Machine Intelligence ex 04 solution

November 15, 2017

```
In [167]: import numpy as np
          from numpy import genfromtxt
          import pandas as pd

          %matplotlib inline
          import matplotlib.pyplot as plt

In [168]: """
          defining our variables: i did it in functions, since our variables will change later
          (in the end it was almost unnnecessary, since i could have done it like g = np.dot(r
          I'm sorry if this got quite confusing in the b and c,.

          The most important thing to know by my solution:

          g = costfunction()


          the functions are self-explanatory
          """

          data = str.split("""-1 -0.1
          0.3 0.5
          2 0.5""", "\n")

          data
          inputs = [float(str.split(data[i]," ")[0]) for i in range(3)]
          targets = [float(str.split(data[i]," ")[1]) for i in range(3)]
          inputs


          def returnX():
             # X = np.matrix('1 inputs[0]; 3 inputs[1];3 inputs[2]')
              X= np.matrix([[1, -1],[1, 0.3],[1, 2] ]).T;
              return X;

          def returny ():
             # X = np.matrix('1 inputs[0]; 3 inputs[1];3 inputs[2]')
```

```
    y = (np.matrix([[targets[0],targets[1],targets[2]]])).T;
    return y;

def returnb():
   # X = np.matrix('1 inputs[0]; 3 inputs[1];3 inputs[2]')
    b = -(np.dot(returnX(),returny()))
    return b;

def returnH ():
   # X = np.matrix('1 inputs[0]; 3 inputs[1];3 inputs[2]')
    H = np.dot(returnX(),(returnX()).T)
    return H;
def costfunction ():
    return np.dot(returnH(),w) + returnb();
```

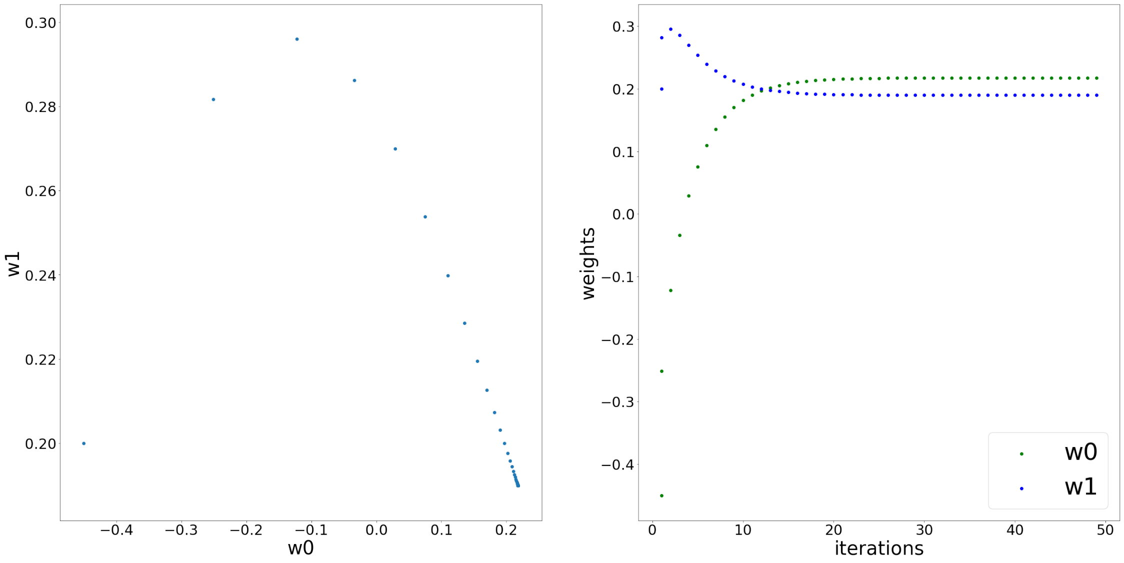
```
In [169]: #A

          i =1;

          w= np.matrix([-0.45,0.2]).T;
          first_iteration = w -0.1*costfunction();
          """
          Xm=np.array([[1, -1],[1, 0.3],[1, 2] ]).transpose()
          t=np.array([[ -0.1, 0.5, 0.5 ]])
          H=np.dot(Xm, Xm.transpose())
          b= - (np.dot(Xm, t.transpose()))
          g= lambda w: np.dot(H, w) + b
          #w = np.random.uniform(0.0, 1.0, [2, 1])
          n = 0.2

          err = lambda w: 0.5*np.dot((np.dot(Xm.transpose(), w)-t.transpose()).transpose(),
          (np.dot(Xm.transpose(), w)-t.transpose()))[0][0]
          """

          full_df_weights = pd.DataFrame({'iteration':[i],'w1':[-0.45],'w2':[0.2]})
          for i in range(1, 1000):
              #the (0.1 is my learningcurve)
              #Et = err(w)
              w = w -0.1*costfunction();
             # print (w);
              weight_updates = pd.DataFrame({'iteration':[i],'w1':[w[0]],'w2':[w[1]]},index=[(
              full_df_weights = pd.concat([full_df_weights,weight_updates])
              i=i+1

          print(full_df_weights)

          print('left: the link between w0 and w1 | Right: the evolution of the weights during
          print('We see that FULL convergence takes many iterations')
```

```
plt.figure(figsize=(40,20))

plt.subplot(1, 2, 1)
scatter1 = plt.scatter(full_df_weights.w1, full_df_weights.w2)
plt.xlabel('w0', fontsize=40)
plt.ylabel('w1', fontsize=40)
plt.suptitle('left: the link between w0 and w1 | Right: the evolution of the weights
             fontsize=40)
plt.tick_params(axis='both', which='major', labelsize=30)



#plot2 = plt.plot(full_df_weights.w1, full_df_weights.w2,Et)
plt.subplot(1, 2, 2)
plt.tick_params(axis='both', which='major', labelsize=30)
plt.scatter(full_df_weights.iteration[0:50],full_df_weights.w1[0:50],color='green', 
plt.scatter(full_df_weights.iteration[0:50],full_df_weights.w2[0:50],  color='blue',

plt.xlabel('iterations', fontsize=40)
plt.ylabel('weights', fontsize=40)

plt.legend(fontsize=50)
#plt.legend(loc='best')

plt.show()
plt.gcf().clear()
```

```
     iteration                     w1                     w2
0            1                  -0.45                    0.2
1            1        [[[[[-0.251]]]]]      [[[[[ 0.2817]]]]]
2            2     [[[[[-0.122321]]]]]   [[[[[ 0.2959447]]]]]
3            3   [[[[[-0.03409751]]]]]  [[[[[ 0.28621058]]]]]
4            4   [[[[[ 0.02892437]]]]]  [[[[[ 0.26996207]]]]]
5            5   [[[[[ 0.07515199]]]]]  [[[[[ 0.25379121]]]]]
6            6   [[[[[ 0.10961353]]]]]  [[[[[ 0.23984173]]]]]
7            7   [[[[[ 0.13555005]]]]]  [[[[[ 0.22851253]]]]]
8            8   [[[[[ 0.15517841]]]]]  [[[[[ 0.21957814]]]]]
9            9   [[[[[ 0.17007973]]]]]  [[[[[ 0.21263968]]]]]
10          10   [[[[[ 0.18141265]]]]]  [[[[[ 0.20729572]]]]]
11          11   [[[[[ 0.19004041]]]]]  [[[[[ 0.20319855]]]]]
12          12   [[[[[ 0.19661248]]]]]  [[[[[ 0.20006524]]]]]
13          13   [[[[[ 0.20162025]]]]]  [[[[[ 0.19767241]]]]]
14          14   [[[[[ 0.20543676]]]]]  [[[[[ 0.19584652]]]]]
15          15   [[[[[ 0.20834569]]]]]  [[[[[ 0.19445386]]]]]
16          16   [[[[[ 0.21056298]]]]]  [[[[[ 0.19339191]]]]]
17          17   [[[[[ 0.21225314]]]]]  [[[[[ 0.19258224]]]]]
18          18   [[[[[ 0.21354151]]]]]  [[[[[ 0.19196497]]]]]
```

```
19          19  [[[[ 0.21452361]]]]  [[[[ 0.19149441]]]]
20          20  [[[[ 0.21527225]]]]  [[[[ 0.19113568]]]]
21          21  [[[[ 0.21584294]]]]  [[[[ 0.19086223]]]]
22          22  [[[[ 0.21627797]]]]  [[[[ 0.19065377]]]]
23          23  [[[[ 0.21660959]]]]  [[[[ 0.19049487]]]]
24          24  [[[[ 0.21686238]]]]  [[[[ 0.19037373]]]]
25          25  [[[[ 0.21705508]]]]  [[[[ 0.19028139]]]]
26          26  [[[[ 0.21720197]]]]    [[[[ 0.190211]]]]
27          27  [[[[ 0.21731395]]]]  [[[[ 0.19015735]]]]
28          28  [[[[ 0.21739931]]]]  [[[[ 0.19011644]]]]
29          29  [[[[ 0.21746438]]]]  [[[[ 0.19008526]]]]
..         ...                   ...                   ...
970        970  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
971        971  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
972        972  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
973        973  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
974        974  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
975        975  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
976        976  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
977        977  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
978        978  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
979        979  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
980        980  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
981        981  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
982        982  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
983        983  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
984        984  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
985        985  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
986        986  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
987        987  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
988        988  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
989        989  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
990        990  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
991        991  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
992        992  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
993        993  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
994        994  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
995        995  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
996        996  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
997        997  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
998        998  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
999        999  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]

[1000 rows x 3 columns]
left: the link between w0 and w1 | Right: the evolution of the weights during the iterations.
We see that FULL convergence takes many iterations
```

left: the link between w0 and w1 | Right: the evolution of the weights during the iterations.



```
<matplotlib.figure.Figure at 0x278ce5a3f60>
```

```
In [170]: #B
          plt.gcf().clear()
          i =0;
          w= np.matrix([-0.45,0.2]).T;

          full_df_weights_line = pd.DataFrame({'iteration':[i],'w1':[-0.45],'w2':[0.2]})
          learning_rate = (np.dot(costfunction().T,costfunction())
                           /
                           np.dot(costfunction().T,np.dot(returnH(),costfunction())))

          ln = np.squeeze(np.asarray(learning_rate))
          """
          print(learning_rate)
          print(costfunction())

          test = w = w -np.dot(ln,costfunction(),out=None)

          print(test)



          print (test)

          test = w = w -learning_rate*costfunction()
```

```
print(test)
"""


full_df_weights_line = pd.DataFrame({'iteration':[i],'w1':[-0.45],'w2':[0.2]})
for i in range(1, 1000):

    w = w -ln*costfunction()
    weight_updates = pd.DataFrame({'iteration':[i],'w1':[w[0]],'w2':[w[1]]},index=[(
    full_df_weights_line = pd.concat([full_df_weights_line,weight_updates])
    i=i+1
    learning_rate = (np.dot(costfunction().T,costfunction())
                     /
                     np.dot(costfunction().T,np.dot(returnH(),costfunction())))
    ln = np.squeeze(np.asarray(learning_rate))



print(full_df_weights_line)

print('left: the link between w0 and w1 | Right: the evolution of the weights during
print('We see that FULL convergence happened AFTER 22 iterations.')

plt.figure(figsize=(40,20))

plt.subplot(1, 2, 1)
scatter1 = plt.scatter(full_df_weights_line.w1, full_df_weights_line.w2,s=60)
plt.xlabel('w0', fontsize=40)
plt.ylabel('w1', fontsize=40)
plt.suptitle('left: the link between w0 and w1 | Right: the evolution of the weights
             fontsize=40)
plt.tick_params(axis='both', which='major', labelsize=30)




plt.subplot(1, 2, 2)
plt.tick_params(axis='both', which='major', labelsize=30)
plt.scatter(full_df_weights_line.iteration[0:100],full_df_weights_line.w1[0:100],colo
plt.scatter(full_df_weights_line.iteration[0:100],full_df_weights_line.w2[0:100],  co

plt.xlabel('iterations', fontsize=40)
plt.ylabel('weights', fontsize=40)

plt.legend(fontsize=50)
```

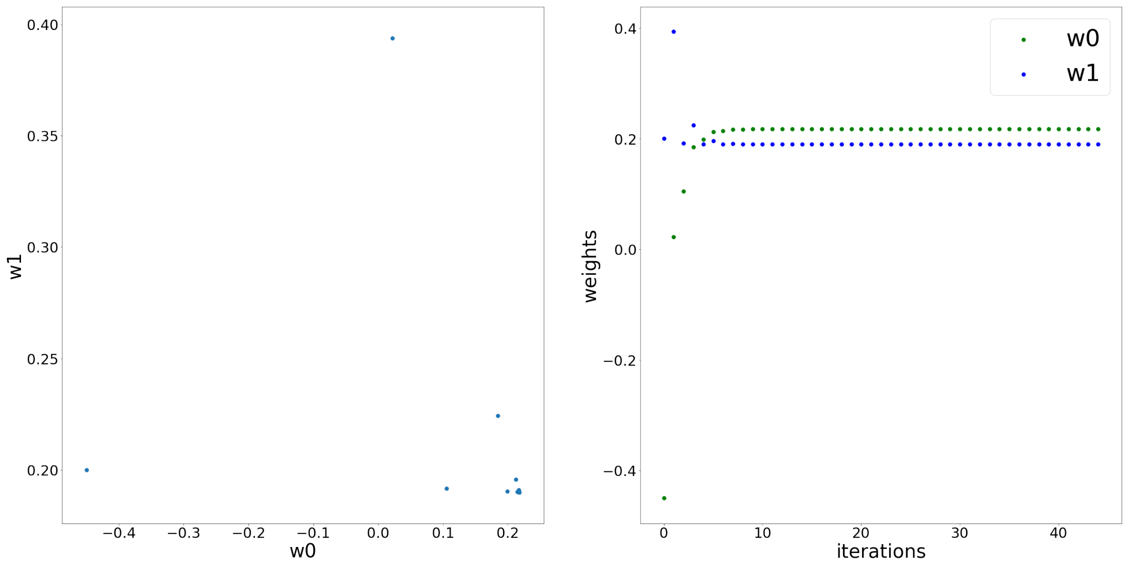
```
        plt.show()
        plt.gcf().clear()
```

```
    iteration                   w1                   w2
0           0                -0.45                  0.2
1           1  [[[[ 0.02213089]]]]  [[[[ 0.39383464]]]]
2           2  [[[[ 0.10512875]]]]  [[[[ 0.19167337]]]]
3           3  [[[[ 0.18471207]]]]  [[[[ 0.22434653]]]]
4           4  [[[[ 0.19870236]]]]  [[[[ 0.19026982]]]]
5           5  [[[[ 0.21211708]]]]  [[[[ 0.19577727]]]]
6           6  [[[[ 0.21447531]]]]  [[[[ 0.19003324]]]]
7           7  [[[[ 0.21673652]]]]  [[[[ 0.19096158]]]]
8           8  [[[[ 0.21713403]]]]  [[[[ 0.18999336]]]]
9           9  [[[[ 0.21751519]]]]  [[[[ 0.19014984]]]]
10         10  [[[[ 0.21758219]]]]  [[[[ 0.18998664]]]]
11         11  [[[[ 0.21764644]]]]  [[[[ 0.19001301]]]]
12         12  [[[[ 0.21765773]]]]   [[[[ 0.1899855]]]]
13         13  [[[[ 0.21766856]]]]  [[[[ 0.18998995]]]]
14         14  [[[[ 0.21767047]]]]  [[[[ 0.18998531]]]]
15         15  [[[[ 0.21767229]]]]  [[[[ 0.18998606]]]]
16         16  [[[[ 0.21767261]]]]  [[[[ 0.18998528]]]]
17         17  [[[[ 0.21767292]]]]  [[[[ 0.18998541]]]]
18         18  [[[[ 0.21767298]]]]  [[[[ 0.18998527]]]]
19         19  [[[[ 0.21767303]]]]  [[[[ 0.18998529]]]]
20         20  [[[[ 0.21767304]]]]  [[[[ 0.18998527]]]]
21         21  [[[[ 0.21767304]]]]  [[[[ 0.18998528]]]]
22         22  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
23         23  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
24         24  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
25         25  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
26         26  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
27         27  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
28         28  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
29         29  [[[[ 0.21767305]]]]  [[[[ 0.18998527]]]]
..        ...                  ...                  ...
970       970         [[[[ nan]]]]         [[[[ nan]]]]
971       971         [[[[ nan]]]]         [[[[ nan]]]]
972       972         [[[[ nan]]]]         [[[[ nan]]]]
973       973         [[[[ nan]]]]         [[[[ nan]]]]
974       974         [[[[ nan]]]]         [[[[ nan]]]]
975       975         [[[[ nan]]]]         [[[[ nan]]]]
976       976         [[[[ nan]]]]         [[[[ nan]]]]
977       977         [[[[ nan]]]]         [[[[ nan]]]]
978       978         [[[[ nan]]]]         [[[[ nan]]]]
979       979         [[[[ nan]]]]         [[[[ nan]]]]
980       980         [[[[ nan]]]]         [[[[ nan]]]]
981       981         [[[[ nan]]]]         [[[[ nan]]]]
982       982         [[[[ nan]]]]         [[[[ nan]]]]
```

```
983         983          [[[[[ nan]]]]]          [[[[[ nan]]]]]
984         984          [[[[[ nan]]]]]          [[[[[ nan]]]]]
985         985          [[[[[ nan]]]]]          [[[[[ nan]]]]]
986         986          [[[[[ nan]]]]]          [[[[[ nan]]]]]
987         987          [[[[[ nan]]]]]          [[[[[ nan]]]]]
988         988          [[[[[ nan]]]]]          [[[[[ nan]]]]]
989         989          [[[[[ nan]]]]]          [[[[[ nan]]]]]
990         990          [[[[[ nan]]]]]          [[[[[ nan]]]]]
991         991          [[[[[ nan]]]]]          [[[[[ nan]]]]]
992         992          [[[[[ nan]]]]]          [[[[[ nan]]]]]
993         993          [[[[[ nan]]]]]          [[[[[ nan]]]]]
994         994          [[[[[ nan]]]]]          [[[[[ nan]]]]]
995         995          [[[[[ nan]]]]]          [[[[[ nan]]]]]
996         996          [[[[[ nan]]]]]          [[[[[ nan]]]]]
997         997          [[[[[ nan]]]]]          [[[[[ nan]]]]]
998         998          [[[[[ nan]]]]]          [[[[[ nan]]]]]
999         999          [[[[[ nan]]]]]          [[[[[ nan]]]]]

[1000 rows x 3 columns]
left: the link between w0 and w1 | Right: the evolution of the weights during the iterations.
We see that FULL convergence happened AFTER 22 iterations.
```

```
<matplotlib.figure.Figure at 0x278cd6e3a20>
```

left: the link between w0 and w1 | Right: the evolution of the weights during the iterations.



```
<matplotlib.figure.Figure at 0x278cdbee550>
```

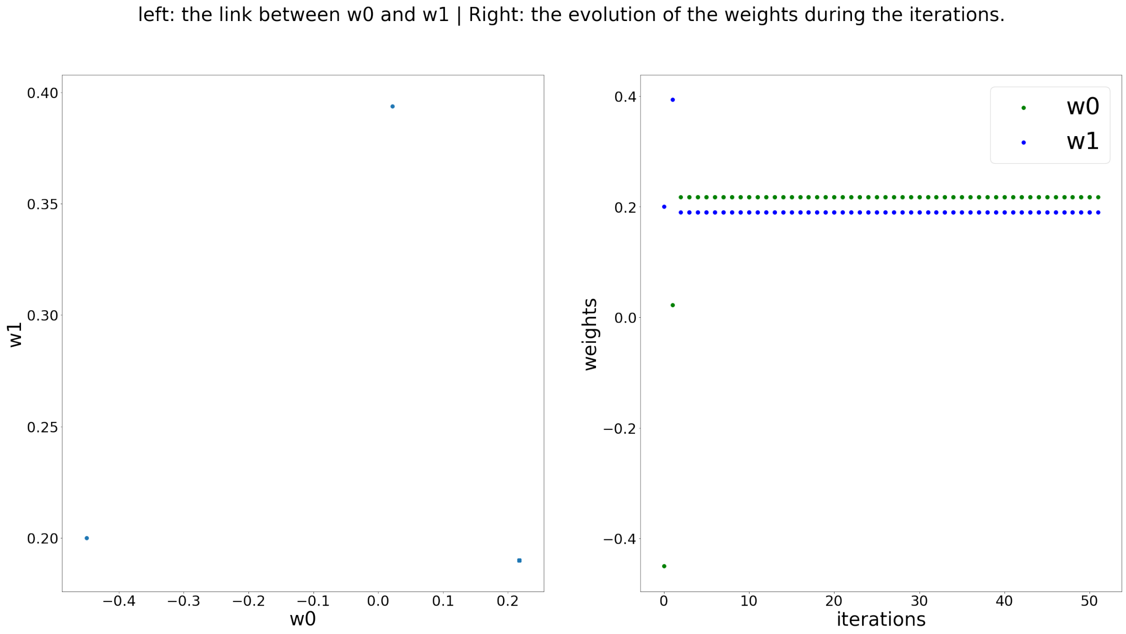
```
In [171]: #C

          i =0;
          w= np.matrix([-0.45,0.2]).T;

          full_df_weights_conj = pd.DataFrame({'iteration':[i],'w1':[-0.45],'w2':[0.2]})


          np.seterr(divide='ignore', invalid='ignore')

          w= np.matrix([-0.45,0.2]).T;
          d = -(costfunction())

          learning_rate_conjugate = -(np.dot(d.T,costfunction())
                           /
                           np.dot(d.T,np.dot(returnH(),d)))
          lnc = np.squeeze(np.asarray(learning_rate_conjugate))



          full_df_weights_conj = pd.DataFrame({'iteration':[i],'w1':[-0.45],'w2':[0.2]})
          beta_upper_half = np.dot(costfunction().T,costfunction())


          for i in range(1, 100):
              d = np.squeeze(np.asarray(d))
              oldgradient = costfunction()
              beta_lower_half = np.dot(costfunction().T,costfunction())
              dneu = np.matrix([d[0],d[1]]).T

              #print(w,dneu,lnc)
              if (beta_upper_half == 0):

                  weight_updates_conj = pd.DataFrame({'iteration':[i],'w1':[w[0]],'w2':[w[1]]}
                  full_df_weights_conj = pd.concat([full_df_weights_conj,weight_updates_conj])
              else:

                  w = w +dneu*lnc

              new_gradient =costfunction()
              beta_upper_half = np.dot(costfunction().T,costfunction())
              weight_updates_conj = pd.DataFrame({'iteration':[i],'w1':[w[0]],'w2':[w[1]]},inde
              full_df_weights_conj = pd.concat([full_df_weights_conj,weight_updates_conj])
              i=i+1

              beta = -(beta_upper_half/beta_lower_half)

              d = costfunction() + dneu*beta
```

```
            d = np.squeeze(np.asarray(d))
            dneu = np.matrix([d[0],d[1]]).T
            learning_rate_conjugate = -(np.dot(dneu.T,costfunction())/np.dot(dneu.T,np.dot(re
            lnc = np.squeeze(np.asarray(learning_rate_conjugate))


        print(full_df_weights_conj)
        print('left: the link between w0 and w1 | Right: the evolution of the weights during
        print('We see that convergence happened AFTER 2(!) iterations.')

        plt.figure(figsize=(40,20))

        plt.subplot(1, 2, 1)
        scatter1 = plt.scatter(full_df_weights_conj.w1, full_df_weights_conj.w2,s=60)
        plt.xlabel('w0', fontsize=40)
        plt.ylabel('w1', fontsize=40)
        plt.suptitle('left: the link between w0 and w1 | Right: the evolution of the weights
                     fontsize=40)
        plt.tick_params(axis='both', which='major', labelsize=30)




        plt.subplot(1, 2, 2)
        plt.tick_params(axis='both', which='major', labelsize=30)
        plt.scatter(full_df_weights_conj.iteration[0:100],full_df_weights_conj.w1[0:100],colo
        plt.scatter(full_df_weights_conj.iteration[0:100],full_df_weights_conj.w2[0:100],  co

        plt.xlabel('iterations', fontsize=40)
        plt.ylabel('weights', fontsize=40)

        plt.legend(fontsize=50)


        plt.show()
        plt.gcf().clear()
```

```
    iteration                     w1                     w2
0           0                  -0.45                    0.2
1           1  [[[[[ 0.02213089]]]]]  [[[[[ 0.39383464]]]]]
2           2  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
3           3  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
3           3  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
4           4  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
4           4  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
5           5  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
5           5  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
```

```
6           6  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
6           6  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
7           7  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
7           7  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
8           8  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
8           8  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
9           9  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
9           9  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
10         10  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
10         10  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
11         11  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
11         11  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
12         12  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
12         12  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
13         13  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
13         13  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
14         14  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
14         14  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
15         15  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
15         15  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
16         16  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
..        ...                    ...                    ...
85         85  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
85         85  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
86         86  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
86         86  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
87         87  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
87         87  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
88         88  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
88         88  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
89         89  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
89         89  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
90         90  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
90         90  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
91         91  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
91         91  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
92         92  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
92         92  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
93         93  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
93         93  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
94         94  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
94         94  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
95         95  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
95         95  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
96         96  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
96         96  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
97         97  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
97         97  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
```

```
98          98  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
98          98  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
99          99  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]
99          99  [[[[[ 0.21767305]]]]]  [[[[[ 0.18998527]]]]]

[197 rows x 3 columns]
left: the link between w0 and w1 | Right: the evolution of the weights during the iterations.
We see that convergence happened AFTER 2(!) iterations.
```



```
<matplotlib.figure.Figure at 0x278ce5bcc88>
```