Activity 18

1. y_i < 1 no / 1 yes

q_i $(i = 1 \dots 100)$

8000×1

$y_i = \text{sign}\{q_i^T w\}$

because there are lots of features; some of them are unimportant. We better choose LR SO.

2. a) $[x\ 1]\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = y = 4.$ (2, 0)

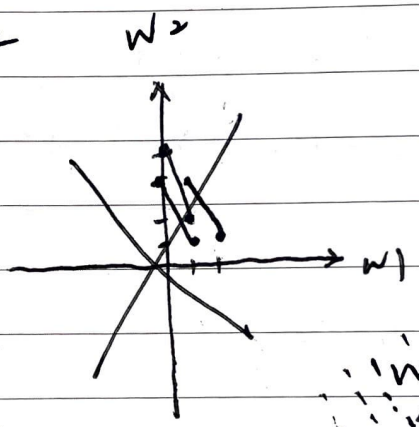
$\text{rank}(X) = 1 < \dim(w) = 2 \rightarrow$ no unique

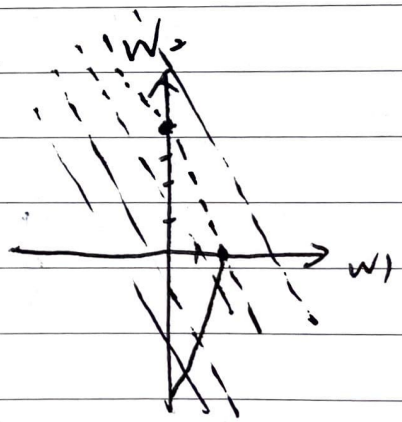
b) $f(w) = \| 4 - (xw_1 + w_2) \|^2$

① $\| 4 - (xw_1 + w_2) \|^2 = 0$

$$\begin{matrix} 4 \\ x \quad 1 \end{matrix} \quad \begin{pmatrix} xw_1 + w_2 \end{pmatrix} = \begin{matrix} 5 \\ 3 \end{matrix}$$

$$\begin{matrix} 1 & 3 \\ 1 & 1 \\ 0 & 3 \end{matrix}$$



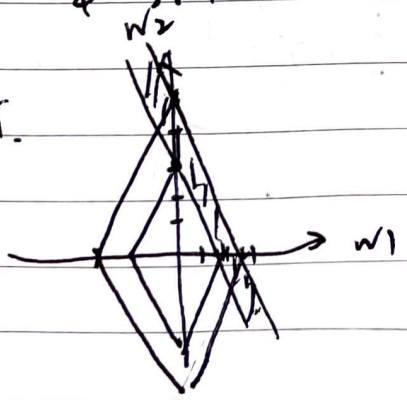


① $4 - xw_1 - w_2 = 0 \rightarrow w_2 = 4 - xw_1$

② $4 - xw_1 - w_2 \leq 1 \rightarrow 3 - xw_1 - w_2 \rightarrow$

$4 - xw_1 - w_2 \geq 1 \rightarrow 5 - xw_1 = w_2$

c) i.



ii. ~~$\|w\| + \|w_2\| = \|w_1 + w_2\|$~~ ①

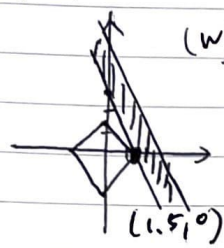
$\|w\| = w_1 + 3 - xw_1$

~~$\rightarrow \|w\| + \|3 - xw_1\| = 3 - xw_1$~~

~~$\rightarrow$ 3+ ② $xw_1 + 5 - xw_1$~~

$5 - xw_1$

ii. $\min \|w\|_1 = |w_1| + |w_2|$

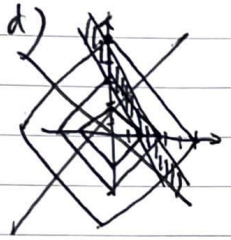


(w)R1 set $\|w\|_1 = 1.5$

in this range

iii. $3 \to w_1 = w_2$

d)



if $0 \approx \lambda$, $\|y - Xw\|_2^2$ is imporeant

if $\lambda \approx \infty$, $\lambda \|w\|_1$ is imporeant

4b) $X = [2 \times 1]$, $w_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $r = 0.1$

$z^0 = w^0 - r X^T (X w^0 - y)$

$= \begin{bmatrix} 0 \\ 0 \end{bmatrix} - 0.1 \begin{bmatrix} 2 \\ 1 \end{bmatrix} \left([2\ 1] \begin{bmatrix} 0 \\ 0 \end{bmatrix} - 4 \right)$

$= \begin{bmatrix} +0.8 \\ +0.4 \end{bmatrix}$

$w^1 = \frac{1}{1+0.4} \begin{bmatrix} +\frac{4}{5} \\ +\frac{2}{5} \end{bmatrix} = \begin{bmatrix} +\frac{4}{7} \\ +\frac{2}{7} \end{bmatrix}$

$\frac{2}{5}$ $\frac{5}{7}$

$\frac{x^T}{2} = \frac{x \cdot 0.1}{x} = 0.2$

$-0.2 < z_i < 0.2$

$z^1 = z^0 - 0.1 \begin{bmatrix} 2 \\ 1 \end{bmatrix} \left([2\ 1] \begin{bmatrix} +\frac{4}{7} \\ +\frac{2}{7} \end{bmatrix} - 4 \right)$

$\begin{bmatrix} +\frac{4}{7} \\ +\frac{2}{7} \end{bmatrix}$

$+\frac{8}{7} + \frac{2}{7} - 4$

$\frac{10}{7}$ $\frac{-18}{7}$

$w^1 = \begin{bmatrix} 0.8 - 0.2 \\ 0.4 - 0.2 \end{bmatrix} = \begin{bmatrix} 0.6 \\ 0.2 \end{bmatrix}$

$z^1 = \begin{bmatrix} 0.6 \\ 0.2 \end{bmatrix} - 0.1 \begin{bmatrix} 2 \\ 1 \end{bmatrix} \left([2\ 1] \begin{bmatrix} 0.6 \\ 0.2 \end{bmatrix} - 4 \right)$

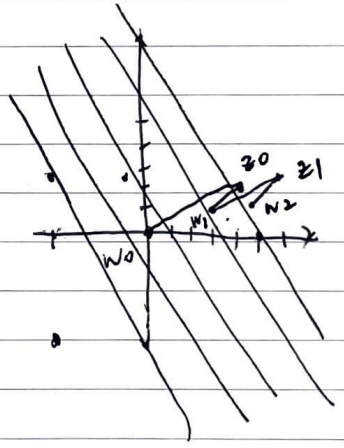
$= \begin{bmatrix} 0.6 \\ 0.2 \end{bmatrix} + 0.1 \begin{bmatrix} 2 \\ 1 \end{bmatrix} (1.2 + 0.2 - 4)$

$1.4 - 4$

$+2.6$

$= \begin{bmatrix} 0.6 \\ 0.2 \end{bmatrix} + \begin{bmatrix} 0.52 \\ 0.26 \end{bmatrix} = \begin{bmatrix} 1.12 \\ 0.46 \end{bmatrix} > 0.2$ > 0.2

$w^2 = \begin{bmatrix} 1.12 - 0.2 \\ 0.46 - 0.2 \end{bmatrix} = \begin{bmatrix} 0.92 \\ 0.26 \end{bmatrix}$



Activity 18

Setup

In [21]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [22]:
```
def prxgraddescent_l1(X,y,tau,lam,w_init,it):

## compute it iterations of L2 proximal gradient descent starting at w1
## w_{k+1}= (w_k - tau*X'*(X*w_k - y)/(1+lam*tau)
## step size tau
    W = np.zeros((w_init.shape[0], it+1))
    Z = np.zeros((w_init.shape[0], it+1))
    W[:,[0]] = w_init
    for k in range(it):
        Z[:,[k+1]] = W[:,[k]] - tau * X.T @ (X @ W[:,[k]] - y);
        W[:,[k+1]] = np.sign(Z[:,[k+1]])* np.clip(np.abs(Z[:,[k+1]])-lam*tau/2,0

    return W,Z
```

In [23]:
```
## Proximal gradient descent trajectories
## Least Squares Problem
X = np.array([[2, 1]])
y = np.array([[4]])

### Find values of f(w), the contour plot surface for
w1 = np.arange(-1,3,.1)
w2 = np.arange(-1,3,.1)
fw = np.zeros((len(w1), len(w2)))
for i in range(len(w2)):
    for j in range(len(w1)):
        w = np.array([ [w1[j]], [w2[i]] ])
        fw[i,j] = (X @ w - y)**2
```

```
C:\Users\ftstc\AppData\Local\Temp\ipykernel_11780\3461163016.py:13: DeprecationWa
rning: Conversion of an array with ndim > 0 to a scalar is deprecated, and will e
rror in future. Ensure you extract a single element from your array before perfor
ming this operation. (Deprecated NumPy 1.25.)
  fw[i,j] = (X @ w - y)**2
```

Question 3a) How many iterations does it take for the algorithm to converge to the solution?

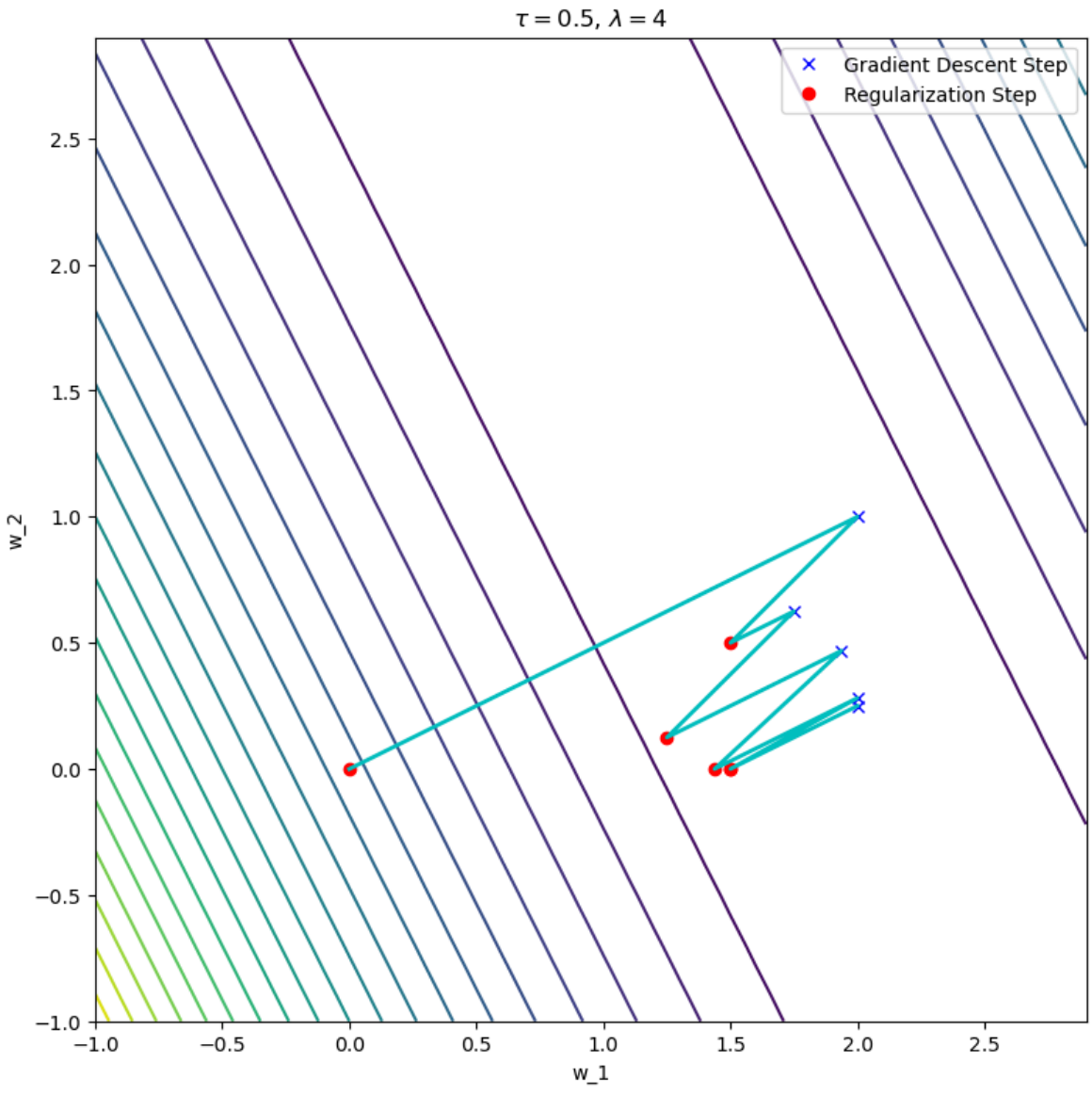
What is the converged value for w?

The iterations are 4 to converge. w is [1.5, 0.].

In [24]:
```
## Find and display weights generated by gradient descent

w_init = np.array([[0],[0]])
lam = 4;
it = 5
tau = 0.25
W,Z = prxgraddescent_l1(X,y,tau,lam,w_init,it)
# Concatenate gradient and regularization steps to display trajectory
G = np.zeros((2,0))
for i in range(it):
    G = np.hstack((G,np.hstack((W[:,[i]],Z[:,[i+1]]))))

plt.figure(figsize=(9,9))
plt.contour(w1,w2,fw,20)
plt.plot(Z[0,1::],Z[1,1:],'bx',linewidth=2, label="Gradient Descent Step")
plt.plot(W[0,:],W[1,:],'ro',linewidth=2, label="Regularization Step")
plt.plot(G[0,:],G[1,:],'-c',linewidth=2)
plt.legend()
plt.xlabel('w_1')
plt.ylabel('w_2')
plt.title('$\\tau = $'+str(.5)+', $\lambda = $'+str(lam));
```



In [25]:
```
print(W)
W[:,-1]
```

```
[[0.     1.5    1.25   1.4375 1.5    1.5   ]
 [0.     0.5    0.125  0.     0.     0.    ]]
```

Out[25]: array([1.5, 0.])

Question 3b) Change to λ = 2. How many iterations does it take for the algorithm to converge

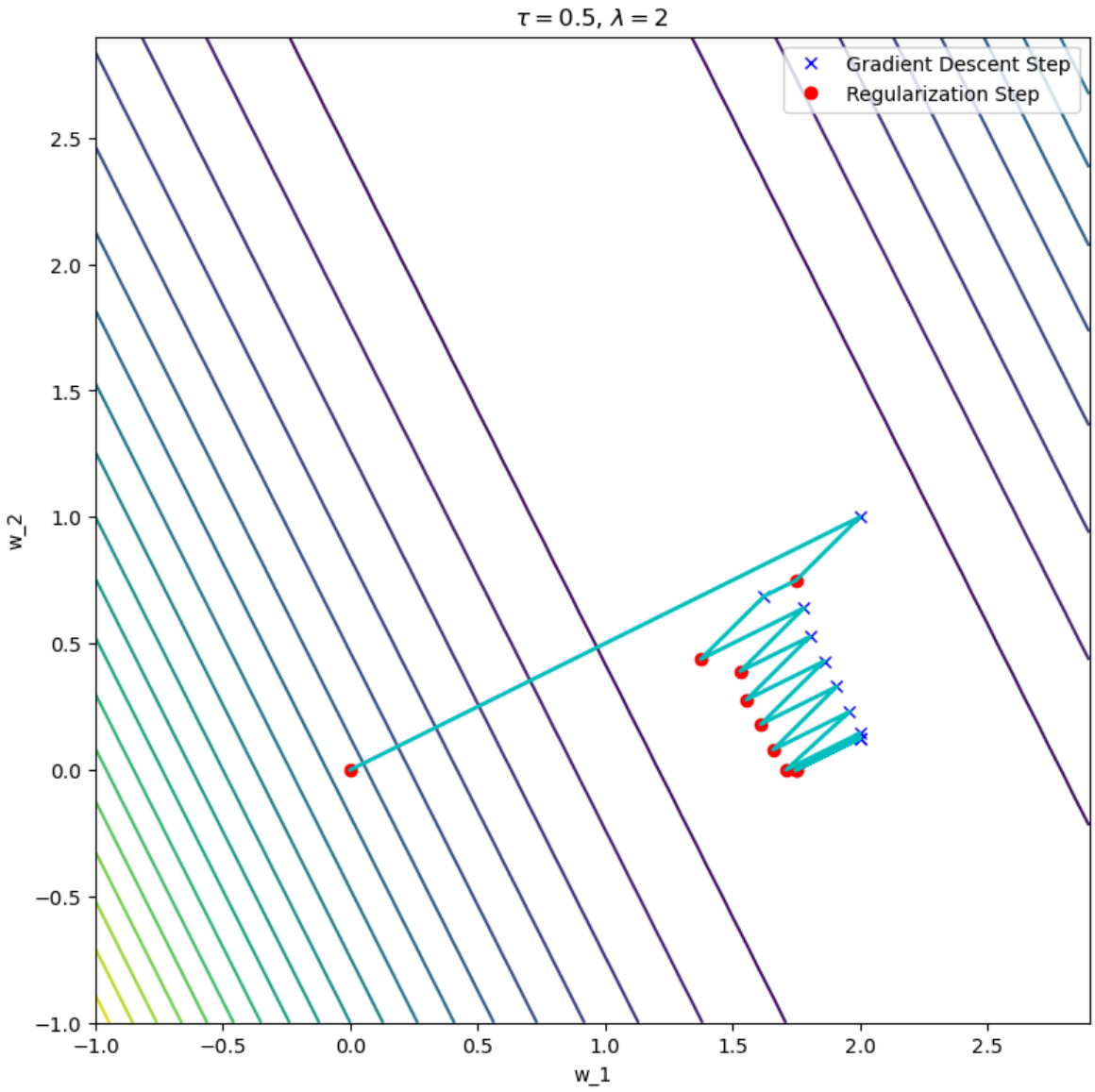
to the solution? What is the converged value for w?

The iterations are 8 to converge. w is [1.75, 0.].

In [26]:
```
## Find and display weights generated by gradient descent

w_init = np.array([[0],[0]])
lam = 2;
it = 10
tau = 0.25
W,Z = prxgraddescent_l1(X,y,tau,lam,w_init,it)
# Concatenate gradient and regularization steps to display trajectory
G = np.zeros((2,0))
for i in range(it):
    G = np.hstack((G,np.hstack((W[:,[i]],Z[:,[i+1]]))))

plt.figure(figsize=(9,9))
plt.contour(w1,w2,fw,20)
plt.plot(Z[0,1::],Z[1,1:],'bx',linewidth=2, label="Gradient Descent Step")
plt.plot(W[0,:],W[1,:],'ro',linewidth=2, label="Regularization Step")
plt.plot(G[0,:],G[1,:],'-c',linewidth=2)
plt.legend()
plt.xlabel('w_1')
plt.ylabel('w_2')
plt.title('$\\tau = $'+str(.5)+', $\lambda = $'+str(lam));
```



In [27]:
```
print(W)
W[:,-1]
```

```
[[0.         1.75       1.375      1.53125    1.5546875  1.61132812
  1.65966797 1.71008301 1.75       1.75       1.75       ]
 [0.         0.75       0.4375     0.390625   0.27734375 0.18066406
  0.07983398 0.         0.         0.         0.         ]]
```

Out[27]:
```
array([1.75, 0.  ])
```

Question 3c) Explain what happens to the weights in the regularization step.

When the lamda becomes smaller, the weights become bigger.

Question 4a) What is the maximum value for the step size in the negative gradient direction, τ?

τ is 1/5.

In [31]:
```
U, s, VT = np.linalg.svd(X, full_matrices=True)
print(s)
s2 = s**(2)
print(s2)
```

```
[2.23606798]
[5.]
```

In [ ]:
```
9
```

In [ ]:

In [ ]:

In [ ]: