admm

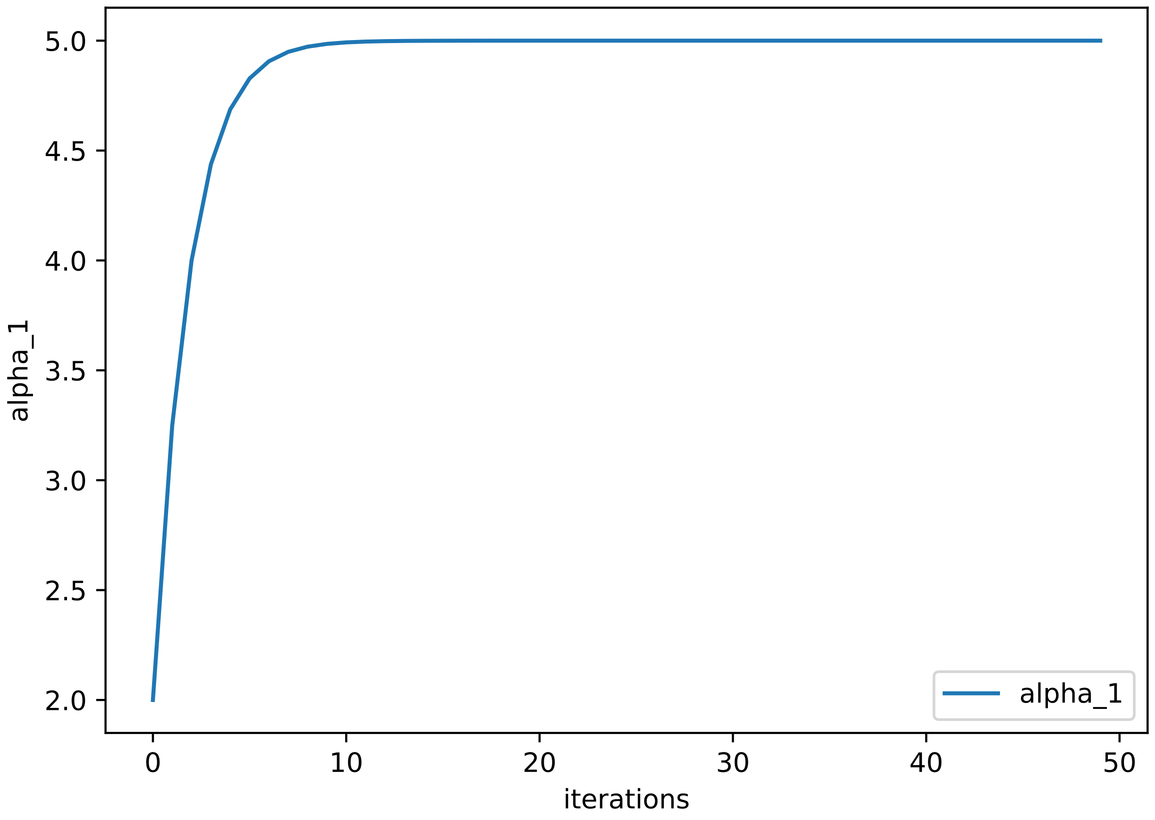
March 6, 2020

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

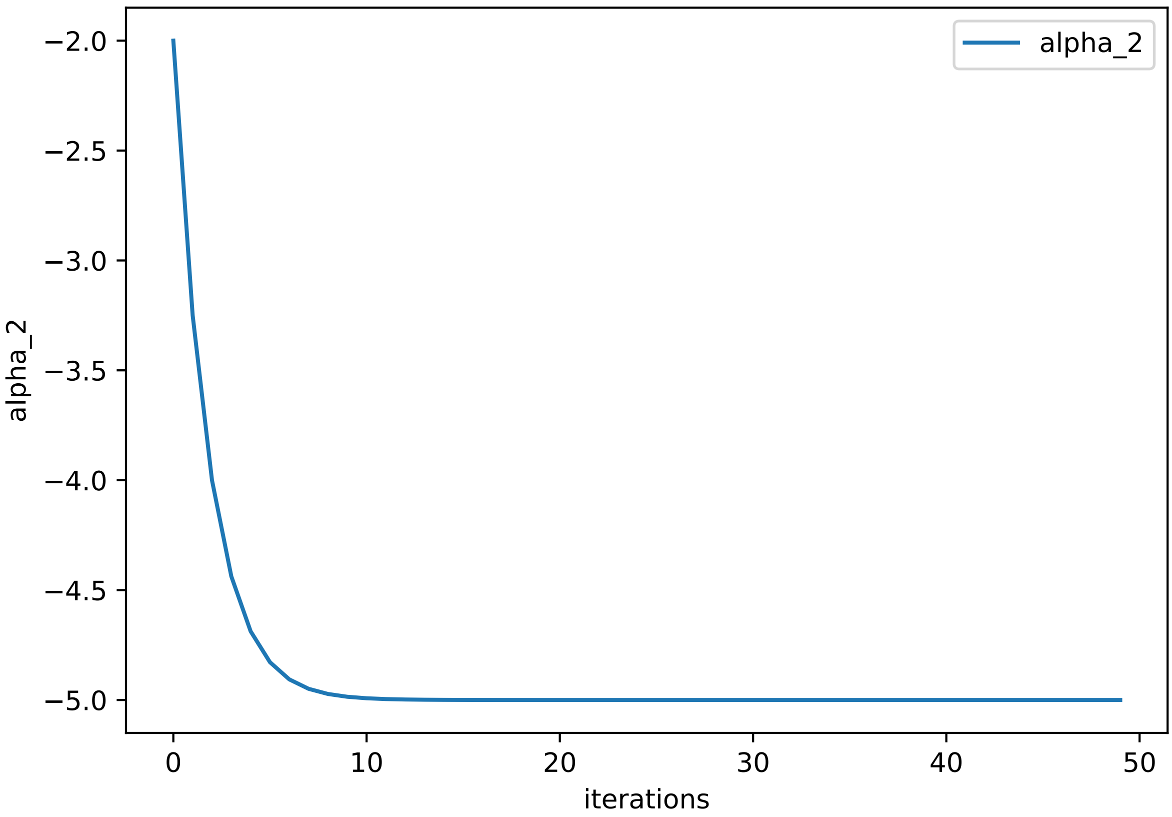
[17]:
```
def plot_props(data,prop_name, figname, xlabel):
    fig = plt.figure(figsize=(7,5))
    plt.plot(data, label=prop_name)
    plt.ylabel(prop_name)
    plt.xlabel(xlabel)
    plt.legend()
    plt.savefig("./{}.pdf".format(figname))
#     plt.show()
```

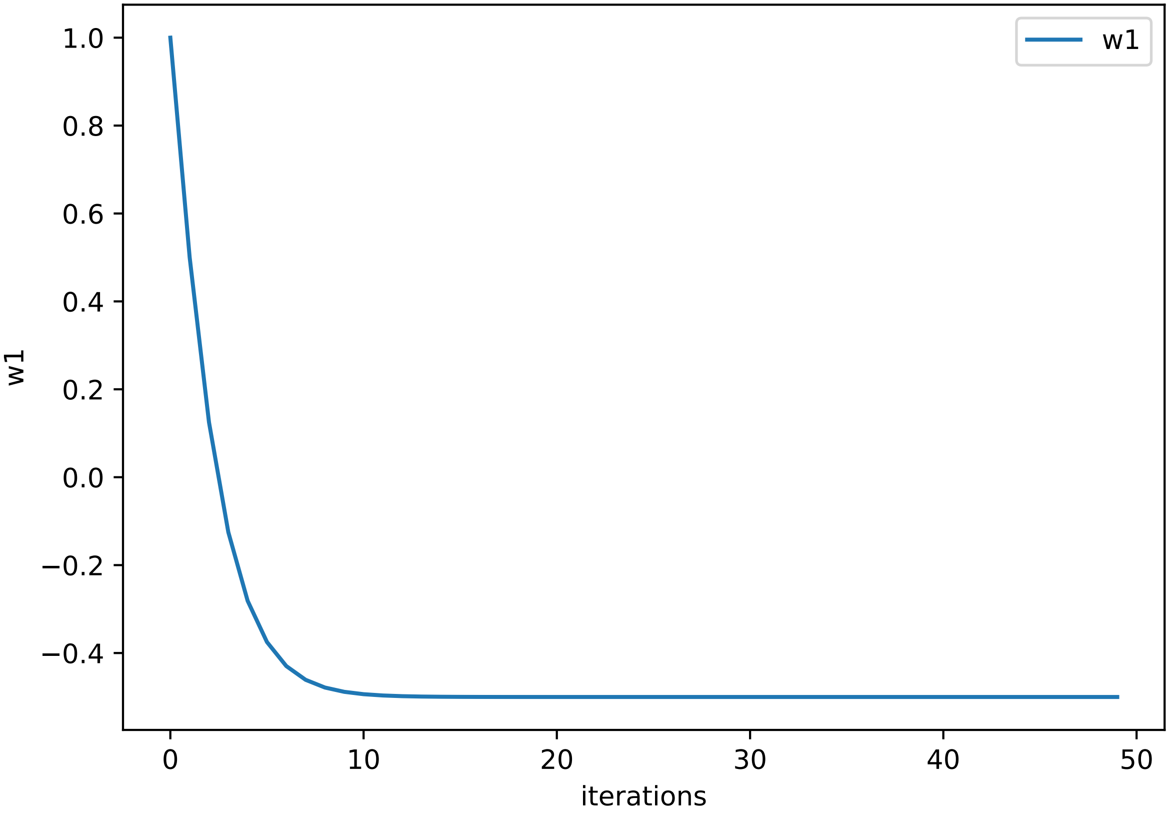
[19]:
```
def admm(lambda_, w1_init, w2_init, W_init, alpha_init):
    # learning rate
    lr = 2*lambda_
    # initialize weights
    w1, w2, W = w1_init, w2_init, W_init

    # initialize alpha
    alpha = alpha_init

    alpha1_arr = []
    alpha2_arr = []

    w1_arr = []
    w2_arr = []

    W_arr = []

    del_W = 1e5

    for i in range(50):
        # inner minimization
        w1 = (2*W +4 - alpha[0]) / 4.0
        w2 = (2*W -8 - alpha[1]) / 8.0
        # update W
        del_W = W*1.0
        W = (alpha.sum()/(4.0*lambda_)) + 0.5*(w1+w2)

        del_W = abs(del_W - W)
#         print(del_W)
        #update alpha
        alpha = alpha - lr*np.array([W-w1, W-w2])
        alpha1_arr.append(alpha[0]*1.0)
        alpha2_arr.append(alpha[1]*1.0)
        w1_arr.append(w1*1.0)
        w2_arr.append(w2*1.0)
        W_arr.append(W*1.0)

    ## plotting

    plot_props(alpha1_arr, "alpha_1", "alpha_1_vs_iters","iterations")
    plot_props(alpha2_arr, "alpha_2", "alpha_2_vs_iters","iterations")
    plot_props(w1_arr, "w1", "w1_vs_iters","iterations")
    plot_props(w2_arr, "w2", "w2_vs_iters","iterations")
    plot_props(W_arr, "W", "W_vs_iters","iterations")
```

[20]:
```
lambda_ = 1.
w1_init = 0.0
w2_init = 0.0
W_init = 0.0
alpha_init = np.zeros(shape=(2,1))

admm(lambda_, w1_init, w2_init, W_init, alpha_init)
```

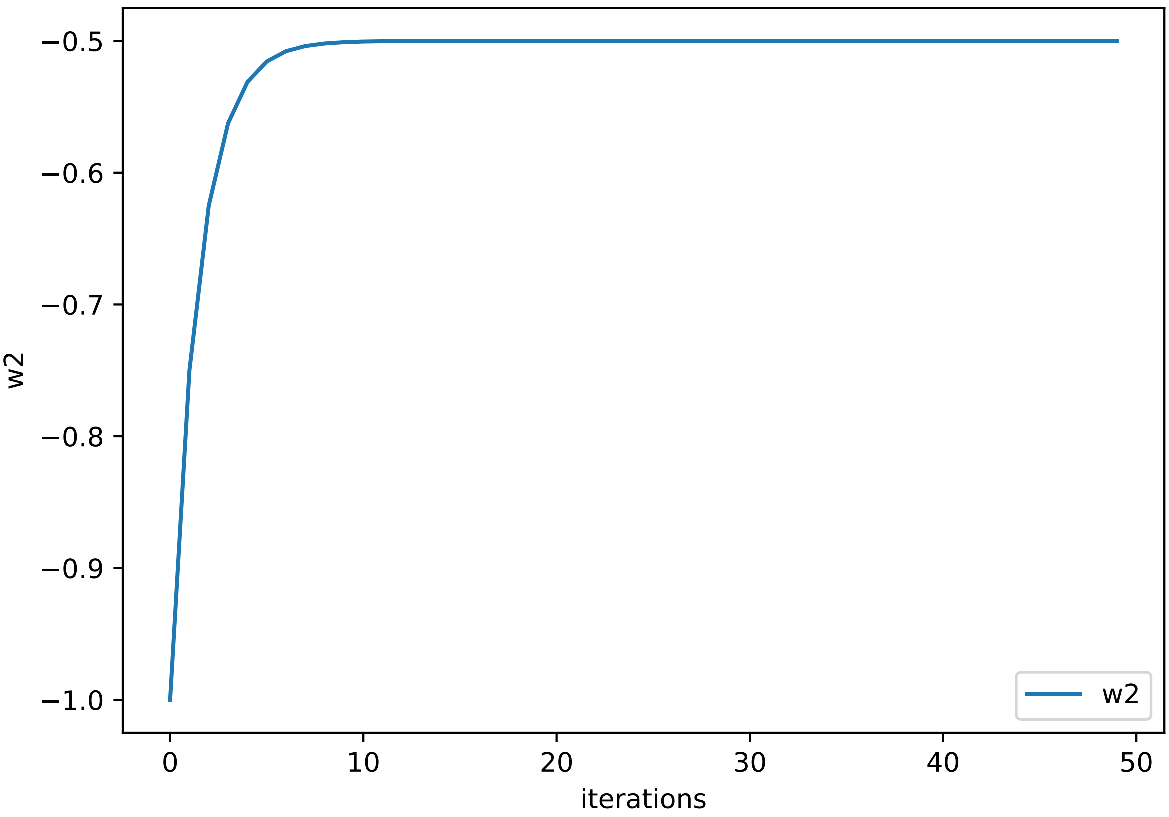


alpha_1
5.0
4.5
4.0
3.5
3.0
2.5
2.0
0
10
20
30
40
50
iterations
alpha_1

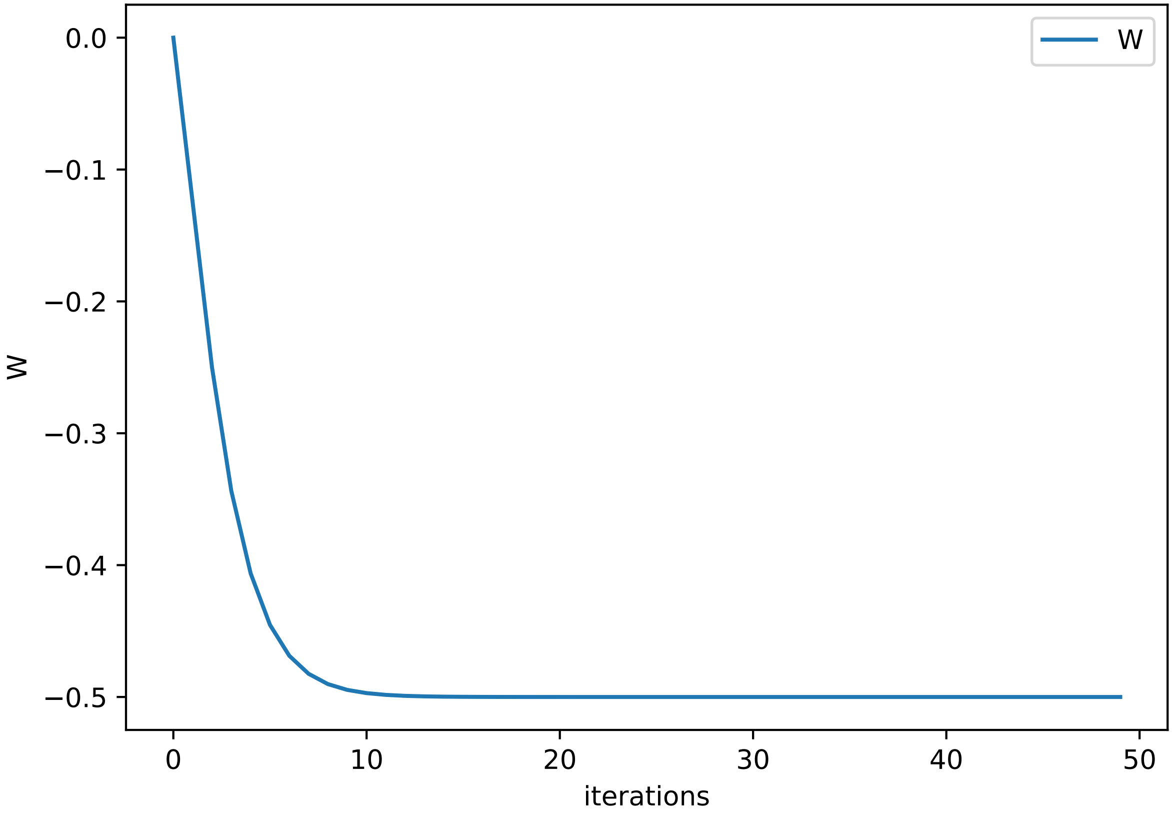

alpha_2
−2.0
−2.5
−3.0
−3.5
−4.0
−4.5
−5.0
0
10
20
30
40
50
iterations
alpha_2



w1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0
10
20
30
40
50
iterations
w1



−0.5
−0.6
−0.7
−0.8
−0.9
−1.0
w2
0
10
20
30
40
50
iterations
w2



W
0.0
−0.1
−0.2
−0.3
−0.4
−0.5
W
0
10
20
30
40
50
iterations