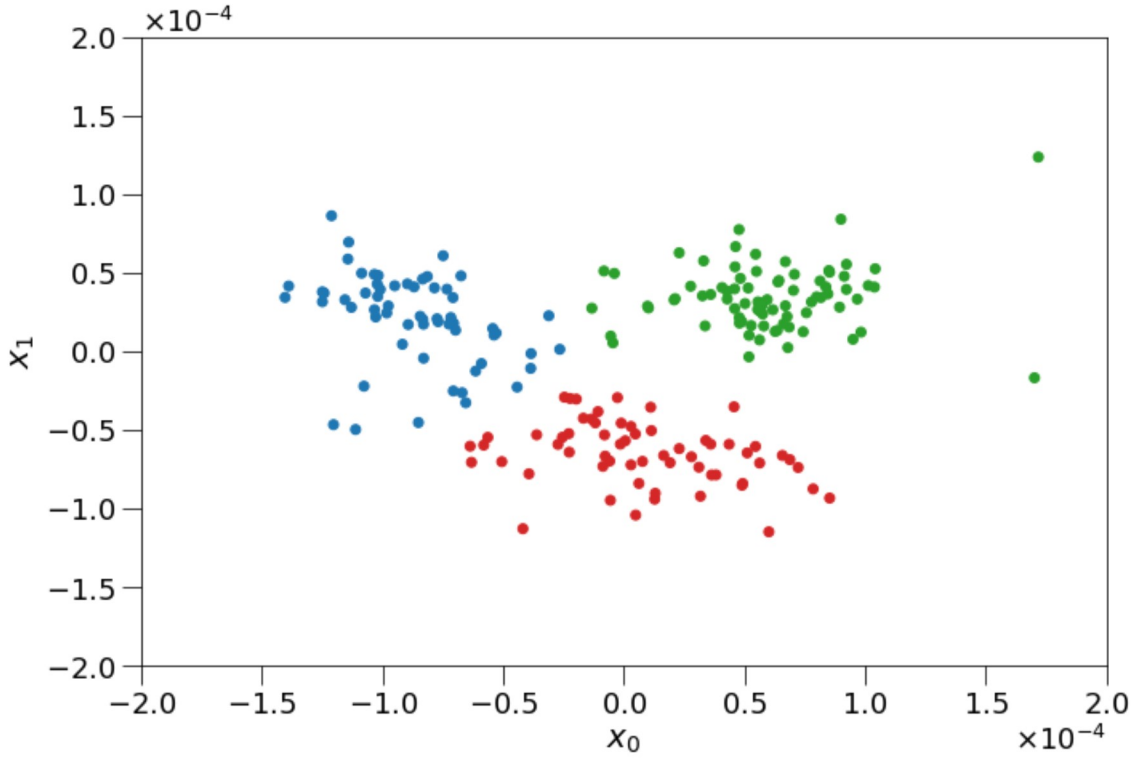
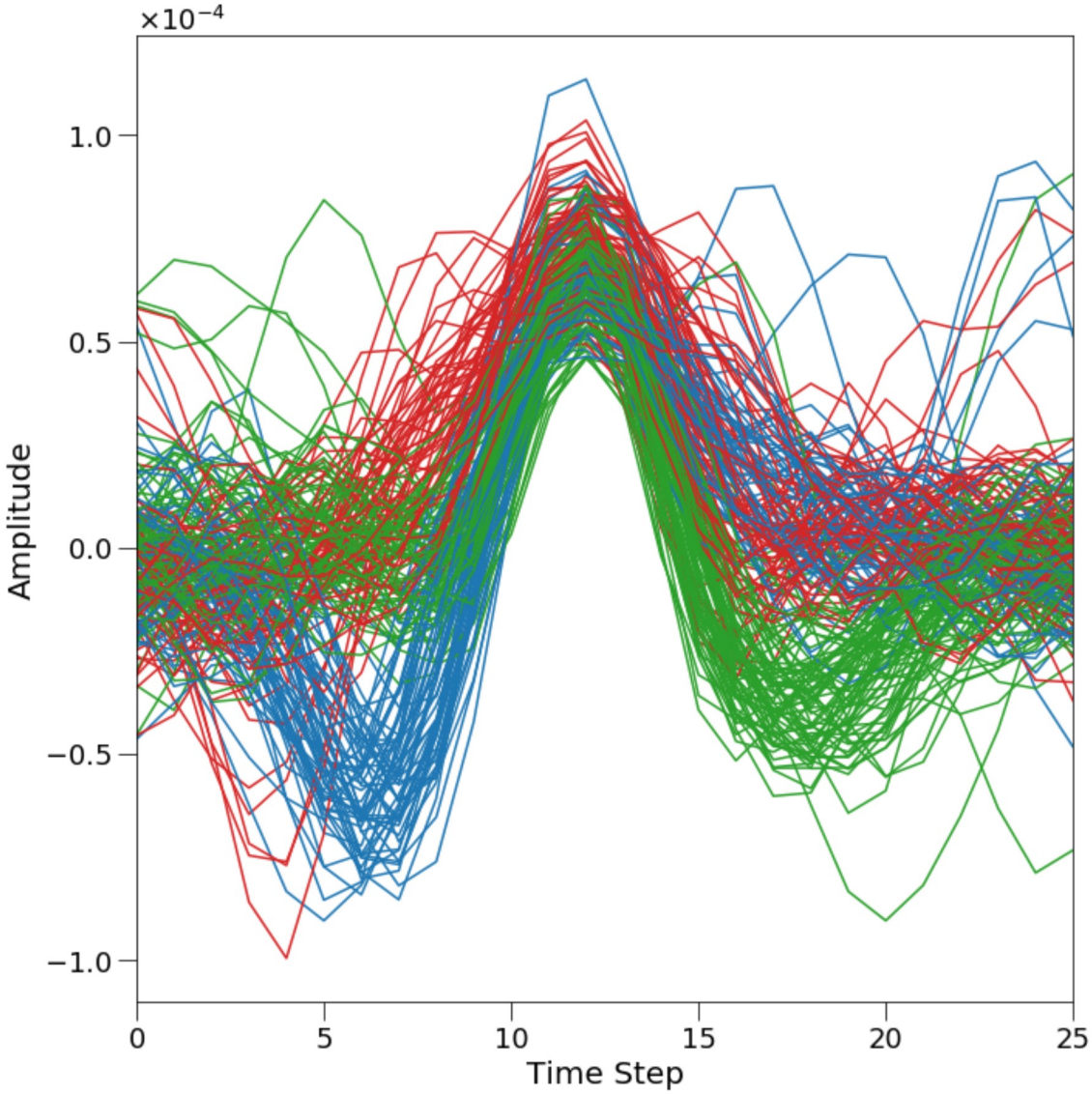
In [5]:
```
# -*- coding: utf-8 -*-
"""
Created on Sun Nov 17 10:10:48 2019

@author: jorge
"""
import numpy as np
import pandas as pd
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score
from requiredFunctions.kMeans import KMeans
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

spikes = pd.read_csv('../data/spikes.csv', header=None)
data = spikes.values

components = 2
pca = PCA(n_components=components, svd_solver='full')
data_trans = pca.fit_transform(data)

trials = 100
k_clusters = np.arange(2, 6, 1)
mse_per_clusters = np.zeros((len(k_clusters),2))
silh_per_clusters = np.zeros((len(k_clusters),2))
best_centroids = None
best_mse = np.inf
best_silh = 0
for i, k in enumerate(k_clusters):
    mse_k = np.zeros(trials)
    silh_k = np.zeros(trials)
    for t in range(trials):
        kmeans = KMeans()
        kmeans.fit_batch(data_trans, k, seed=t)
        mse_k[t] = kmeans.eval_cost(data_trans)
        silh_k[t] = silhouette_score(data_trans, kmeans.predict_cluster(data_trans))
        if silh_k[t] > best_silh:
            best_mse = mse_k[t]
            best_silh = silh_k[t]
            best_centroids = kmeans.centroids
    mse_per_clusters[i,0] = mse_k.mean()
    mse_per_clusters[i,1] = mse_k.std()
    silh_per_clusters[i,0] = silh_k.mean()
    silh_per_clusters[i,1] = silh_k.std()

kmeans.centroids = best_centroids
labels = kmeans.predict_cluster(data_trans)
```

In [6]:
```
cmap = plt.get_cmap('tab20')
cmap_scatter = mpl.colors.ListedColormap(cmap((0, 6, 4)))
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].errorbar(k_clusters, silh_per_clusters[:,0], yerr=silh_per_clusters[:,1],
               color='black', capsize=5, label='Testing')
ax[0].xaxis.set_major_locator(mtick.MultipleLocator(1))
ax[0].xaxis.set_minor_locator(mtick.MultipleLocator(1))
ax[0].yaxis.set_major_locator(mtick.MultipleLocator(0.1))
ax[0].yaxis.set_minor_locator(mtick.MultipleLocator(0.025))
ax[0].set_ylim(0.375, 0.65)
ax[0].set_xlabel('Number of Clusters')
ax[0].set_ylabel('Silhouette Score')
ax[1].scatter(data_trans[:,0], data_trans[:,1], c=labels, cmap=cmap_scatter)
ax[1].set_xlim(-0.00020, 0.00020)
ax[1].set_ylim(-0.00020, 0.00020)
ax[1].set_ylabel(r'$x_1$')
ax[1].set_xlabel(r'$x_0$')
ax[1].ticklabel_format(axis='both', style='sci', scilimits=(-4,-4), useMathText=True)
fig.tight_layout(pad=0.5)
fig.savefig('../prob5e.eps', dpi=500)

time = np.arange(0, 26)
fig1, ax1 = plt.subplots()
for i, l in enumerate(labels):
    ax1.plot(time, data[i], color=cmap_scatter.colors[l])
ax1.ticklabel_format(axis='y', style='sci', scilimits=(-4,-4), useMathText=True)
ax1.set_xlim(0, 25)
ax1.set_xlabel('Time Step')
ax1.set_ylabel('Amplitude')
fig1.tight_layout(pad=0.5)
#fig1.savefig('../prob5f.eps', dpi=500)
```


Silhouette Score
0.6
0.5
0.4
2
3
4
5
Number of Clusters


×10⁻⁴
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
x1
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
x0
×10⁻⁴


×10−4
1.0
0.5
0.0
−0.5
−1.0
Amplitude
0
5
10
15
20
25
Time Step