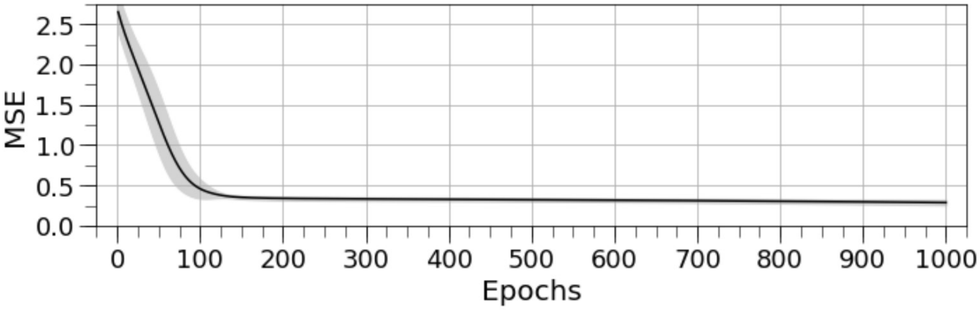
In [3]:

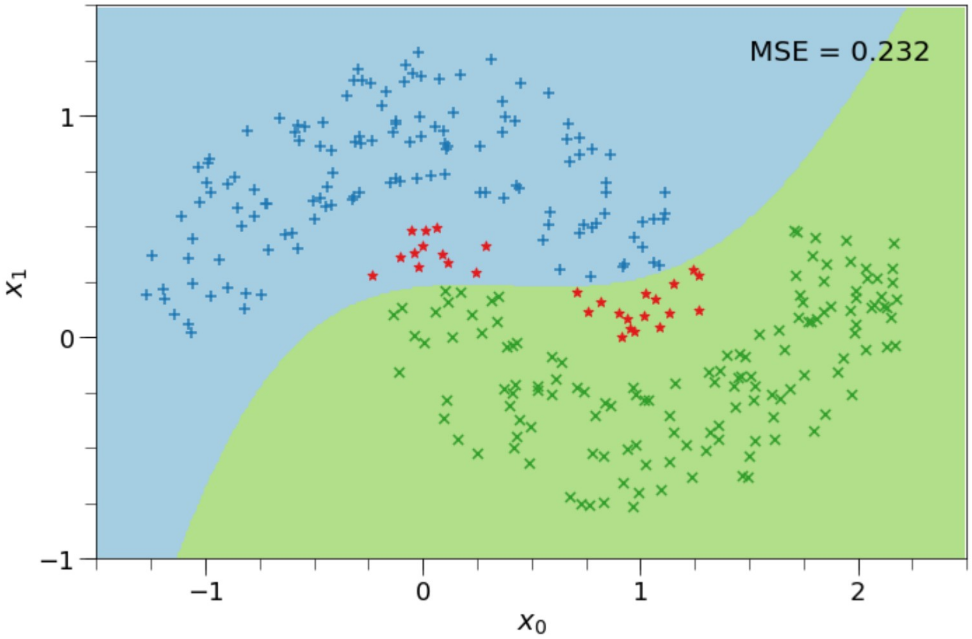
```
# -*- coding: utf-8 -*-
"""
Created on Mon Nov 11 21:51:50 2019

@author: jorge
"""
import numpy as np
from requiredFunctions.trainMLP import trainMLP
from requiredFunctions.MLP import MLP
from requiredFunctions.doubleMoon import doubleMoon
import matplotlib as mpl
import matplotlib.pyplot as plt
import matplotlib.ticker as mtick

height = 10
width = 10

mpl.rcParams['figure.figsize'] = (width, height)
mpl.rcParams['font.size'] = 20
mpl.rcParams['figure.titlesize'] = 'small'
mpl.rcParams['legend.fontsize'] = 'small'
mpl.rcParams['xtick.major.size'] = 12
mpl.rcParams['xtick.minor.size'] = 8
mpl.rcParams['xtick.labelsize'] = 18
mpl.rcParams['ytick.major.size'] = 12
mpl.rcParams['ytick.minor.size'] = 8
mpl.rcParams['ytick.labelsize'] = 18

cmap = plt.get_cmap('Paired')
cmap_scatter = mpl.colors.ListedColormap(cmap((1, 3, 5)))
cmap_contour = mpl.colors.ListedColormap(cmap((0, 2)))
titles = (('a', 'b'), ('c', 'd'), ('e', 'f'))

# Data parameters
N = 300
r = 1
w = 0.6
d = -0.5

# Model parameters
learning_rate = 1e-4
alpha = 0
epochs = 1000
```

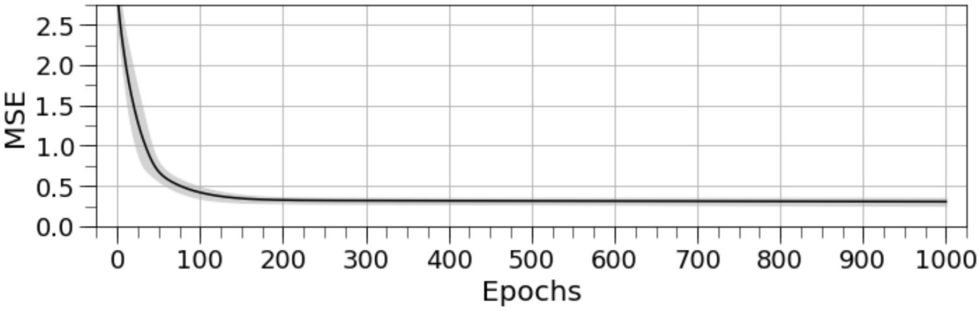
In [4]:

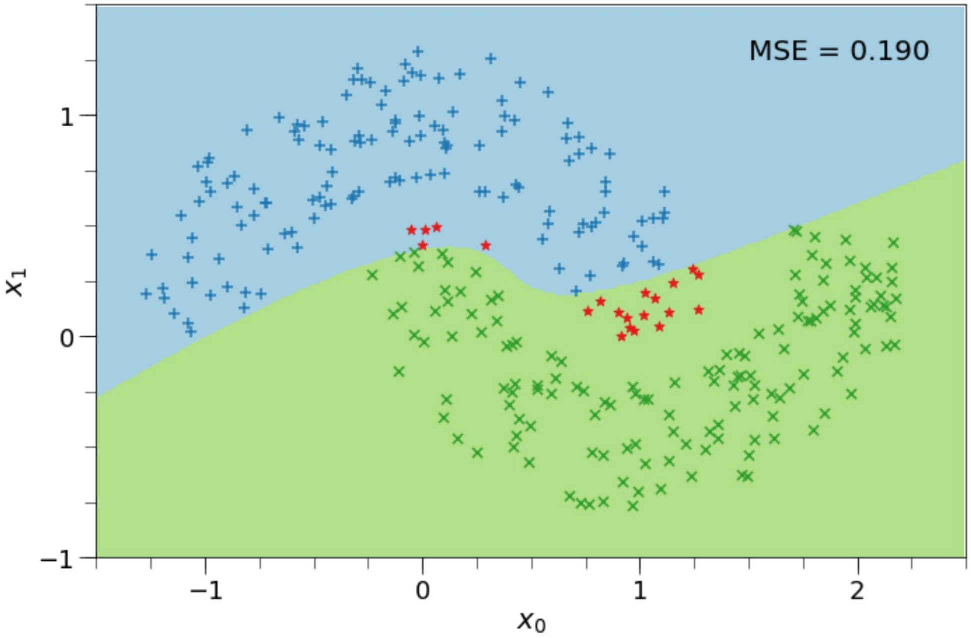
```
layers = {0:[10,], 1:[6,4], 2:[5,3,2]}
best_mse = np.inf
for i in range(3):
    # Part 1
    trials = 10
    trial_mse = np.zeros((trials, epochs))
    for t in range(trials):
        data = doubleMoon(N, w, r, d, seed=t)
        x_train, y_train = data[:,:2], data[:,2]
        x_train, y_train = x_train.T, y_train.reshape(1, len(y_train))
        wh, wo, mse = trainMLP(x_train, y_train, layers[i], learning_rate, alpha, epochs, verbose=False)
        trial_mse[t] = mse.flatten()
        if mse.flatten()[-1] < best_mse:
            best_wh, best_wo, best_mse = wh, wo, mse.flatten()[-1]

    mse_avg = trial_mse.mean(axis=0)
    mse_std = trial_mse.std(axis=0)
    print(len(layers[i]), 'Layers, Best Final MSE:', best_mse)

    epoch_grid = np.arange(1, epochs+1, 1)

    fig = plt.figure()
    ax = [plt.subplot2grid((3,1), (0,0), colspan=1, rowspan = 1, fig=fig),
          plt.subplot2grid((3,1), (1,0), colspan=1, rowspan = 2, fig=fig)]

    ax[0].plot(epoch_grid, mse_avg, color='black')
    ax[0].fill_between(epoch_grid, mse_avg - mse_std, mse_avg + mse_std, color='lightgray')
    ax[0].xaxis.set_major_locator(mtick.MultipleLocator(100))
    ax[0].xaxis.set_minor_locator(mtick.MultipleLocator(25))
    ax[0].yaxis.set_major_locator(mtick.MultipleLocator(0.5))
    ax[0].yaxis.set_minor_locator(mtick.MultipleLocator(0.25))
    ax[0].set_xlim(-25, epochs+25)
    ax[0].set_ylim(0, 2.75)
    ax[0].set_xlabel('Epochs')
    ax[0].set_ylabel('MSE')
    ax[0].grid()

    # Part 2
    test_data = doubleMoon(N, w, r, d, seed=100)
    x_test, y_test = test_data[:,:2], test_data[:,2]
    x_test, y_test = x_test.T, y_test.reshape(1, len(y_test))

    #x_test, y_test = x_train, y_train

    y_pred = MLP(x_test, best_wh, best_wo).flatten()
    y_test = y_test.flatten()
    x_test = x_test.T
    y_pred[y_pred > 0] = 1
    y_pred[y_pred < 0] = -1

    y_right = np.where(y_pred == y_test)[0]
    y_wrong = np.where(y_pred != y_test)[0]
    blue_ind = y_right[np.where(y_test[y_right]==-1)]
    green_ind = y_right[np.where(y_test[y_right]==1)]

    x0_min, x0_max = -1.5, 2.5
    x1_min, x1_max = -1, 1.5
    xx0, xx1 = np.meshgrid(np.arange(x0_min, x0_max, 0.01),
                           np.arange(x1_min, x1_max, 0.01))
    cc = MLP(np.c_[xx0.ravel(), xx1.ravel()].T, best_wh, best_wo)
    cc[cc > 0] = 1
    cc[cc < 0] = -1
    cc = cc.reshape(xx0.shape)

    ax[1].contourf(xx0, xx1, cc, cmap=cmap_contour)
    ax[1].scatter(x_test[:,0][blue_ind], x_test[:,1][blue_ind], 50,
                  c=[cmap_scatter(0)], marker='+')
    ax[1].scatter(x_test[:,0][green_ind], x_test[:,1][green_ind], 50,
                  c=[cmap_scatter(1)], marker='x')
    ax[1].scatter(x_test[:,0][y_wrong], x_test[:,1][y_wrong], 50,
                  c=[cmap_scatter(2)], marker='*')
    ax[1].set_xlim(x0_min, x0_max)
    ax[1].set_ylim(x1_min, x1_max)
    ax[1].xaxis.set_major_locator(mtick.MultipleLocator(1))
    ax[1].xaxis.set_minor_locator(mtick.MultipleLocator(0.25))
    ax[1].yaxis.set_major_locator(mtick.MultipleLocator(1))
    ax[1].yaxis.set_minor_locator(mtick.MultipleLocator(0.25))
    ax[1].set_xlabel(r'$x_0$')
    ax[1].set_ylabel(r'$x_1$')
    ax[1].text(1.5, 1.25, 'MSE = {:.3f}'.format(best_mse))
    fig.tight_layout(pad=0.5)
    #fig.savefig('../prob1' + str(titles[i][0]) + str(titles[i][1]) + '.eps', dpi=500)
```

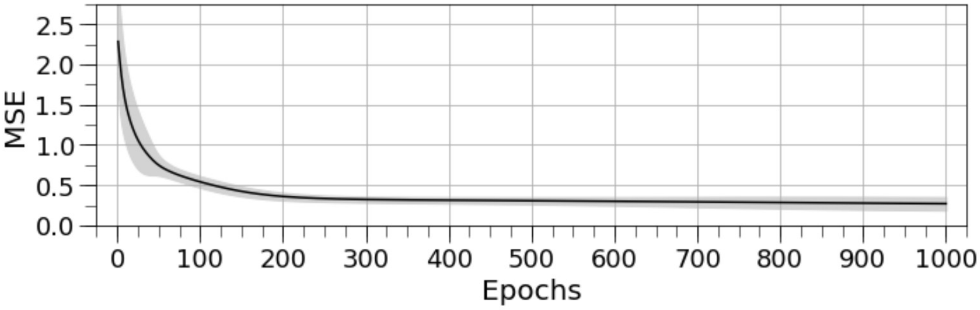
```
1 Layers, Best Final MSE: 0.23182632470268116
2 Layers, Best Final MSE: 0.18986019943346888
3 Layers, Best Final MSE: 0.10508344094404794
```

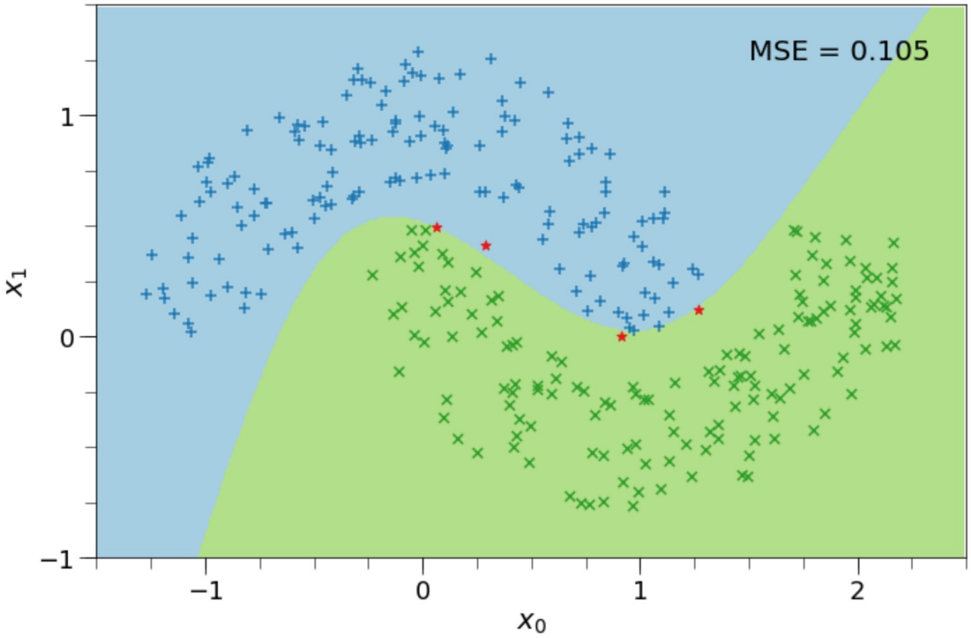





MSE
2.5
2.0
1.5
1.0
0.5
0.0
0
100
200
300
400
500
600
700
800
900
1000
Epochs


MSE = 0.190
x_1
1
0
−1
−1
0
1
2
x_0


MSE
2.5
2.0
1.5
1.0
0.5
0.0
0
100
200
300
400
500
600
700
800
900
1000
Epochs


MSE = 0.105
x_1
1
0
−1
−1
0
1
2
x_0