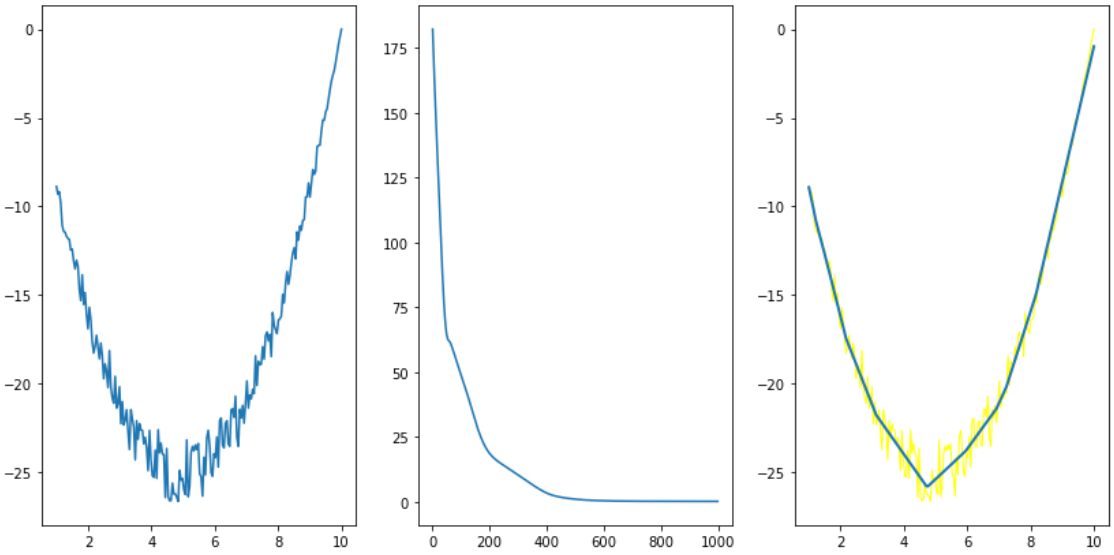


```
Best Loss: 0.40194229562303896

Media do Best Losses: 0.6935828377369089
Desvio padrao do Best Losses: 0.30941703603335324
```

3.3 Simulação 3(c)

Hidden_layer_sizes = (100,100)



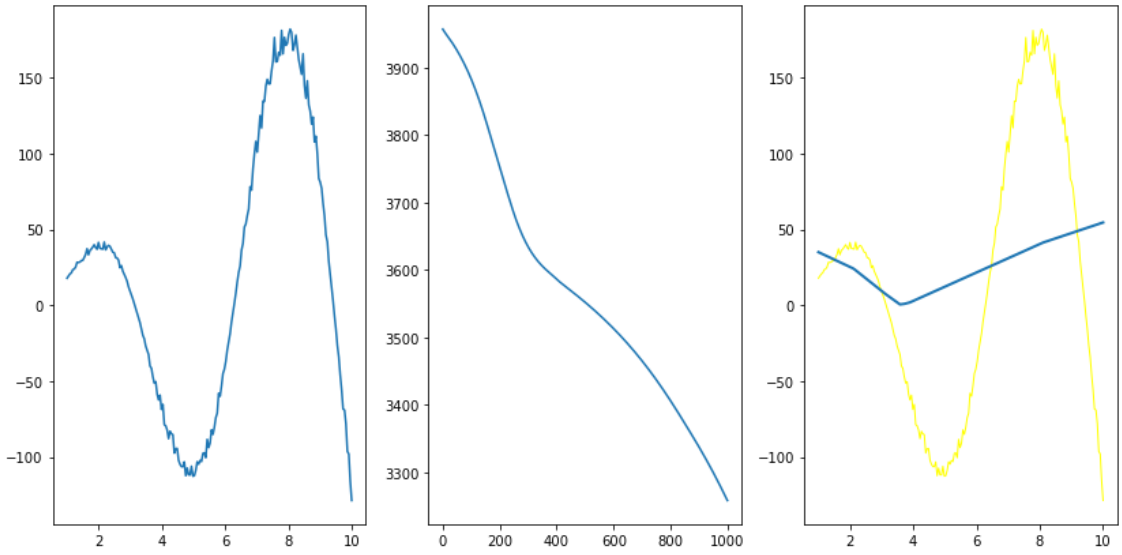
```
Best Loss: 0.2925124993099258

Media do Best Losses: 0.33518330693347603
Desvio padrao do Best Losses: 0.03857518924070262
```

4 Melhores simulações para teste 4

4.1 Simulação 4(a)

Hidden_layer_sizes = (10,10)

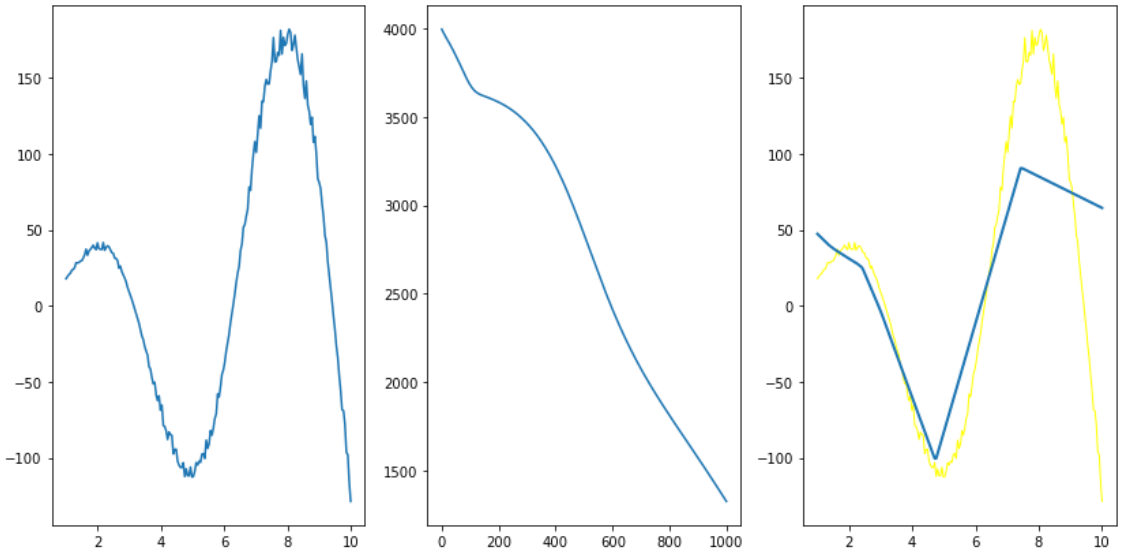


```
Best Loss: 3257.9681816984826
```

```
Media do Best Losses: 3462.9244183558744
Desvio padrao do Best Losses: 103.03056693884626
```

4.2 Simulação 4(b)

Hidden_layer_sizes = (50,50)

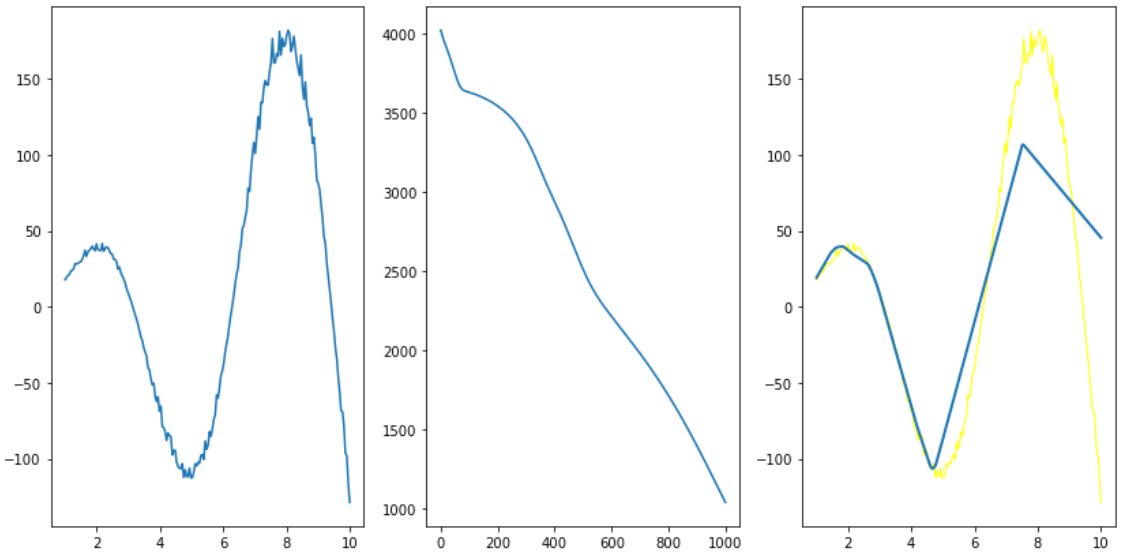


```
Best Loss: 1323.8382989213503

Media do Best Losses: 1946.7460647209386
Desvio padrao do Best Losses: 367.8487663553504
```

4.3 Simulação 4(c)

Hidden_layer_sizes = (100,100)



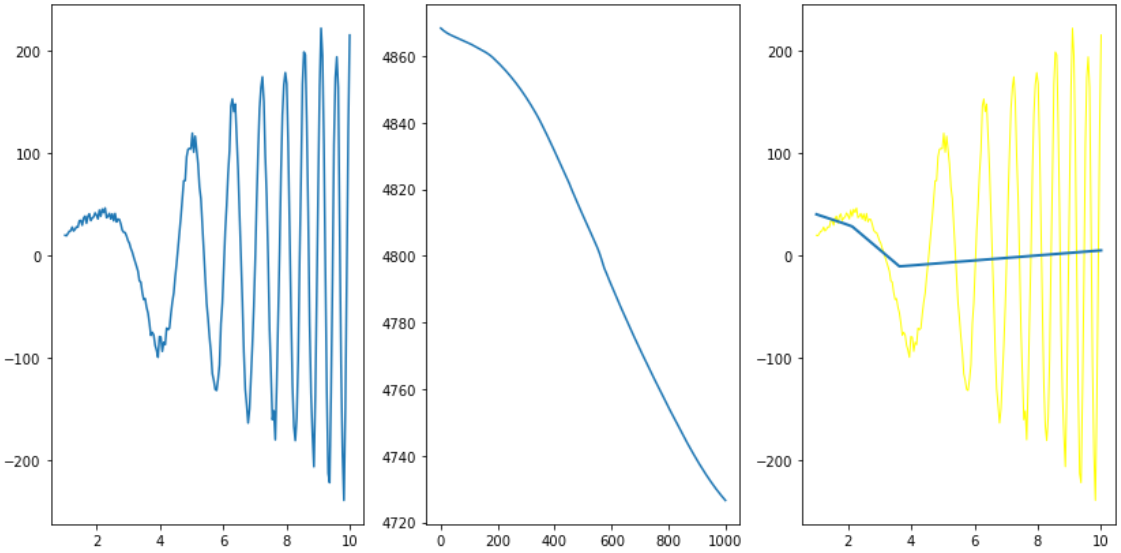
```
Best Loss: 1035.8400602451422

Media do Best Losses: 1546.7555662398645
Desvio padrao do Best Losses: 367.3711911661163
```

5 Melhores simulações para teste 5

5.1 Simulação 5(a)

Hidden_layer_sizes = (10,10)

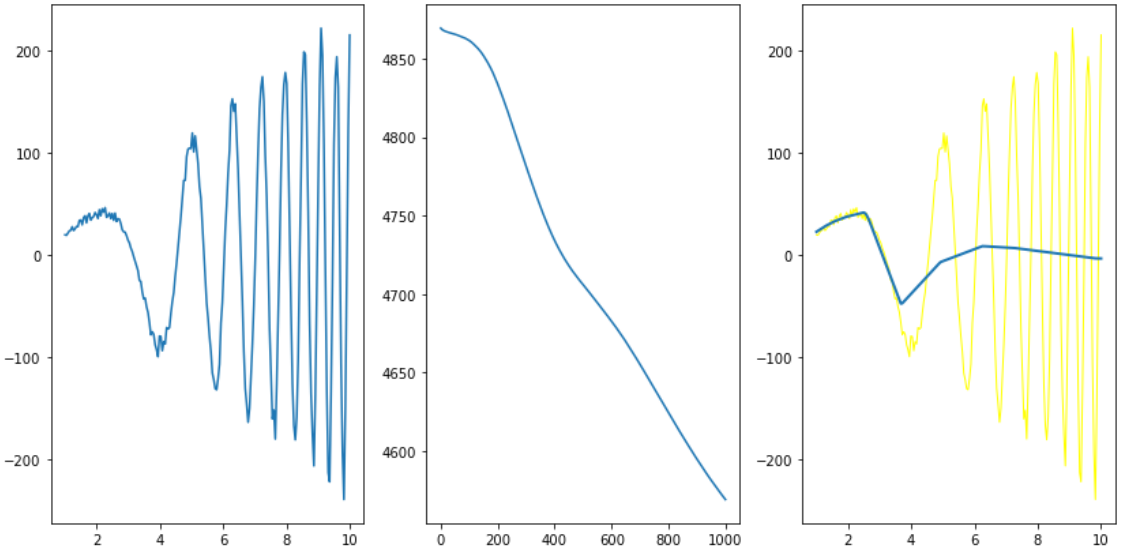


```
Best Loss: 4726.586545979326

Media do Best Losses: 4764.018127918791
Desvio padrao do Best Losses: 24.89851497406556
```

5.2 Simulação 5(b)

Hidden_layer_sizes = (50,50)



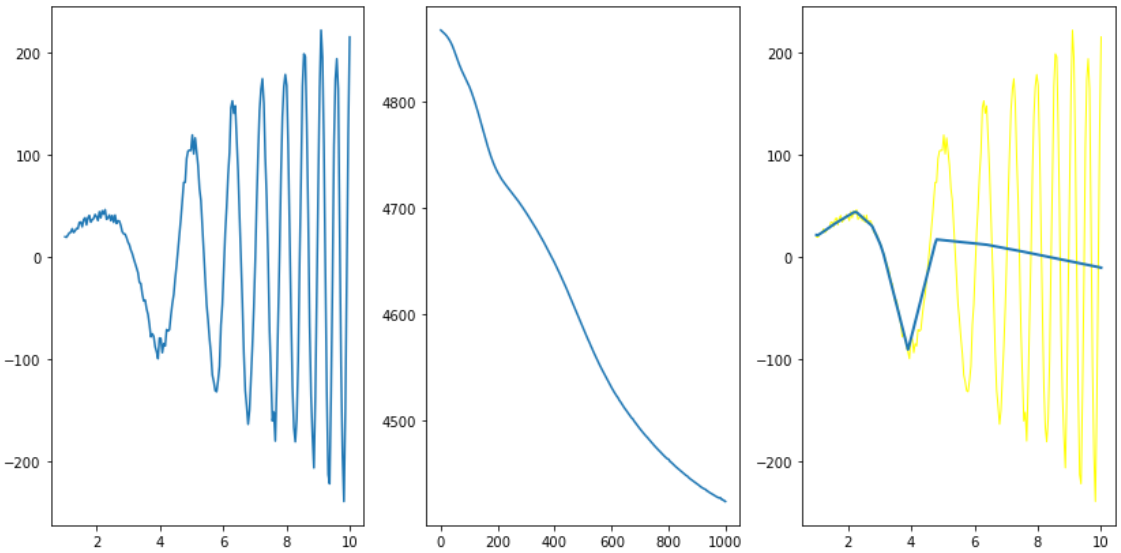
```
Best Loss: 4569.1738832937945

Media do Best Losses: 4638.445403402941
```

```
Desvio padrao do Best Losses: 34.625850957558285
```

5.3 Simulação 5(c)

Hidden_layer_sizes = (200,200)




```
Best Loss: 4423.160778028075

Media do Best Losses: 4499.100137345268
Desvio padrao do Best Losses: 56.58900086127444
```