Practica 1

Daniel López Acero Antonio Luis Suraez Solis

1. Regresión lineal con una variable

En la primera parte de la practica hay que aplicar el método de regresión lineal sobre los datos del fichero ex1data1.csv

In [5]:

```
#IMPORTS
import numpy as np
import matplotlib.pyplot as plt
from pandas.io.parsers import read_csv
import scipy as sp
import time
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
```

In [8]:

```
def carga_csv(file_name):
    """carga el fichero csv especificado y lo
 devuelve en un array de numpy
    """
    valores = read_csv(file_name, header=None).to_numpy()
    # suponemos que siempre trabajaremos con float
    return valores.astype(float)
```

In [5]:

```
def costeFun(theta, X, Y):  #entendamos 0 como theta
    # H0(x(i)) =
    H = np.dot(X,theta)
    #H0(X(i)- Y(i)) ^2 = interior del sumatiorio
    temp = (H-Y)**2
    #1/2m * sumatiorio = J(0) --> coste a devolver
    return temp.sum() / (2*len(X))
```

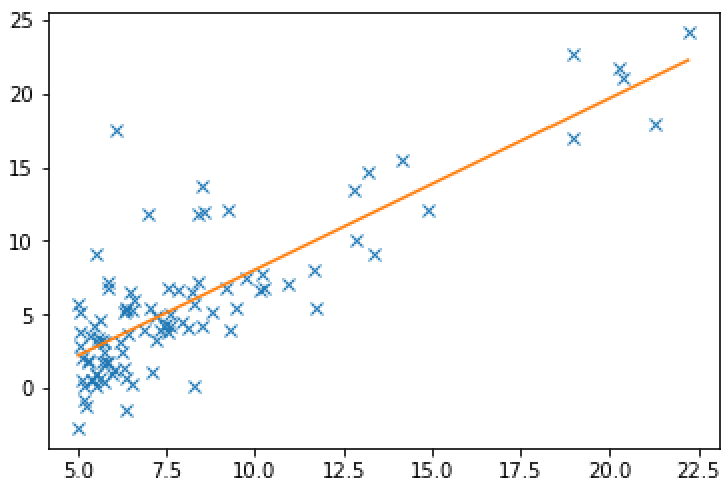
In [10]:

```
def descenso_grad(X, Y, alpha):
    theta0 =0
    theta1 =0
    thetaFinal = [theta0,theta1]
    costeaux = costeFun(thetaFinal,X,Y)
    print(costeaux)
    print("La de encima es la primera y deberia ser 32.07")
    for n in range (1500):
        sumatiorioT0 = 0
        sumatiorioT1 = 0
        for i in range(len(X)):
           sumatiorioT0 += (theta0 + theta1 * X[i,1]) - Y[i]
           sumatiorioT1 += ((theta0+theta1 * X[i,1]) -Y[i]) * X[i,1]
        theta0 = theta0 - (alpha/len(X)) * sumatiorioT0
        theta1 = theta1 - (alpha/len(X)) * sumatiorioT1
        thetaFinal = [theta0,theta1]
        costes=costeFun(thetaFinal,X,Y)
    thetaFinal = [theta0,theta1]
    costes = costeFun(thetaFinal, X, Y)
    return thetaFinal, costes

datos = carga_csv('c:/Users/Daniel/Desktop/AprendizajeAutomatico/AprendizajeAutomatico/P1/ex1data1.csv'
)
X = datos[:, :-1]
np.shape(X)
Y = datos[:, -1]
np.shape(Y)
alpha = 0.01
m = np.shape(X)[0]
n = np.shape(X)[1]
X = np.hstack([np.ones([m,1]), X])

Theta, costes = descenso_grad(X,Y,alpha)



plt.plot(X[:, 1:], Y, "x")
min_x = min(X[:, 1:])
max_x = max(X[:, 1:])
min_y = Theta[0] + Theta[1] * min_x
max_y = Theta[0] + Theta[1] * max_x
plt.plot([min_x, max_x], [min_y, max_y])
```

```
32.072733877455676
La de encima es la primera y deberia ser 32.07
```

Out[10]:

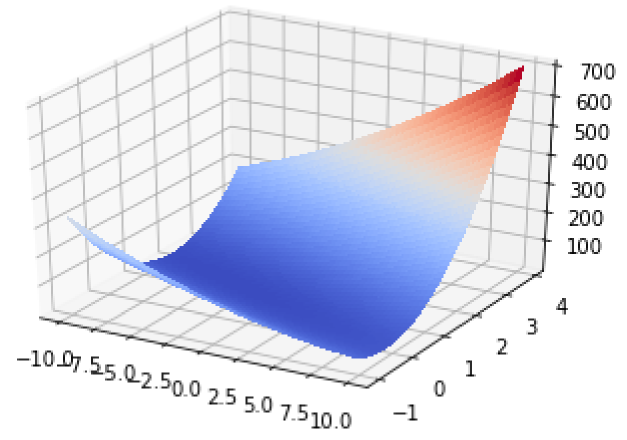
```
[<matplotlib.lines.Line2D at 0x1ef0f9777c0>]
```

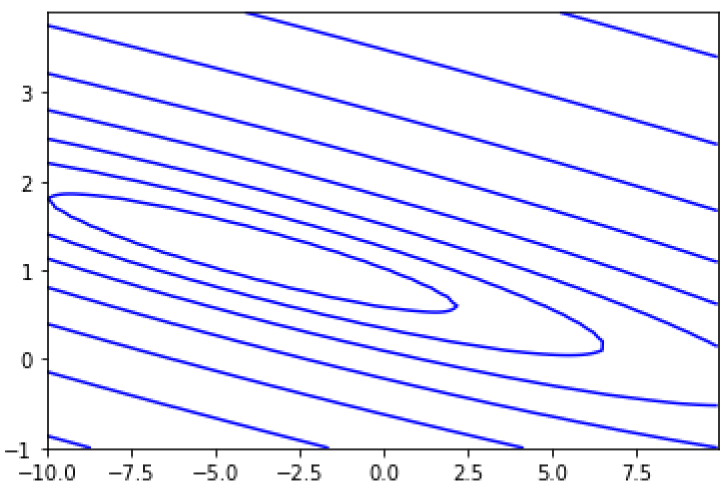


1.1. Visualización de la función de coste

In [12]:

```
def make_data2(t0_range , t1_range , X , Y ):
    step=0.1
    Theta0=np.arange(t0_range[0],t0_range[1],step)
    Theta1=np.arange(t1_range[0],t1_range[1],step)
    Theta0,Theta1 =np.meshgrid(Theta0,Theta1)
    Coste = np.empty_like(Theta0)
    for ix,iy in np.ndindex(Theta0.shape):
        Coste[ix,iy] = costeFun([Theta0[ix,iy], Theta1[ix,iy]], X, Y)
    return [Theta0,Theta1,Coste]


aux = make_data2([-10,10],[-1,4], X, Y)
eje3D = np.logspace(-2,3,20)
fig=plt.figure()
#ax=fig.gca(projection='3d')
surf=fig.gca(projection = '3d').plot_surface(aux[0], aux[1],aux[2],cmap=cm.coolwarm,linewidth=0,antiali
ased=False)


#fig.colorbar(surf,shrink=0.5,aspect=5)
fig2 = plt.figure()
aux2 =fig2.gca()
surf2 = aux2.contour(aux[0], aux[1], aux[2],eje3D ,colors = 'blue')
#surf2 = plt.contour(aux[0],aux[1],aux[2],eje3D,colors='blue')

#surf2.clabel(surf2, inline=1, fontsize=10)
#ax.set_title('Movidas')

plt.show()
```


700
600
500
400
300
200
100
−10.0
7.5
5.0
2.5
0.0
2.5
5.0
7.5
10.0
−1
0
1
2
3
4


3
2
1
0
−1
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5