```
U-Dec 9, LuL
 1 import numpy as np
 2 import scipy.io.wavfile
   import os
   from numpy import linalg as LA
   def update_W(W, x, learning rate)
                                    prd.edu - Dec 9, 2021, 12:34:36 PM PST
       Perform a gradient ascent update on W using data element x and the provided learning rate.
 8
10
        This function should return the updated W.
11
12
13
           W: The W matrix for ICA
14
           x: A single data element
15
           learning_rate: The learning rate to use
16
17
       Returns:
18
           The updated W
        11 11 11
19
20
       # *** START CODE HERE ***
21
22
       laplace, logistic = True, False
23
24
       if logistic:
           g = 1./(1 + np.exp(-W.dot(x)))
25
                                  tell@stanford.edu - Dec 9, 2021, 12:34:36 PN
the split r
           updated_W = W + learning_rate * (np.outer(1 - 2*g, x) + np.linalg.inv(W.T))
26
27
28
       if laplace:
           g = np.sign(W.dot(x))
29
30
           updated_W = W + learning_rate * (-np.outer(g, x) + np.linalg.inv(W.T))
31
32
       # *** END CODE HERE ***
33
34
       return updated W
35
36
37 def unmix(X, W):
38
39
       Unmix an X matrix according to W using ICA.
40
41
       Args:
42
           X: The data matrix
           W: The W for ICA
43
44
45
       Returns:
           A numpy array S containing the split data
46
                                         vgPatel1@stanford.edu-Dec9,2021,12
47
48
49
       S = np.zeros(X.shape)
50
             START CODE HERE ***
51
52
         = X.dot(W.T)
53
             END CODE HERE ***
54
55
       return S
56
57
   Fs = 11025
59
   def normalize(dat):
61
       return 0.99 * dat / np.max(np.abs(dat))
62
   def load_data():
63
       mix = np.loadtxt('./mix.dat')
64
65
       return mix
66
   def save_W(W):
       np.savetxt('./W.txt\,W)
68
                                                                                          adii-Dec 9
69
   def save sound(audio, name):
       scipy.io.wavfile.write('./{}.wav'.format(name), Fs, audio)
71
72
```

```
Dec 9, Lul
 73 def unmixer(X):
        M, N = X.shape
 74
        W = np.eye(N)
 75
 76
 77
        anneal = [0.1, 0.1, 0.1, 0.05, 0.05, 0.05, 0.02, 0.02, 0.01, 0.01, 0.005, 0.005, 0.002, 0.002, 0.001, 0.001
        print('Separating tracks
 78
        for lr in anneal:
 79
                                                                         12:34:36 PM PST
 80
           print(lr)
 81
           rand = np.random.permutation(range(M))
 82
           for i in rand:
 83
               x = X[i]
84
               W = update_W(W, x, lr)
85
 86
        return W
 87
 88
    def main():
                                       rd.edu - Dec 91
 89
        # Seed the randomness of the simulation so this outputs the same thing each time
        np.random.seed(0)
 90
 91
        X = normalize(load_data())
 92
 93
        print(X.shape)
 94
 95
        for i in range(X.shape[1]):
           save_sound(X[:, i], 'mixed_{}'.format(i))
 96
                        vgPatel1@stanford.edu-Dec9,2021,12:34:36 PN
 97
 98
        W = unmixer(X)
 99
        print(W)
100
        save_W(W)
        S = normalize(unmix(X, W))
101
        assert S.shape[1] == 5
102
        for i in range(S.shape[1]):
103
104
           if os.path.exists('split_{}'.format(i)):
105
               os.unlink('split_{}'.format(i))
           save_sound(S[:, i], 'split_{}'.format(i))
106
107
108 if __name__ == '__main__':
109
        main()
```

vgPatel1@stanford.edu-Dec9,2021,12

12:34:36 PM PST

4:36 PM PST

odii-Dec9