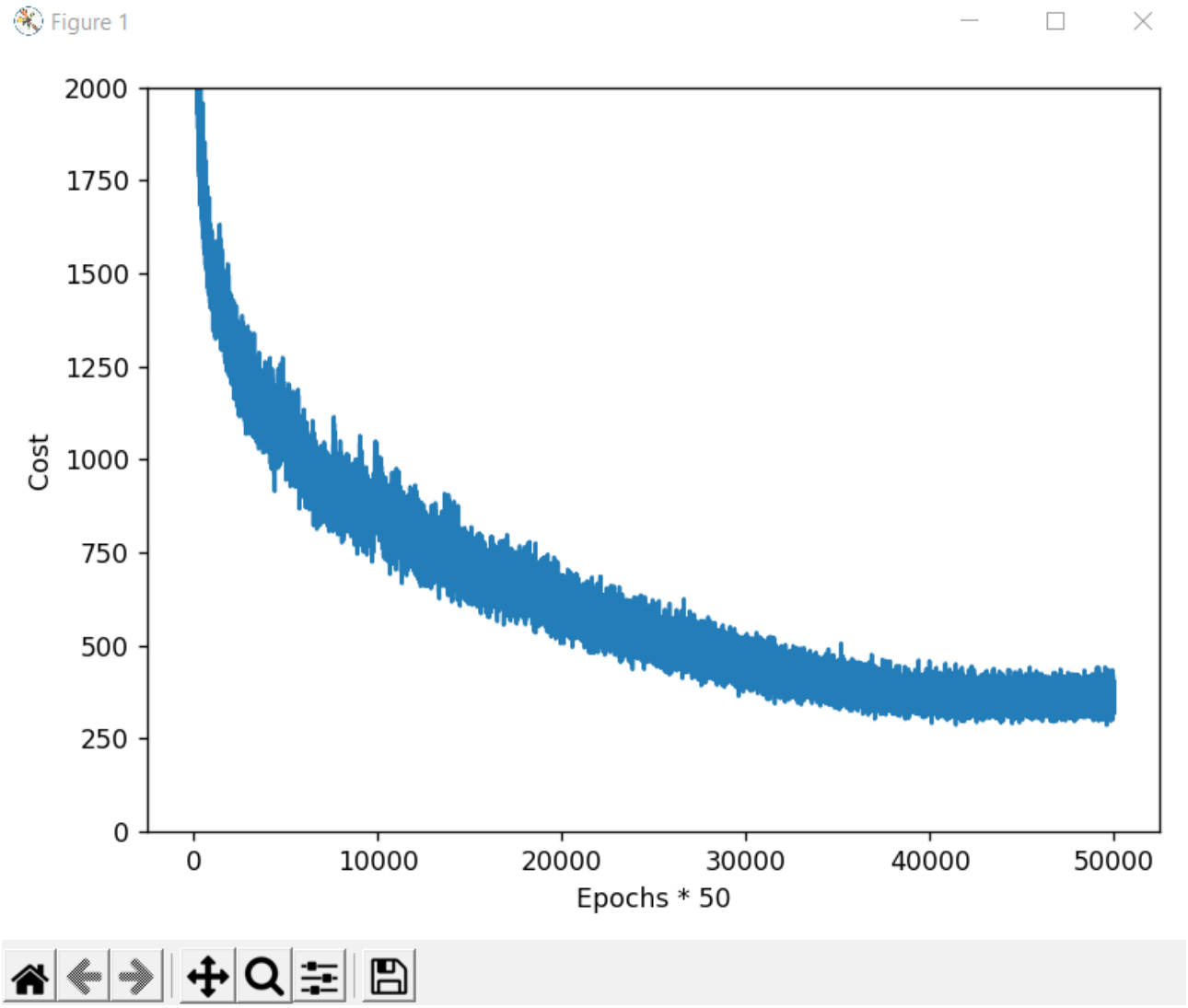
MLP Custom

```python
import os
import sys
import struct
import numpy as np
import matplotlib as plt

from sklearn.neural_network import MLPClassifier
import MLP as mlp

#Les images sont dans le format Byte, on les mets dans un NP
def load_mnist(path, kind='train'):
    """Load MNIST data from `path`"""
    labels_path = os.path.join(path,'%s-labels.idx1-ubyte' % kind)
    images_path = os.path.join(path,'%s-images.idx3-ubyte' % kind)

    with open(labels_path, 'rb') as lbpath:
        magic, n = struct.unpack('>II',lbpath.read(8)) #Utiliser pour la lecture du format, > veut dire Big Endian et I entier positif
        labels = np.fromfile(lbpath,dtype=np.uint8)

    with open(images_path, 'rb') as imgpath:
        magic, num, rows, cols = struct.unpack(">IIII",imgpath.read(16)) #m◆me chose
        images = np.fromfile(imgpath,dtype=np.uint8).reshape(len(labels), 784)

    return images, labels
```

```python
X_train, y_train = load_mnist(r'C:/Users/reyna/Desktop/Travail/UQAC/Trimestre2-Hiver/DeepLearning/Travail4/mnist/', kind='train')
print('Rows: %d, columns: %d' % (X_train.shape[0], X_train.shape[1]))

#Rows: 60000, columns: 784

X_test, y_test = load_mnist(r'C:/Users/reyna/Desktop/Travail/UQAC/Trimestre2-Hiver/DeepLearning/Travail4/mnist/', kind='t10k')
print('Rows: %d, columns: %d' % (X_test.shape[0], X_test.shape[1]))

#Rows: 10000, columns: 784

nn = mlp.NeuralNetMLP(n_output=10,
                      n_features=X_train.shape[1],
                      n_hidden=50,
                      l2=0.1,
                      l1=0.0,
                      epochs=1000,
                      eta=0.001,
                      alpha=0.001,
                      decrease_const=0.00001,
                      minibatches=50,
                      shuffle=True,
                      random_state=1)
```

```
0.9502
PS C:\Users\reyna> & C:/Users/reyna/AppData/Local/Programs/Python/Python39/python.exe c:/Users/reyna/Desktop/Travail/UQAC/Trimestre2-Hiver/DeepLearning/Travail4/MNIST.py
Rows: 60000, columns: 784
Rows: 10000, columns: 784
Epoch: 162/1000
```

```python
nn.fit(X_train, y_train, print_progress=True)

import matplotlib.pyplot as plt

plt.plot(range(len(nn.cost_)), nn.cost_)
plt.ylim([0, 2000])
plt.ylabel('Cost')
plt.xlabel('Epochs * 50')
plt.tight_layout()
# plt.savefig('./figures/cost.png', dpi=300)
plt.show()
```




```python
y_test_pred = nn.predict(X_test)

if sys.version_info < (3, 0):
    acc = ((np.sum(y_test == y_test_pred, axis=0)).astype('float') /
           X_test.shape[0])
else:
    acc = np.sum(y_test == y_test_pred, axis=0) / X_test.shape[0]

print('Test accuracy MLP custom: %.2f%%' % (acc * 100))
```

```
PS C:\Users\reyna> & C:/Users/reyna/AppData/Local/Programs/Python/Python39/python.exe c:/Users/reyna/Desktop/Travail/UQAC/Trimestre2-Hiver/DeepLearning/Travail4/MNIST.py
Rows: 60000, columns: 784
Rows: 10000, columns: 784
Epoch: 1000/1000Test accuracy MLP custom: 96.07%
```

96.07%

Comparaison avec le MLP de scikit learn

```
mlpscikit = MLPClassifier(hidden_layer_sizes=50, alpha= 0.001, learning_rate_init= 0.001, random_state= 1, shuffle=True)
mlpscikit.fit(X_train, y_train)
print("Score MLP scikit learn :")
print(mlpscikit.score(X_test, y_test))
```

```
0.9502
PS C:\Users\reyna> & C:/Users/reyna/AppData/Local/Programs/Python/Python39/python.exe c:/Users/reyna/Desktop/Travail/UQAC/Trimestre2-Hiver/DeepLearning/Travail4/MNIST.py
Rows: 60000, columns: 784
Rows: 10000, columns: 784
Epoch: 1000/1000Test accuracy MLP custom: 96.07%
Score MLP scikit learn :
0.9502
PS C:\Users\reyna>
```

0.9502 soit 95.02%

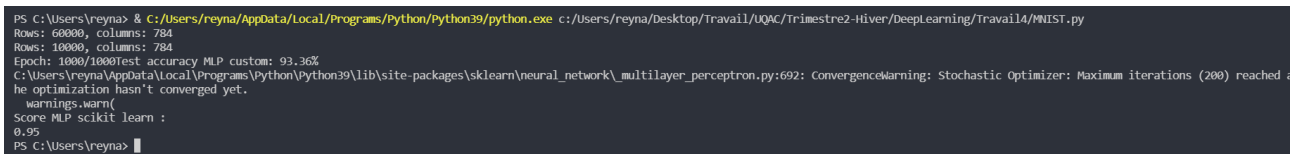
Changement d'hyper paramètre

```
nn = mlp.NeuralNetMLP(n_output=10,
                      n_features=X_train.shape[1],
                      n_hidden=60,
                      l2=0.15,
                      l1=0.0,
                      epochs=1000,
                      eta=0.0001,
                      alpha=0.001,
                      decrease_const=0.00001,
                      minibatches=50,
                      shuffle=True,
                      random_state=1)
```

eta = 0.001 → 0.0001
n_hidden = 50 → 60
l2 = 0.1 → 0.15

```
mlpscikit = MLPClassifier(hidden_layer_sizes=60, alpha= 0.001, learning_rate_init= 0.0001, random_state= 1, shuffle=True)
mlpscikit.fit(X_train, y_train)
print("Score MLP scikit learn :")
print(mlpscikit.score(X_test, y_test))
```

eta = 0.001 → 0.0001
n_hidden = 50 → 60



93.36 % et 0.95 soit 95 %