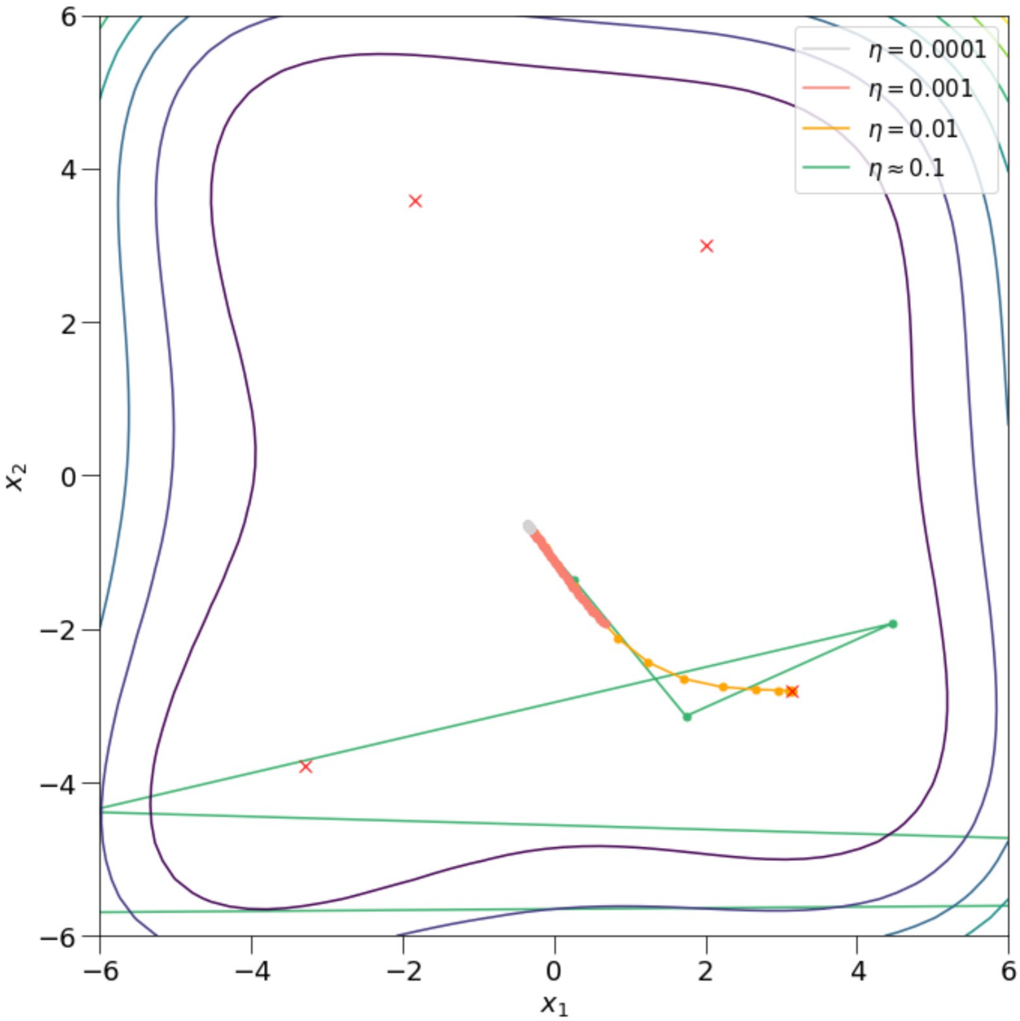
In [9]:

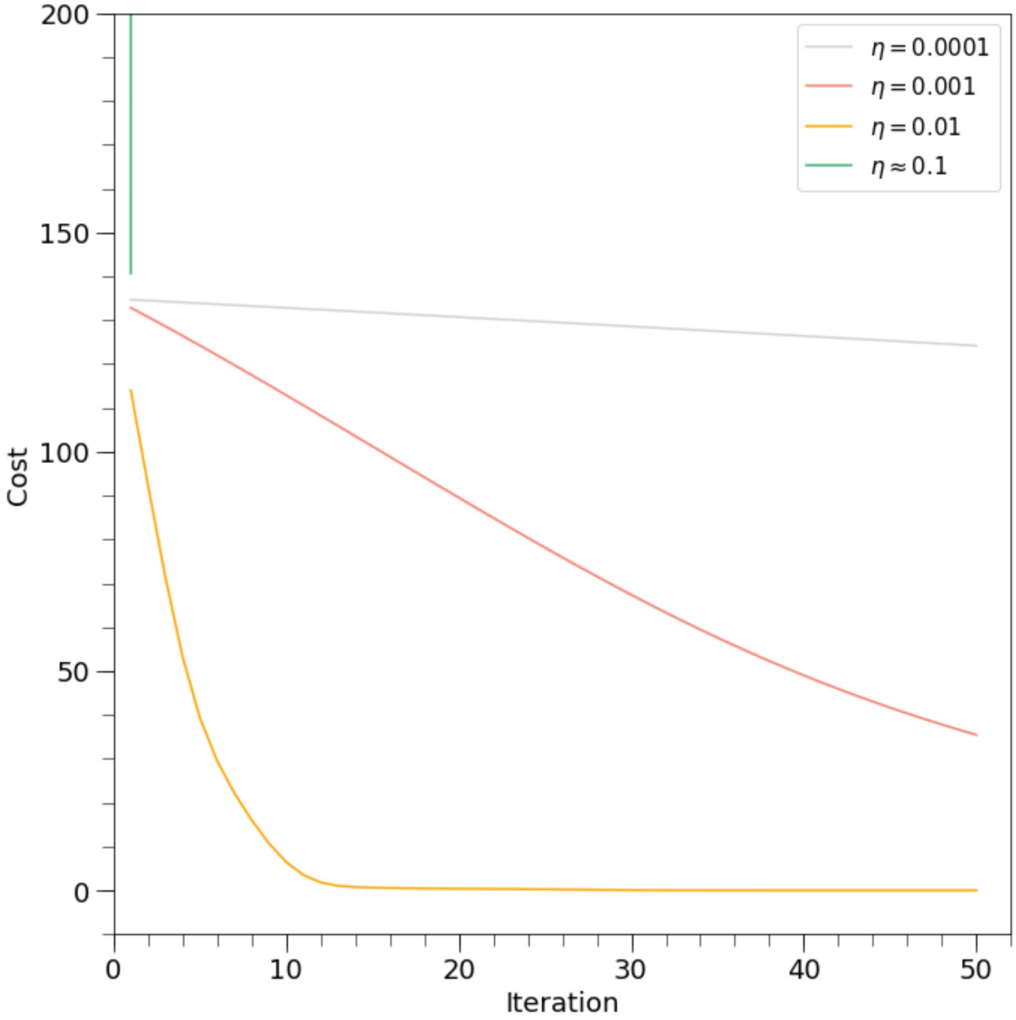
```
# -*- coding: utf-8 -*-
"""
Created on Sat Oct 26 22:11:59 2019

@author: jorge
"""

import sympy as sym
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl
import matplotlib.ticker as mtick
#get_ipython().run_line_magic('matplotlib', 'inline')
#https://scipy-lectures.org/packages/sympy.html
# ^^ how to use sympy ^^

height = 10
width = 10

mpl.rcParams['figure.figsize'] = (width, height)
mpl.rcParams['font.size'] = 18
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

HX ,HY  = 50,50 #number of x,y  points for countour
xmin,xmax = -6,6
ymin,ymax = -6,6
x1 = np.linspace(xmin,xmax,HX)
x2 = np.linspace(ymin,ymax,HY)
X1,X2 = np.meshgrid(x1,x2) # genertate mesh grid
w1=sym.Symbol('w1') # define symbols
w2=sym.Symbol('w2')

j=(w1**2 + w2 - 11)**2 + (w1 + w2**2 - 7)**2# define equation

#compute gradient
j_grad1=sym.diff(j,w1)
j_grad2=sym.diff(j,w2)
#compute hessian
hess11=sym.diff(j_grad1,w1)
hess12=sym.diff(j_grad1,w2)
hess21=sym.diff(j_grad2,w1)
hess22=sym.diff(j_grad2,w2)

#generate contour map
ConMap=np.zeros((HX,HY))
for i in range(HX):
    for k in range(HY):
        ConMap[i,k]=j.subs({w1:x1[i],w2:x2[k]})

logetas = np.array([-4, -3, -2, -1.25], dtype=np.float)
etas = 10**logetas
lim=50 # number of iterations
threshold = 1e-5
jw_threshold = 1e-5
trials=30
iters=[] # number of iterations until conveged
wStar=[[3,2],[-2.8,3.13],[-3.78,-3.28],[3.58,-1.85]] # ending point

colors = ('lightgray', 'salmon', 'orange', 'mediumseagreen')
labels = [r'$\eta =$' + str(eta) for eta in etas]
labels[-1] = r'$\eta\approx 0.1$'
```

In [10]:

```
fig, ax = plt.subplots()
ax.contour(X1,X2,ConMap)

fig2, ax2 = plt.subplots()

for i in range(len(etas)):
    np.random.seed(0)
    cost_trials = np.zeros((trials, lim))
    for t in range(trials):
        w=np.random.normal(0, 1, 2) # starting point
        ew=[]
        jw=[]
        count = 1
        line=[]
        while(True):
            #compute gradient matrix and hessian matrix
            g= np.array([float(j_grad1.subs({w1:w[0],w2:w[1]})),float(j_grad2.subs({w1:w[0],w2:w
[1]}))])
            H= np.array([[float(hess11.subs({w1:w[0],w2:w[1]})),float(hess12.subs({w1:w[0],w2:w
[1]}))],
                        [float(hess21.subs({w1:w[0],w2:w[1]})),float(hess22.subs({w1:w[0],w2:w
[1]}))]])

            dw = -g
            eta = etas[i]

            wnew =  w+eta*dw
            if(np.abs(wnew - w).mean() > threshold):
                line.append(w)
            #loop check
            #if(np.abs(wnew - w).mean() < threshold):
                #print('Converge break')
            #    break
            if( count>lim ):
                #print('Count Break')
                break
            #elif(np.isnan(g).any()):
                #print('nan break')
            #    break
            else:
                count=count +1
                wprev=w.copy()

                w=wnew.copy()

                jw.append(j.subs({w1:w[0],w2:w[1]}))
                #if(jw[-1] <= jw_thresh):
                #            iters.append(count)
                #            break
        cost_trials[t] = jw
        line=np.array(line)
    ax.plot(line[:,1],line[:,0], color=colors[i], label=labels[i], zorder=len(etas)-i)
    ax.scatter(line[:,1],line[:,0], 25, color=colors[i], zorder=len(etas)-i)

    cost_avg = cost_trials.mean(axis=0)
    ax2.plot(np.arange(1,lim+1,1), cost_avg, color=colors[i], label=labels[i])

for n in range(len(wStar)):
    ax.plot(wStar[n][1],wStar[n][0], 'rx', markersize=8)
ax.set_xlim(xmin, xmax)
ax.set_ylim(ymin, ymax)
ax.set_xlabel(r'$x_1$')
ax.set_ylabel(r'$x_2$')
ax.legend()
fig.tight_layout()
#fig.savefig('../prob6a.eps', dpi=500)

ax2.xaxis.set_major_locator(mtick.MultipleLocator(10))
ax2.xaxis.set_minor_locator(mtick.MultipleLocator(2))
ax2.yaxis.set_major_locator(mtick.MultipleLocator(50))
ax2.yaxis.set_minor_locator(mtick.MultipleLocator(10))
ax2.set_ylim(-10, 200)
ax2.set_xlim(0, 52)
ax2.set_xlabel('Iteration')
ax2.set_ylabel('Cost')
ax2.legend()
fig2.tight_layout()
#fig2.savefig('../prob6b.eps', dpi=500)
```

```
C:\Users\jorge\Anaconda3\lib\site-packages\ipykernel_launcher.py:24: RuntimeWarning: invalid valu
e encountered in add
```




$\eta = 0.0001$
$\eta = 0.001$
$\eta = 0.01$
$\eta \approx 0.1$
Cost
200
150
100
50
0
Iteration
0
10
20
30
40
50