EE475 Homework 1

Michael Rencheck

April 17, 2020

Problem 3.1: Find the stationary points and plot the functions.

a:

$$g(w) = w\log(w) + (1-w)\log(1-w)$$
$$g'(w) = \log(w) + w\frac{1}{w\ln(10)} - \log(1-w) - (1-w)\frac{1}{(1-w)\ln(10)}$$
$$g'(w) = \log(w) - \log(1-w)$$

Stationary point @ g' = 0:

$$0 = \log(w) - \log(1-w)$$
$$w = \frac{1}{2}$$

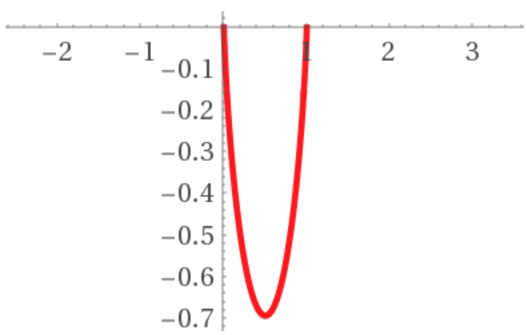


Figure 1: Minimum at $w = 0.5$

b:

$$g(w) = \log(1+e^w)$$
$$g'(w) = \frac{1}{(1+e^w)\ln(10)}e^w\log(e)$$
$$g'(w) = \frac{e^w}{(1+e^w)}$$

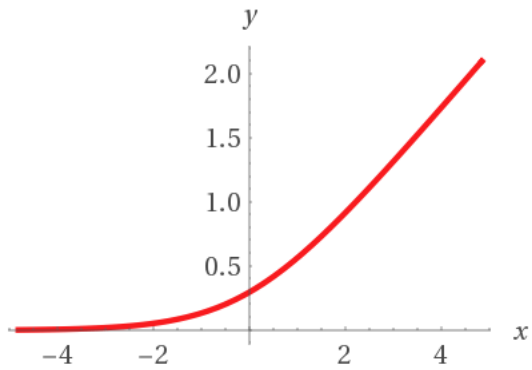


Figure 2: Saddle Point at $w = -\infty$

Stationary point @ g' = 0:

$$0 = \frac{e^w}{(1+e^w)}$$
$$0 = e^w$$
$$w = \ln(0) = -\infty$$

c:

$$g(w) = w\tanh(w)$$
$$g'(w) = \tanh(w) + w\frac{1}{\cosh^2(w)}$$

Stationary point @ g' = 0:

$$0 = \tanh(w) + w\frac{1}{\cosh^2(w)}$$
$$-w = \sinh(w)\cosh(w)$$
$$w = 0$$

d:

$$g(w) = \frac{1}{2}w^TCw + b^Tw$$
$$g'(w) = b^T + w^TC$$

Stationary point @ g' = 0:

$$\begin{bmatrix}0 & 0\end{bmatrix} = \begin{bmatrix}1 & 1\end{bmatrix} + \begin{bmatrix}w_1 & w_2\end{bmatrix}\begin{bmatrix}2 & 1\\ 1 & 3\end{bmatrix}$$

$$\begin{bmatrix}w_1\\ w_2\end{bmatrix} = \begin{bmatrix}-0.4\\ -0.2\end{bmatrix}$$

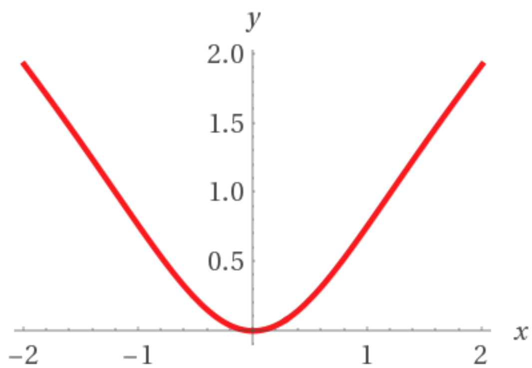


Figure 3: Minimum at $w = 0$

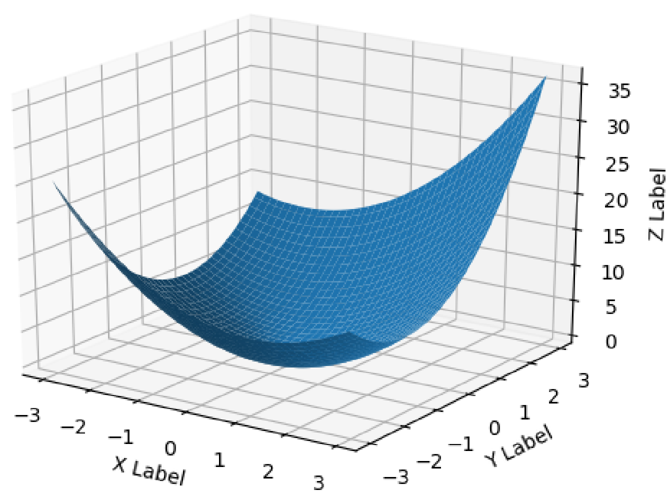


Figure 4: Minimum at $x = -0.4$ and $y = -0.2$

Problem 3.3

$g(w) = \frac{w^T C w}{w^T w}$

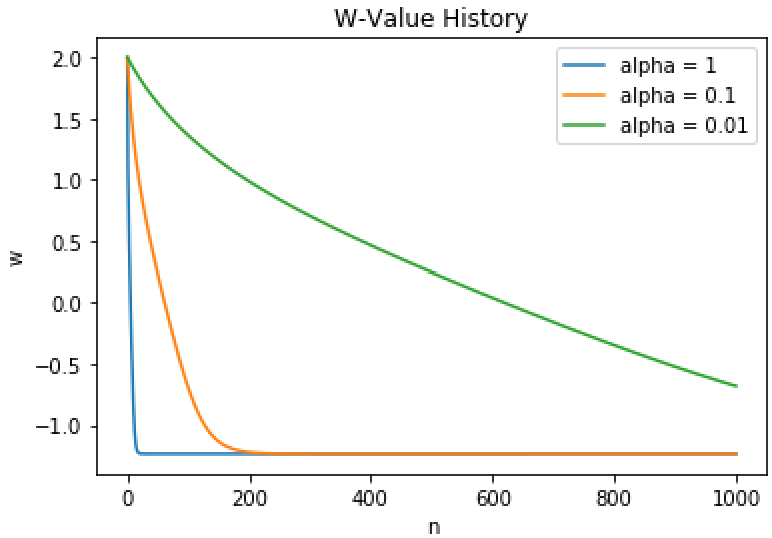
$Dg(w) = \frac{2w^T C w^T w - w^T C w (2w^T)}{(w^T w)^2}$

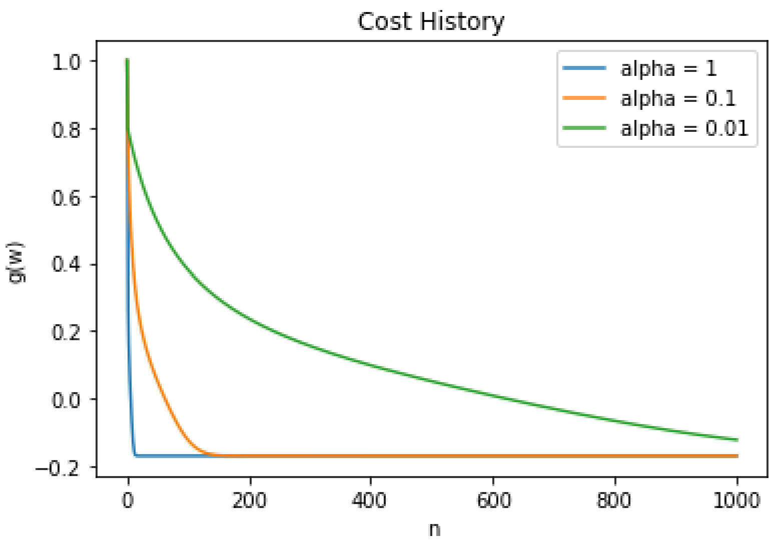
Stationary point @ $Dg(w) = \mathbf{0}$:

$\mathbf{0} = 2w^T C w^T w - w^T C w (2w^T)$

Problem 3.5

In [3]:
```
import numpy as np
import matplotlib.pyplot as plt

def f(w):
    return (1./50.)*(w**4 + w**2 + 10*w)

def df(w):
    return (1./50.)*(4.0*w**3 + 2*w + 10)



w_init = 2
n= 1000

alpha = [1, 0.1, 0.01]
w = np.ones([len(alpha), n+1])
output_w = np.copy(w)

for i, a in enumerate(alpha):
    w[i][0] = w_init
    for j in range(n):
        w[i][j+1] = w[i][j] - a * df(w[i][j])

        output_w[i][j+1] = f(w[i][j])

x = np.arange(n+1)

plt.figure()
plt.plot(x,w[0])
plt.plot(x,w[1])
plt.plot(x,w[2])

plt.legend(['alpha = 1', 'alpha = 0.1', 'alpha = 0.01'])
plt.xlabel("n")
plt.ylabel("w")
plt.title("W-Value History")

plt.figure()
plt.plot(x,output_w[0])
plt.plot(x,output_w[1])
plt.plot(x,output_w[2])

plt.title("Cost History")
plt.legend(['alpha = 1', 'alpha = 0.1', 'alpha = 0.01'])
plt.xlabel("n")
plt.ylabel("g(w)")

plt.show()
```

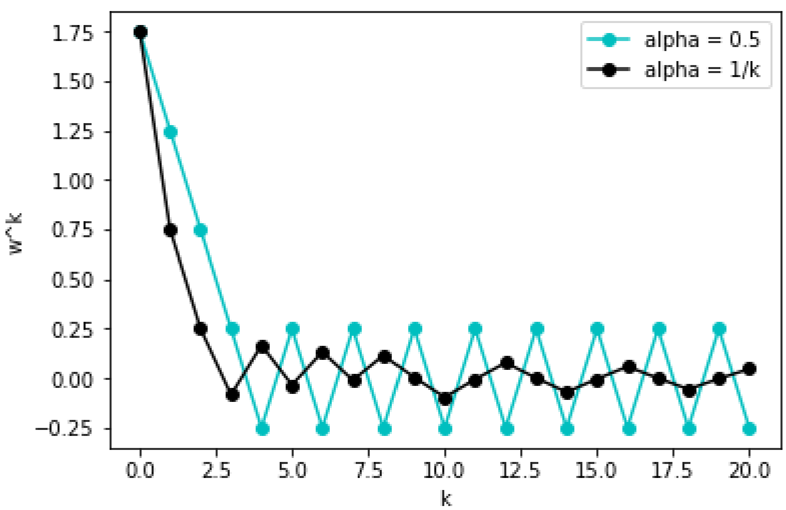




For this particular function and initial condition, $\alpha = 1$ converged the quickest.

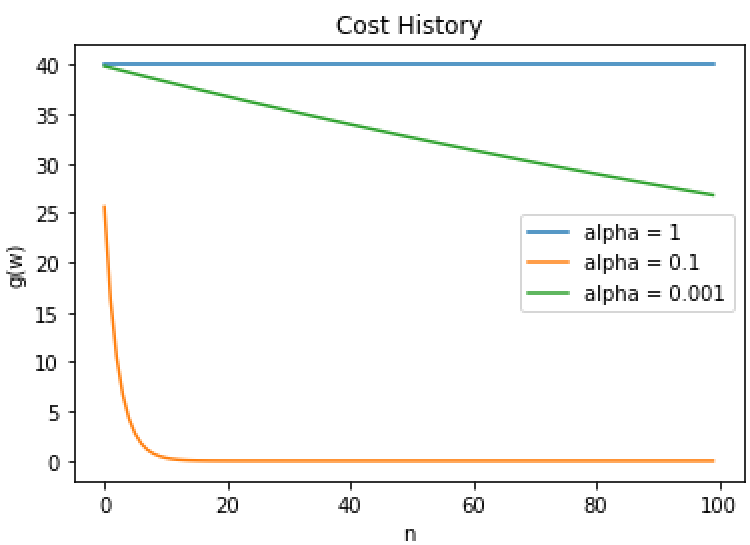
Problem 3.6

In [5]:
```
def g(w):
    return abs(w)

def dg(w):
    if(w > 0):
        return 1
    elif(w < 0):
        return -1

w_init = 1.75
n = 20

w = np.ones([2, n+1])*w_init

for i in range(n):

    try:
        w[0][i+1] = w[0][i] - 0.5 * dg(w[0][i])
    except:
        w[0][i+1] = w[0][i]
    try:
        w[1][i+1] = w[1][i] - 1/(i+1) * dg(w[1][i])
    except:
        print("error")


x = np.arange(n+1)

plt.plot(x,w[0],'co-')
plt.plot(x,w[1],'ko-')

plt.legend(['alpha = 0.5', 'alpha = 1/k'])
plt.xlabel("k")
plt.ylabel("w^k")

plt.show()
```



Problem 3.8

In [6]:
```
def h(w):
    w = np.reshape(w, [10,1])
    return np.dot(np.transpose(w), w)

def dh(w):
    return (2 * np.transpose(w))


w_init = 2
n= 100

alpha = [1, 0.1, 0.001]
h_out = np.zeros([len(alpha),n])

for j, a in enumerate(alpha):
    w = np.ones([10,n+1])*w_init
    for i in range(n):
        w[:,i+1] = w[:,i] - a * dh(w[:,i])
        h_out[j][i] = h(w[:,i+1])

x = np.arange(n)
plt.plot(x,h_out[0])
plt.plot(x,h_out[1])
plt.plot(x,h_out[2])

plt.title("Cost History")
plt.legend(['alpha = 1', 'alpha = 0.1', 'alpha = 0.001'])
plt.xlabel("n")
plt.ylabel("g(w)")
plt.show()
```



$\alpha = 0.1$ performs the best