In [ ]:
```
import matplotlib.pyplot as plt
import numpy as np
from os import listdir
from scipy.stats.contingency import margins

font_size = 30
font_size_ticks = 25

np.set_printoptions(precision=5)

def process_name(x):
    x = x[14:]
    x = x[:-7]
    x = x.translate(str.maketrans('','','_1234567890'))
    return x[2:] + ", " + x[:2]

file_path = "./data/weather/hourly_precipitation_2015.npy"

# data_matrix contains precipitation data from 134 stations (each station is a c
data_matrix = np.load(file_path)

print(data_matrix.shape)
print(data_matrix[25:35,:10])
```

```
(8760, 134)
[[0.  1.  0.3 0.7 0.  0.2 1.  0.2 0.  0.4]
 [0.  0.4 0.2 0.3 0.  0.6 1.9 1.1 0.  0. ]
 [0.  0.6 0.  0.  0.  0.2 0.7 1.1 0.  0. ]
 [0.  0.2 0.  0.  0.  0.  0.4 0.2 0.  0. ]
 [0.  0.  0.  0.  0.  0.  0.7 0.  0.  0. ]
 [0.  0.  0.6 0.2 0.  0.  0.5 0.2 0.  0.7]
 [0.  0.  0.9 0.9 0.  0.  1.3 0.  0.  0.6]
 [0.  0.3 0.6 1.2 0.  0.4 1.9 0.  0.  0.2]
 [0.  0.  0.3 0.  0.  0.6 0.6 0.4 0.  0.6]
 [0.  0.  0.5 0.  0.  0.2 0.5 0.6 0.  0.3]]
```

In [ ]:
```
# We select three stations to study, stations contains the corresponding indices
# 24: Bodega, CA
# 99: Coos Bay, OR
# 102: Riley, OR
stations = [24,99,102]
```

In [ ]:
```
# Compute joint pmf of three Bernoulli random variables indicating whether it ra
# in Bodega, Coos Bay and Riley
def compute_joint_pmf(station_1,station_2,station_3,data_matrix):
    # INSERT YOUR CODE HERE
    '''
    data_matrix[data_matrix>0] = 1
    station_data = (data_matrix[[station_1, station_2, station_3]] > 0).astype(i

    combinations, counts = np.unique(station_data, return_counts = True, axis =

    counts = np.zeros((2,2,2))
    total = 0
    for hour in station_data.transpose():
        counts[hour[0], hour[1], hour[2]] += 1
        total += 1
```

```
    probabilities = counts / total
    '''

    data_matrix[data_matrix>0] = 1
    data_matrix[data_matrix<=0] = 0
    station1_data = data_matrix[:,stations[0]]
    station2_data = data_matrix[:,stations[1]]
    station3_data = data_matrix[:,stations[2]]

    counts = np.zeros((2,2,2))

    for i in range(0,len(station1_data)):
        val1 = int(station1_data[i])
        val2 = int(station2_data[i])
        val3 = int(station3_data[i])

        counts[val1, val2, val3] += 1

    probabilities = np.divide(counts, len(station1_data))



    return counts, probabilities

counts,joint_pmf = compute_joint_pmf(stations[0],stations[1],stations[2],data_ma

print(counts)
print(joint_pmf)
```

```
[[[7472.  186.]
  [ 814.   89.]]

 [[  88.   21.]
  [  75.   15.]]]
[[[0.85297 0.02123]
  [0.09292 0.01016]]

 [[0.01005 0.0024 ]
  [0.00856 0.00171]]]
```

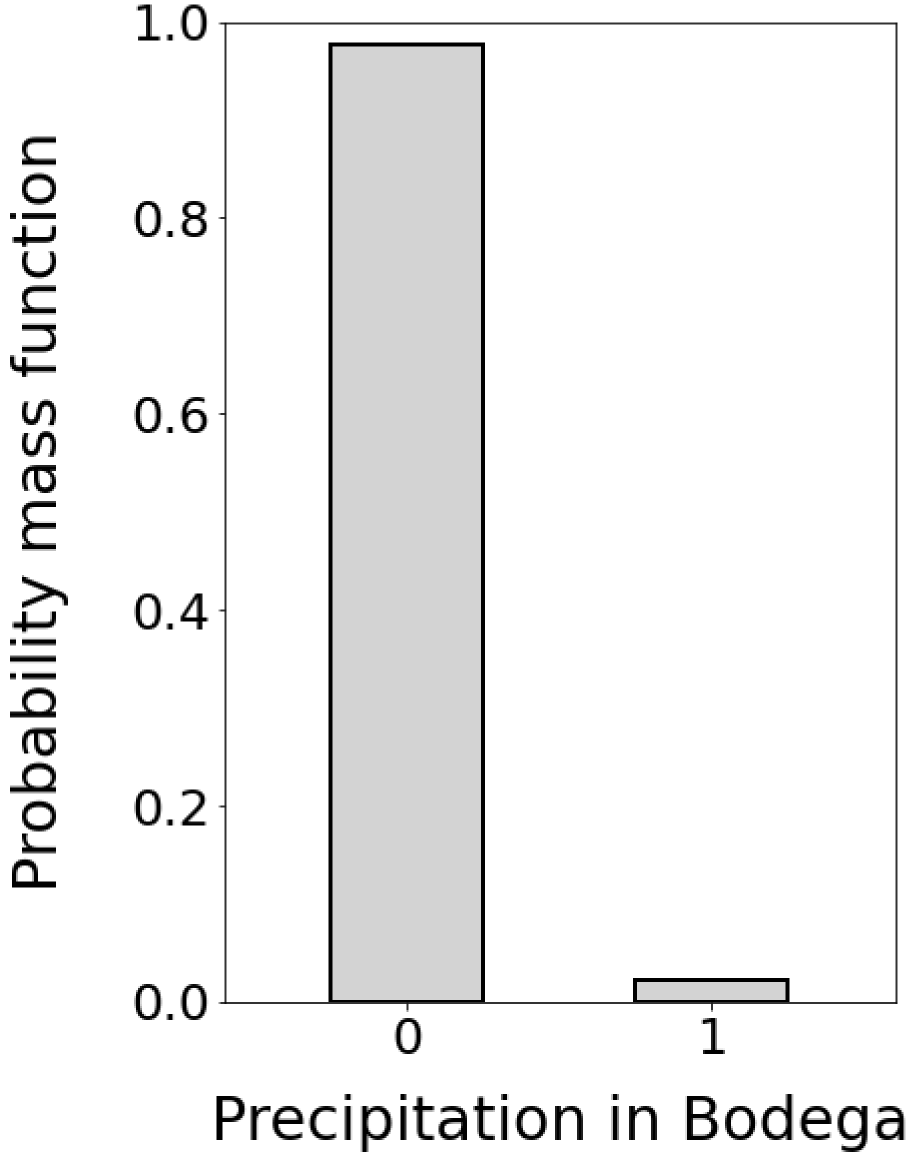
In [ ]:

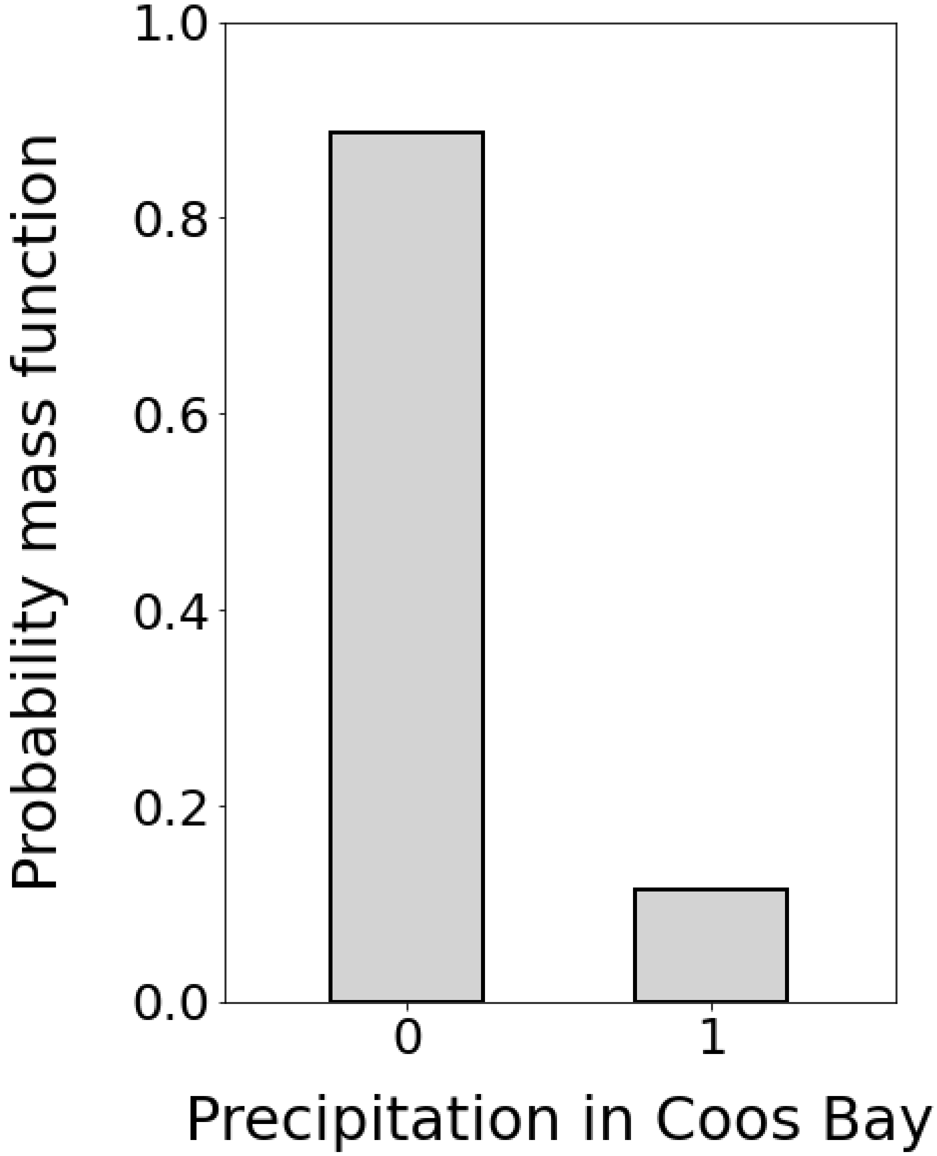
```
# Compute marginal pmf of each of the Bernoulli random variables
def marginal_1_station(joint_pmf):
    # INSERT YOUR CODE HERE
    m0, m1, m2 = margins(joint_pmf)
    return m0.flatten(), m1.flatten(), m2.flatten()

marginal_pmf_1,marginal_pmf_2,marginal_pmf_3 = marginal_1_station(joint_pmf)
print(marginal_pmf_1)
print(marginal_pmf_2)
print(marginal_pmf_3)

vals = [0,1]
ymax = 1.0
xmin = -0.6
xmax = 1.6
plt.figure(figsize=(6,9))
plt.bar(vals,marginal_pmf_1, width = 0.5, color = "lightgray", edgecolor="black"
plt.xticks(np.arange(0, 1+1, 1))
plt.xticks(fontsize=font_size_ticks)
plt.yticks(fontsize=font_size_ticks)
plt.ylim([0,ymax])
```

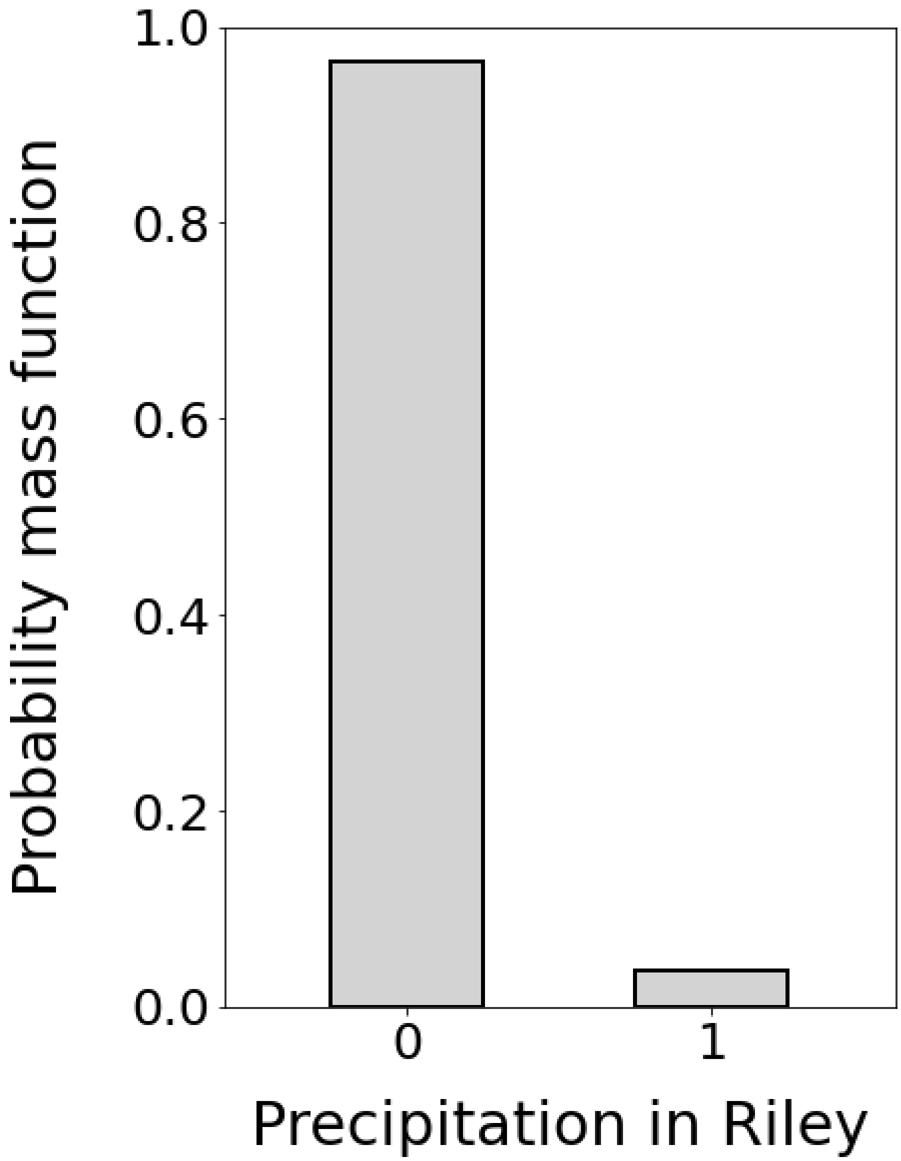
```
plt.xlim([xmin,xmax])
plt.ylabel("Probability mass function",fontsize=font_size,labelpad = 30)
plt.xlabel("Precipitation in Bodega",fontsize=font_size,labelpad = 15)
# plt.savefig('plots/precipitation_marginal_pmf_1.pdf',bbox_inches="tight")

plt.figure(figsize=(6,9))
plt.bar(vals,marginal_pmf_2, width = 0.5, color = "lightgray", edgecolor="black"
plt.xticks(np.arange(0, 1+1, 1))
plt.xticks(fontsize=font_size_ticks)
plt.yticks(fontsize=font_size_ticks)
plt.ylim([0,ymax])
plt.xlim([xmin,xmax])
plt.ylabel("Probability mass function",fontsize=font_size,labelpad = 30)
plt.xlabel("Precipitation in Coos Bay",fontsize=font_size,labelpad = 15)
# plt.savefig('plots/precipitation_marginal_pmf_2.pdf',bbox_inches="tight")

plt.figure(figsize=(6,9))
plt.bar(vals,marginal_pmf_3, width = 0.5, color = "lightgray", edgecolor="black"
plt.xticks(np.arange(0, 1+1, 1))
plt.xticks(fontsize=font_size_ticks)
plt.yticks(fontsize=font_size_ticks)
plt.ylim([0,ymax])
plt.xlim([xmin,xmax])
plt.ylabel("Probability mass function",fontsize=font_size,labelpad = 30)
plt.xlabel("Precipitation in Riley",fontsize=font_size,labelpad = 15)
# plt.savefig('plots/precipitation_marginal_pmf_3.pdf',bbox_inches="tight")
```

```
[0.97728 0.02272]
[0.88664 0.11336]
[0.9645 0.0355]
```

Out[ ]:
```
Text(0.5, 0, 'Precipitation in Riley')
```


1.0
0.8
0.6
0.4
0.2
0.0
Probability mass function
0
1
Precipitation in Bodega


1.0
0.8
0.6
0.4
0.2
0.0
Probability mass function
0
1
Precipitation in Coos Bay



In [ ]:

```
# Compute marginal joint pmf of each pair of the Bernoulli random variables
def marginal_2_stations(joint_pmf):
    # INSERT YOUR CODE HERE
    ## Station 1 and Station 2
    station12 =  [[np.sum(joint_pmf[0,0,:]), np.sum(joint_pmf[0,1,:])], [np.sum(

    ## Station 1 and Station 3
    station13 =  [[np.sum(joint_pmf[0,:,0]), np.sum(joint_pmf[0,:,1])], [np.sum(

    ## Station 2 and Station 3
    station23 =  [[np.sum(joint_pmf[:,0,0]), np.sum(joint_pmf[:,0,1])], [np.sum(

    return station12, station13, station23


marginal_pmf_12,marginal_pmf_13,marginal_pmf_23 = marginal_2_stations(joint_pmf)

print(marginal_pmf_12)
print(marginal_pmf_13)
print(marginal_pmf_23)
```

```
[[0.8742009132420091, 0.10308219178082191], [0.012442922374429224, 0.01027397260
```

```
2739725]]
[[0.9458904109589041, 0.03139269406392694], [0.018607305936073057, 0.00410958904
1095891]]
[[0.863013698630137, 0.02363013698630137], [0.10148401826484019, 0.0118721461187
21462]]
```

In [ ]:

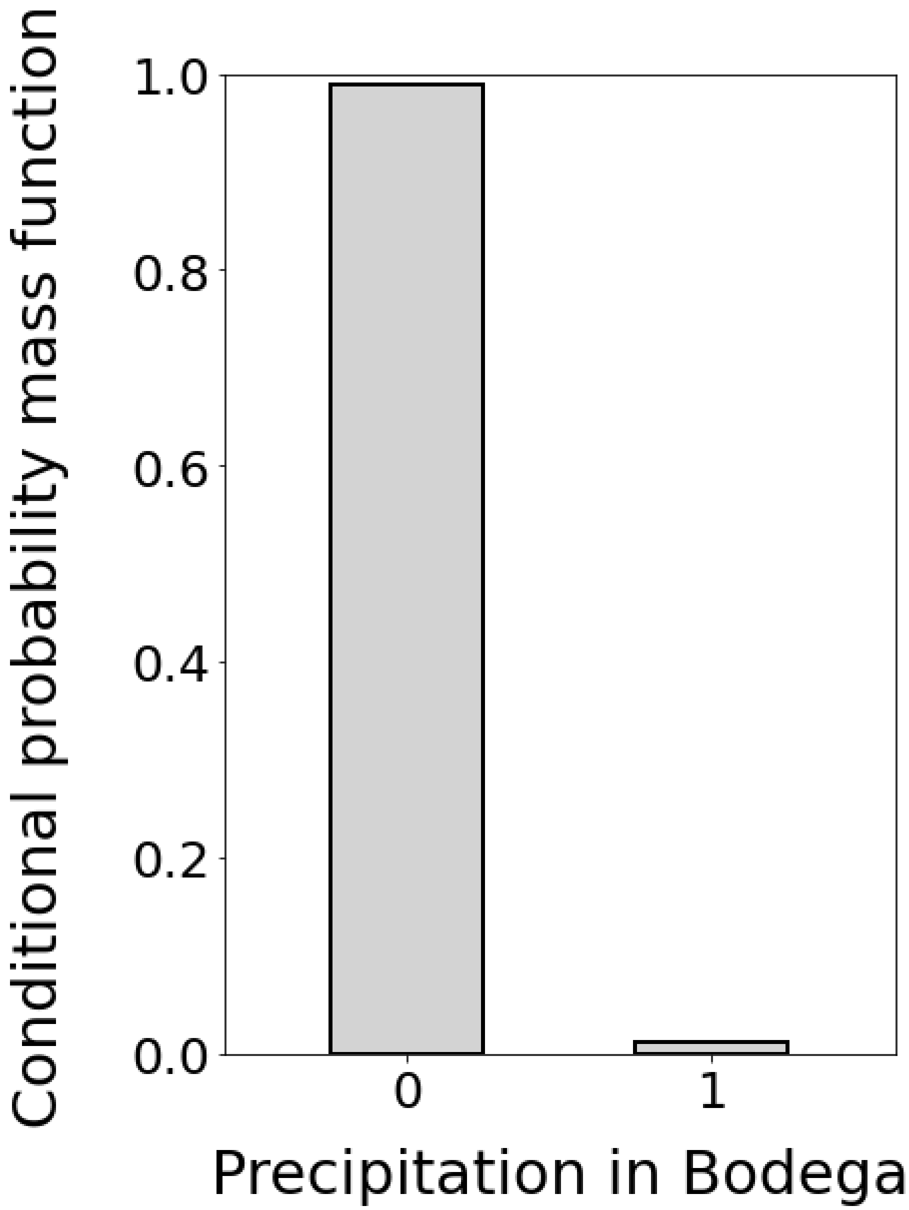
```python
# Compute conditional pmf of each of the Bernoulli random variables given the ot
def conditional_1_station_given_2(joint_pmf):
    # INSERT YOUR CODE HERE

    ### STATION ONE ###

    # C1: s1 = 0, s2 = 0, s3 = 0
    c1_s1 = joint_pmf[0,0,0] / marginal_2_stations(joint_pmf)[2][0][0]
    c1_s1 = np.nan_to_num(c1_s1)

    # C2: s1 = 0, s2 = 0, s3 = 1
    c2_s1 = joint_pmf[0,0,1] / marginal_2_stations(joint_pmf)[2][0][1]
    c2_s1 = np.nan_to_num(c2_s1)

    # C3: s1 = 0, s2 = 1, s3 = 0
    c3_s1 = joint_pmf[0,1,0] / marginal_2_stations(joint_pmf)[2][1][0]
    c3_s1 = np.nan_to_num(c3_s1)

    # C4: s1 = 0, s2 = 1, s3 = 1
    c4_s1 = joint_pmf[0,1,1] / marginal_2_stations(joint_pmf)[2][1][1]
    c4_s1 = np.nan_to_num(c4_s1)

    # C5: s1 = 1, s2 = 0, s3 = 0
    c5_s1 = joint_pmf[1,0,0] / marginal_2_stations(joint_pmf)[2][0][0]
    c5_s1 = np.nan_to_num(c5_s1)

    # C6: s1 = 1, s2 = 0, s3 = 1
    c6_s1 = joint_pmf[1,0,1] / marginal_2_stations(joint_pmf)[2][0][1]
    c6_s1 = np.nan_to_num(c6_s1)

    # C7: s1 = 1, s2 = 1, s3 = 0
    c7_s1 = joint_pmf[1,1,0] / marginal_2_stations(joint_pmf)[2][1][0]
    c7_s1 = np.nan_to_num(c7_s1)

    # C8: s1 = 1, s2 = 1, s3 = 1
    c8_s1 = joint_pmf[1,1,1] / marginal_2_stations(joint_pmf)[2][1][1]
    c8_s1 = np.nan_to_num(c8_s1)

    # c1 = [[c1_s1, c2_s1, c3_s1, c4_s1], [c5_s1, c6_s1, c7_s1, c8_s1]]

    c1 = [[[c1_s1, c5_s1], [c2_s1, c6_s1]], [[c3_s1, c7_s1], [c4_s1, c8_s1]]]


    #c1 = np.zeros((2,2))
    #c1[0,0] = [c1_s1, c5_s1]
    #c1[0,1] = [c2_s1, c6_s1]
    #c1[1,0] = [c3_s1, c7_s1]
    #c1[1,1] = [c4_s1, c8_s1]



    ### STATION TWO ###

    # C1: s1 = 0, s2 = 0, s3 = 0
    c1_s2 = joint_pmf[0,0,0] / marginal_2_stations(joint_pmf)[1][0][0]
```

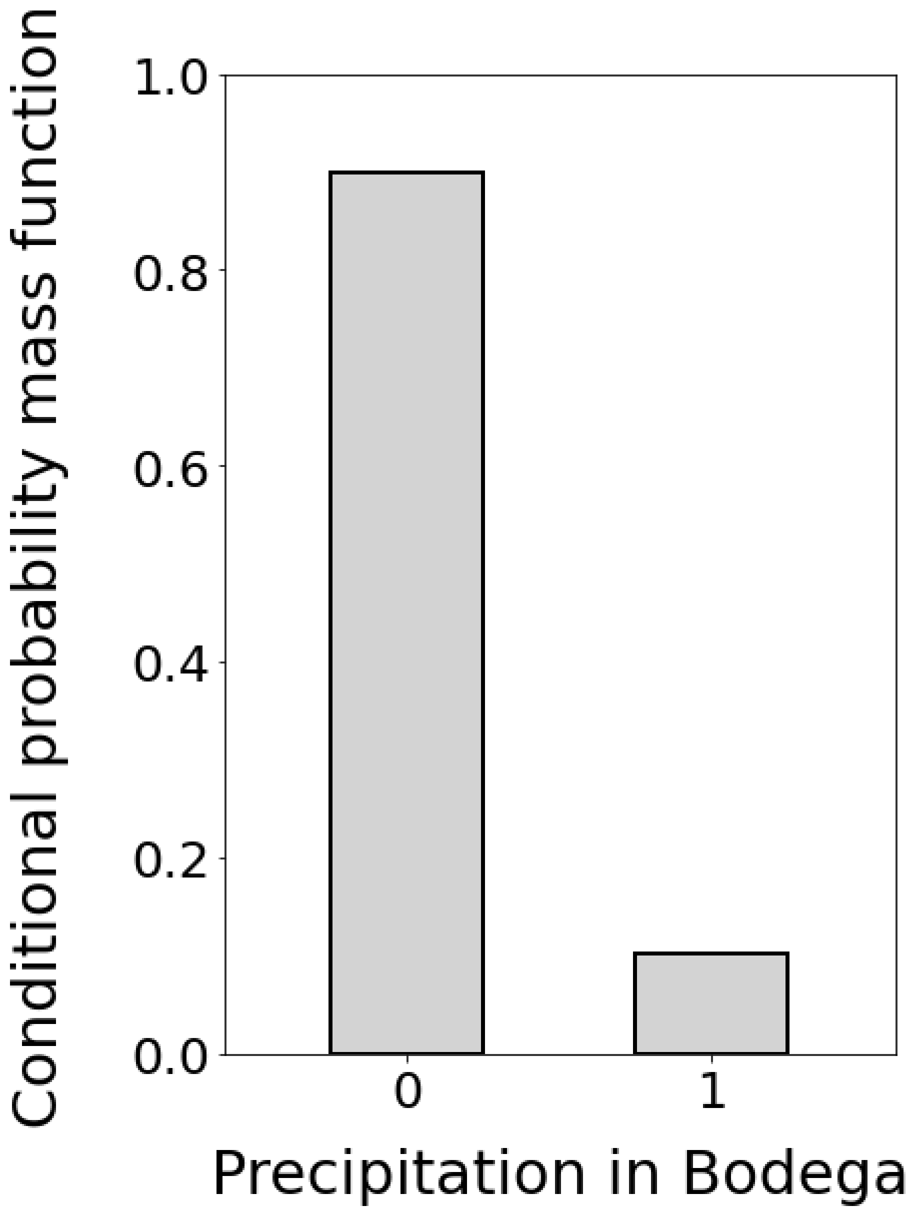
```
c1_s2 = np.nan_to_num(c1_s2)

# C2: s1 = 0, s2 = 0, s3 = 1
c2_s2 = joint_pmf[0,0,1] / marginal_2_stations(joint_pmf)[1][0][1]
c2_s2 = np.nan_to_num(c2_s2)

# C3: s1 = 1, s2 = 0, s3 = 0
c3_s2 = joint_pmf[1,0,0] / marginal_2_stations(joint_pmf)[1][1][0]
c3_s2 = np.nan_to_num(c3_s2)

# C4: s1 = 1, s2 = 0, s3 = 1
c4_s2 = joint_pmf[1,0,1] / marginal_2_stations(joint_pmf)[1][1][1]
c4_s2 = np.nan_to_num(c4_s2)

# C5: s1 = 0, s2 = 1, s3 = 0
c5_s2 = joint_pmf[0,1,0] / marginal_2_stations(joint_pmf)[1][0][0]
c5_s2 = np.nan_to_num(c5_s2)

# C6: s1 = 0, s2 = 1, s3 = 1
c6_s2 = joint_pmf[0,1,1] / marginal_2_stations(joint_pmf)[1][0][1]
c6_s2 = np.nan_to_num(c6_s2)

# C7: s1 = 1, s2 = 1, s3 = 0
c7_s2 = joint_pmf[1,1,0] / marginal_2_stations(joint_pmf)[1][1][0]
c7_s2 = np.nan_to_num(c7_s2)

# C8: s1 = 1, s2 = 1, s3 = 1
c8_s2 = joint_pmf[1,1,1] / marginal_2_stations(joint_pmf)[1][1][1]
c8_s2 = np.nan_to_num(c8_s2)

#c2 = [[c1_s2, c2_s2, c3_s2, c4_s2], [c5_s2, c6_s2, c7_s2, c8_s2]]
c2 = [[[c1_s2, c5_s2], [c2_s2, c6_s2]], [[c3_s2, c7_s2], [c4_s2, c8_s2]]]


### STATION THREE ###

# C1: s1 = 0, s2 = 0, s3 = 0
c1_s3 = joint_pmf[0,0,0] / marginal_2_stations(joint_pmf)[0][0][0]
c1_s3 = np.nan_to_num(c1_s3)

# C2: s1 = 0, s2 = 1, s3 = 0
c2_s3 = joint_pmf[0,1,0] / marginal_2_stations(joint_pmf)[0][0][1]
c2_s3 = np.nan_to_num(c2_s3)

# C3: s1 = 1, s2 = 0, s3 = 0
c3_s3 = joint_pmf[1,0,0] / marginal_2_stations(joint_pmf)[0][1][0]
c3_s3 = np.nan_to_num(c3_s3)

# C4: s1 = 1, s2 = 1, s3 = 0
c4_s3 = joint_pmf[1,1,0] / marginal_2_stations(joint_pmf)[0][1][1]
c4_s3 = np.nan_to_num(c4_s3)

# C5: s1 = 0, s2 = 0, s3 = 1
c5_s3 = joint_pmf[0,0,1] / marginal_2_stations(joint_pmf)[0][0][0]
c5_s3 = np.nan_to_num(c5_s3)

# C6: s1 = 0, s2 = 1, s3 = 1
c6_s3 = joint_pmf[0,1,1] / marginal_2_stations(joint_pmf)[0][0][1]
c6_s3 = np.nan_to_num(c6_s3)

# C7: s1 = 1, s2 = 0, s3 = 1
```

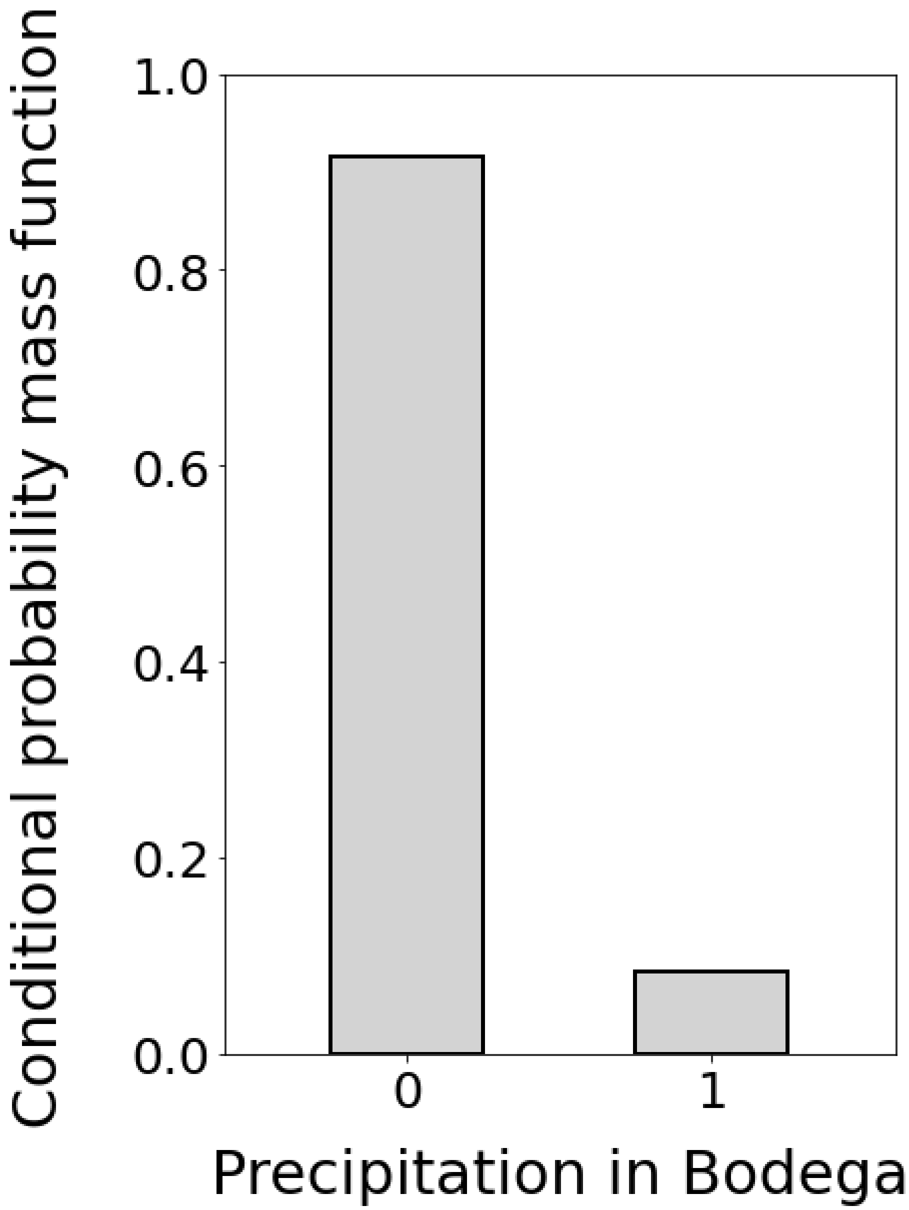
```
    c7_s3 = joint_pmf[1,0,1] / marginal_2_stations(joint_pmf)[0][1][0]
    c7_s3 = np.nan_to_num(c7_s3)

    # C8: s1 = 1, s2 = 1, s3 = 1
    c8_s3 = joint_pmf[1,1,1] / marginal_2_stations(joint_pmf)[0][1][1]
    c8_s3 = np.nan_to_num(c8_s3)

    # c3 = [[c1_s3, c2_s3, c3_s3, c4_s3], [c5_s3, c6_s3, c7_s3, c8_s3]]
    c3 = [[[c1_s3, c5_s3], [c2_s3, c6_s3]], [[c3_s3, c7_s3], [c4_s3, c8_s3]]]

    return c1, c2, c3




    '''
    ## 1 Given 23
    numerator1 = joint_pmf[1,:,:]
    denom1 = marginal_2_stations(joint_pmf)[2]

    c_1_23 = np.divide(numerator1, denom1)

    ## 2 Given 13
    numerator2 = joint_pmf[:,1,:]
    denom2 = marginal_2_stations(joint_pmf)[1]

    c_2_13 = np.divide(numerator2, denom2)

    ## 3 Given 12
    numerator3 = joint_pmf[:,:,1]
    denom3 = marginal_2_stations(joint_pmf)[0]

    c_3_12 = np.divide(numerator2, denom2)

    return c_1_23, c_2_13, c_3_12
    '''

cond_1_given_23 = conditional_1_station_given_2(joint_pmf)
print(cond_1_given_23)

cond_1_given_23,cond_2_given_13,cond_3_given_12 = conditional_1_station_given_2(

for ind_2 in range(2):
    for ind_3 in range(2):
        print(cond_1_given_23[ind_2][ind_3])
        plt.figure(figsize=(6,9))
        plt.bar(vals,cond_1_given_23[ind_2][ind_3], width = 0.5, color = "lightg
        plt.xticks(np.arange(0, 1+1, 1))
        plt.xticks(fontsize=font_size_ticks)
        plt.yticks(fontsize=font_size_ticks)
        plt.ylim([0,ymax])
        plt.xlim([xmin,xmax])
        plt.ylabel("Conditional probability mass function",fontsize=font_size,la
        plt.xlabel("Precipitation in Bodega",fontsize=font_size,labelpad = 15)
        #plt.savefig('plots/precipitation_cond_pmf_1_given_2eq'+str(ind_2)+'_3eq

for ind_1 in range(2):
    for ind_3 in range(2):
        print(cond_2_given_13[ind_1][ind_3])
        plt.figure(figsize=(6,9))
        plt.bar(vals,cond_2_given_13[ind_1][ind_3], width = 0.5, color = "lightg
```

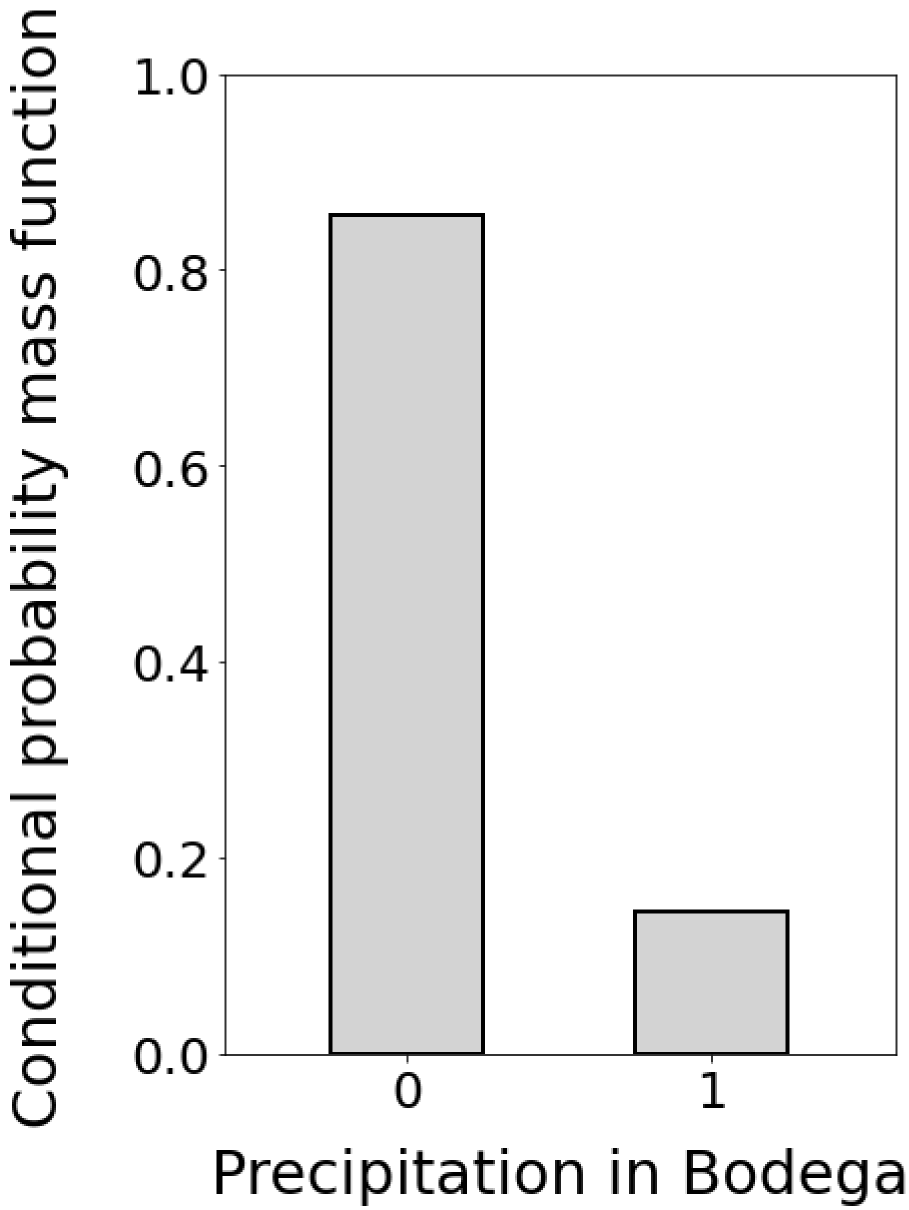
```
        plt.xticks(np.arange(0, 1+1, 1))
        plt.xticks(fontsize=font_size_ticks)
        plt.yticks(fontsize=font_size_ticks)
        plt.ylim([0,ymax])
        plt.xlim([xmin,xmax])
        plt.ylabel("Conditional probability mass function",fontsize=font_size,la
        plt.xlabel("Precipitation in Coos Bay",fontsize=font_size,labelpad = 15)
        #plt.savefig('plots/precipitation_cond_pmf_2_given_1eq'+str(ind_1)+'_3eq

for ind_1 in range(2):
    for ind_2 in range(2):
        print(cond_3_given_12[ind_1][ind_2])
        plt.figure(figsize=(6,9))
        plt.bar(vals,cond_3_given_12[ind_1][ind_2], width = 0.5, color = "lightg
        plt.xticks(np.arange(0, 1+1, 1))
        plt.xticks(fontsize=font_size_ticks)
        plt.yticks(fontsize=font_size_ticks)
        plt.ylim([0,ymax])
        plt.xlim([xmin,xmax])
        plt.ylabel("Conditional probability mass function",fontsize=font_size,la
        plt.xlabel("Precipitation in Riley",fontsize=font_size,labelpad = 15)
        #plt.savefig('plots/precipitation_cond_pmf_3_given_1eq'+str(ind_1)+'_2eq
```

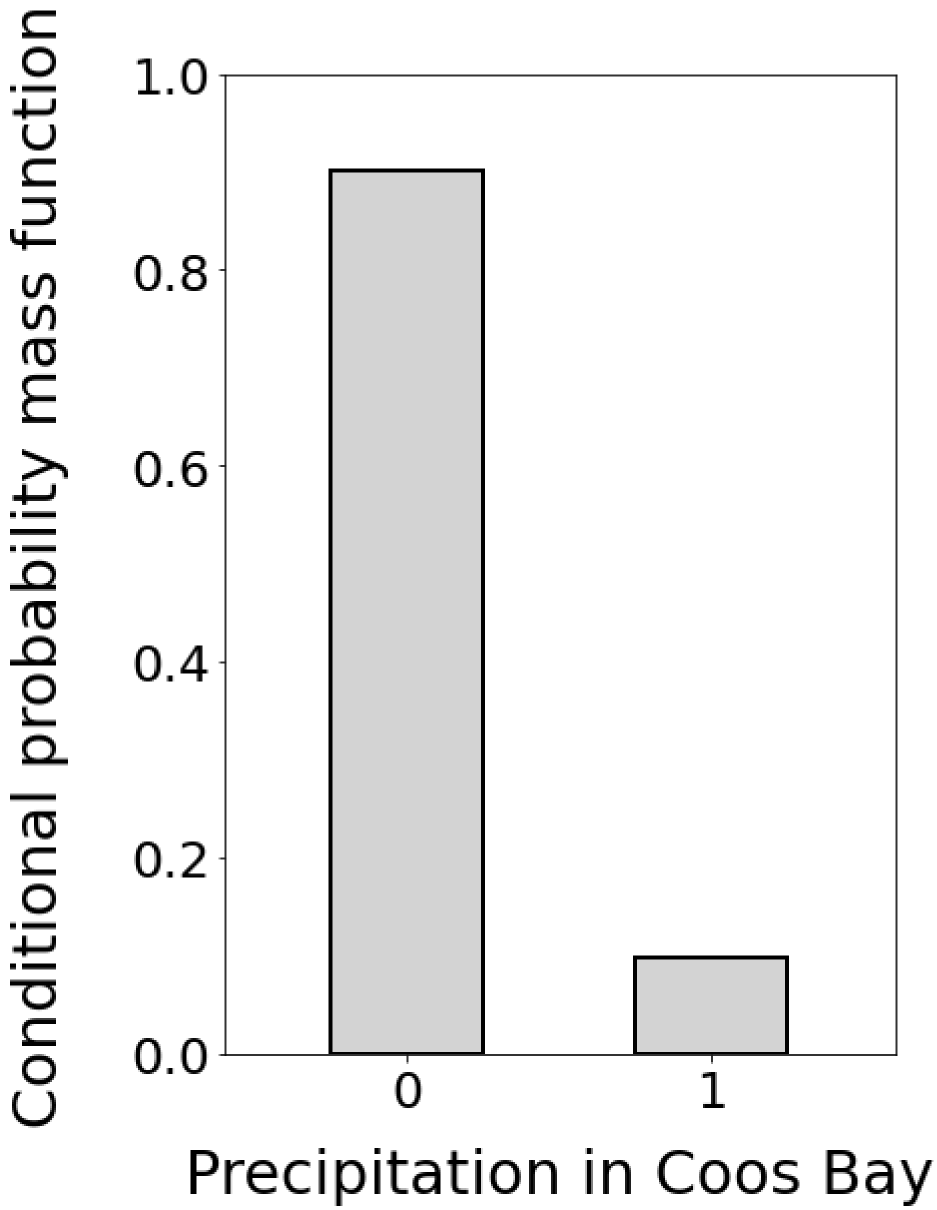
```
([[[0.9883597883597883, 0.01164021164021164], [0.8985507246376812, 0.10144927536
231885]], [[0.9156355455568054, 0.0843644544431946], [0.8557692307692307, 0.1442
3076923076922]]], [[[0.9017620082066136, 0.09823799179338642], [0.67636363636363
64, 0.32363636363636367]], [[0.539877300613497, 0.4601226993865031], [0.58333333
33333333, 0.4166666666666666]]], [[[0.9757116740663359, 0.024288325933664142],
[0.9014396456256922, 0.09856035437430788]], [[0.8073394495412843, 0.192660550458
7156], [0.8333333333333334, 0.16666666666666666]]])
[0.9883597883597883, 0.01164021164021164]
[0.8985507246376812, 0.10144927536231885]
[0.9156355455568054, 0.0843644544431946]
[0.8557692307692307, 0.14423076923076922]
[0.9017620082066136, 0.09823799179338642]
[0.6763636363636364, 0.32363636363636367]
[0.539877300613497, 0.4601226993865031]
[0.5833333333333333, 0.4166666666666666]
[0.9757116740663359, 0.024288325933664142]
[0.9014396456256922, 0.09856035437430788]
[0.8073394495412843, 0.1926605504587156]
[0.8333333333333334, 0.16666666666666666]
```

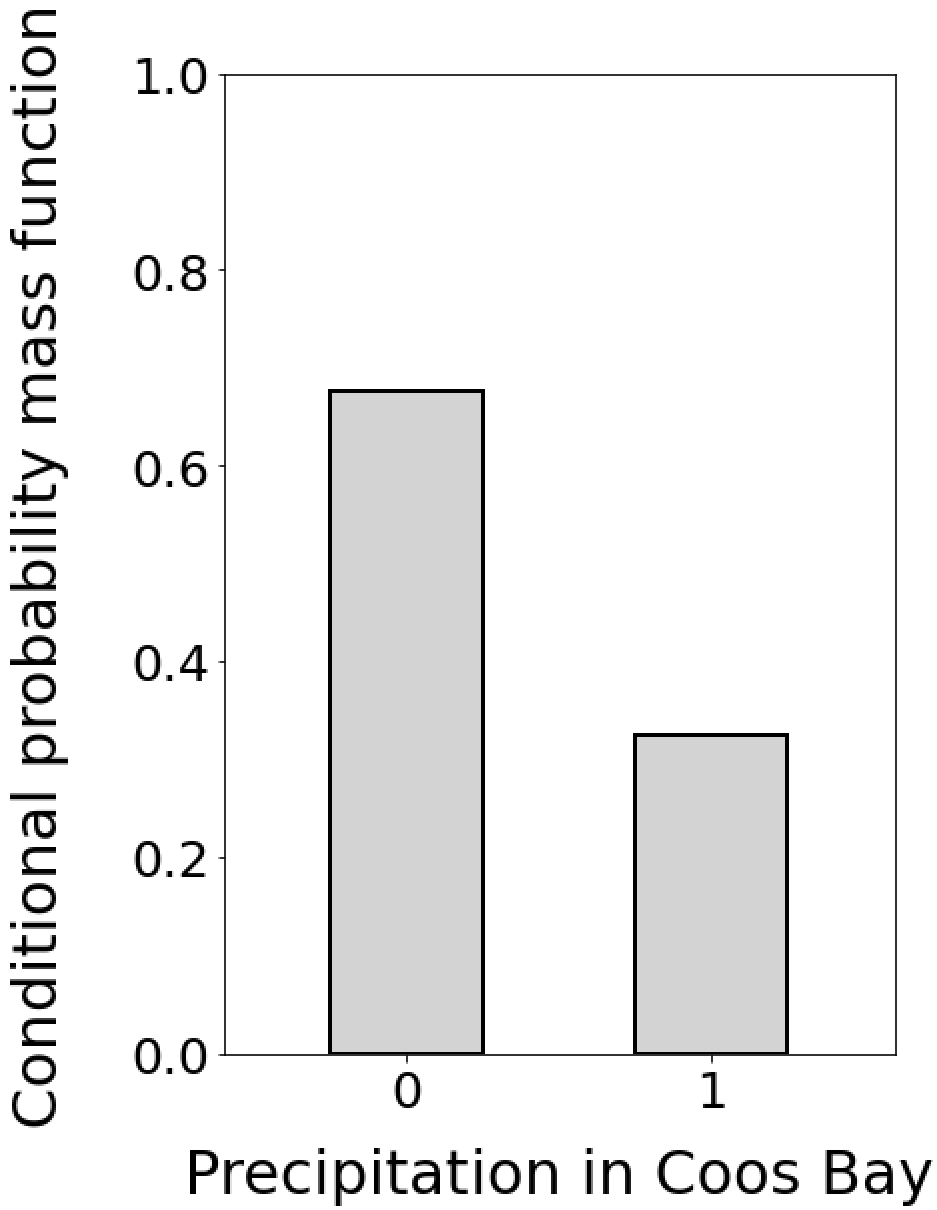

Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Bodega

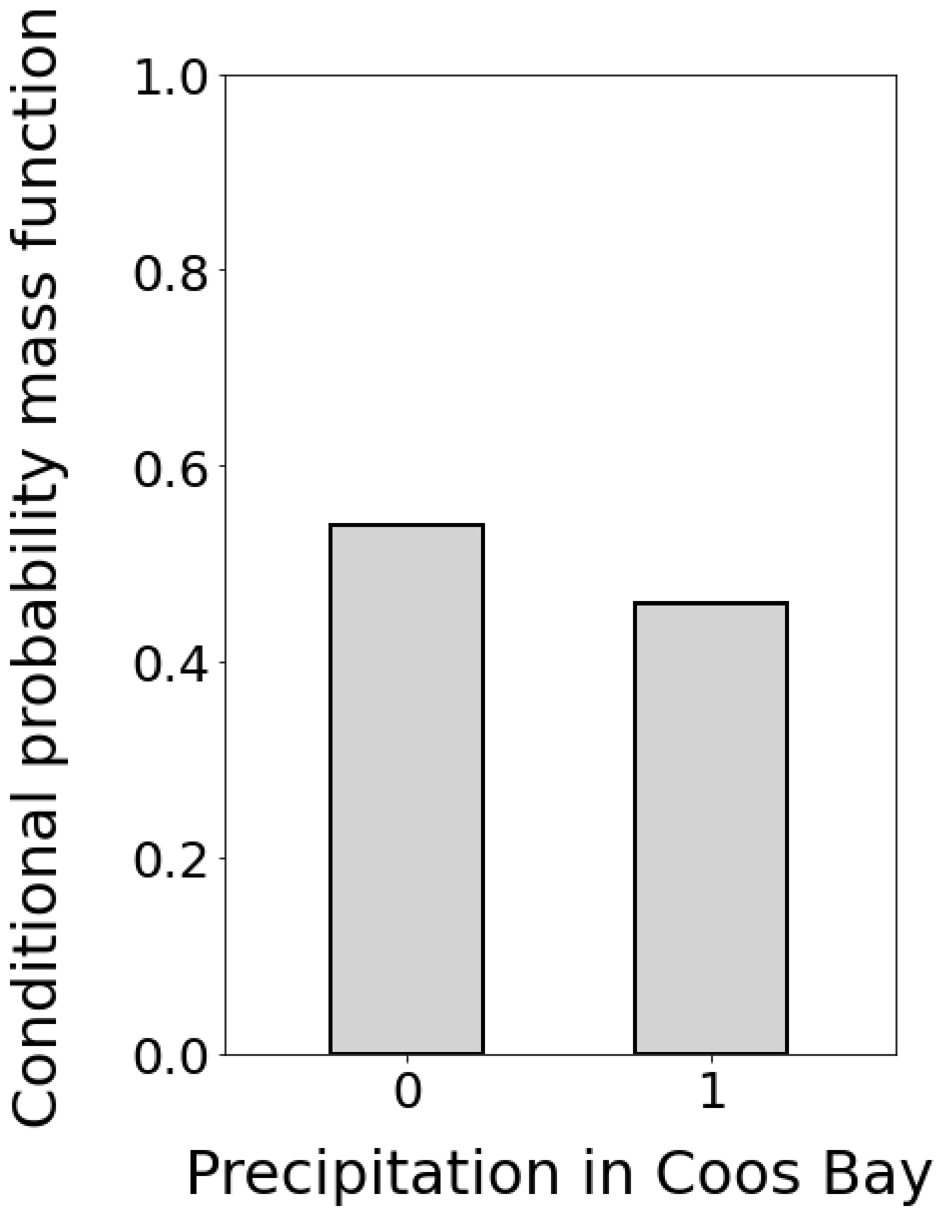

Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Bodega

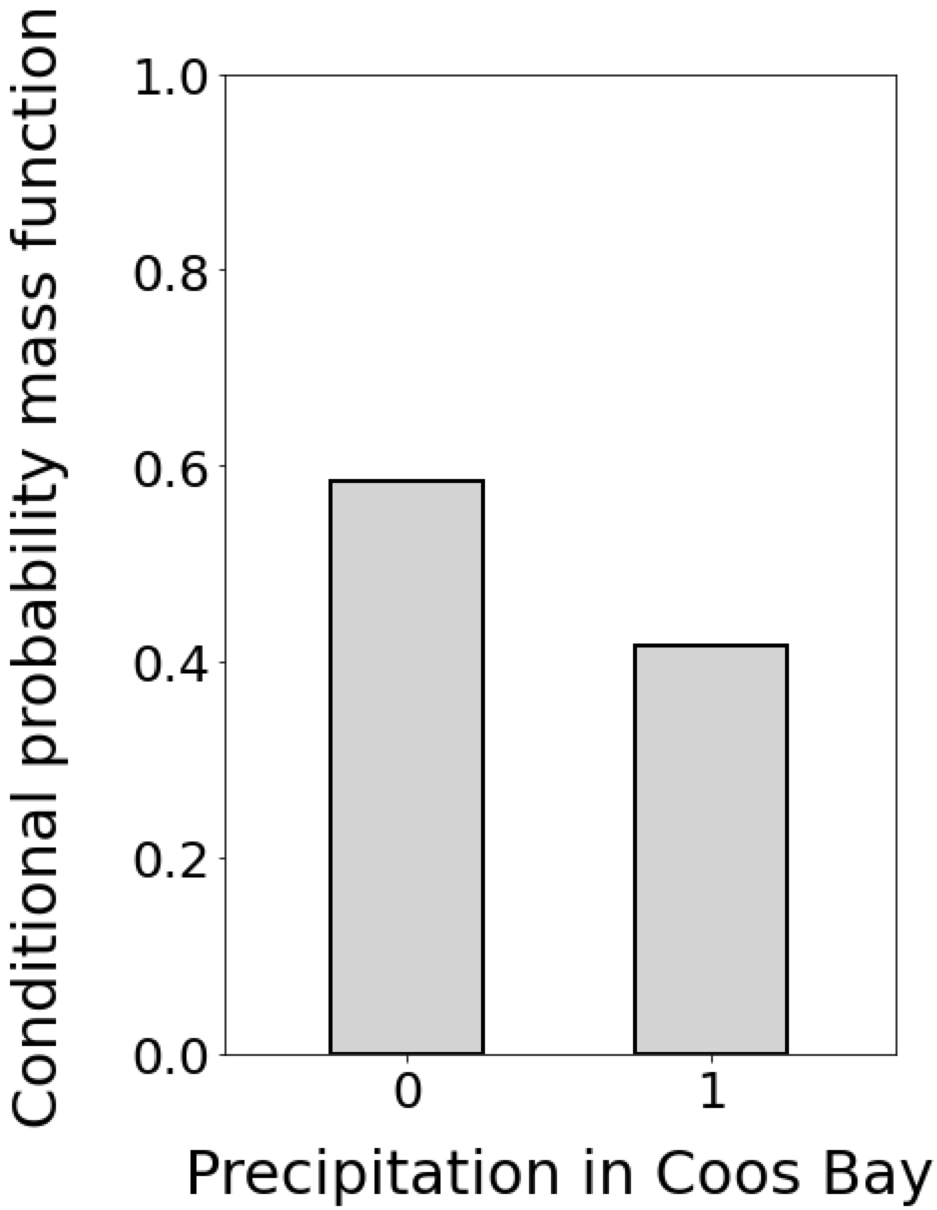

Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Bodega

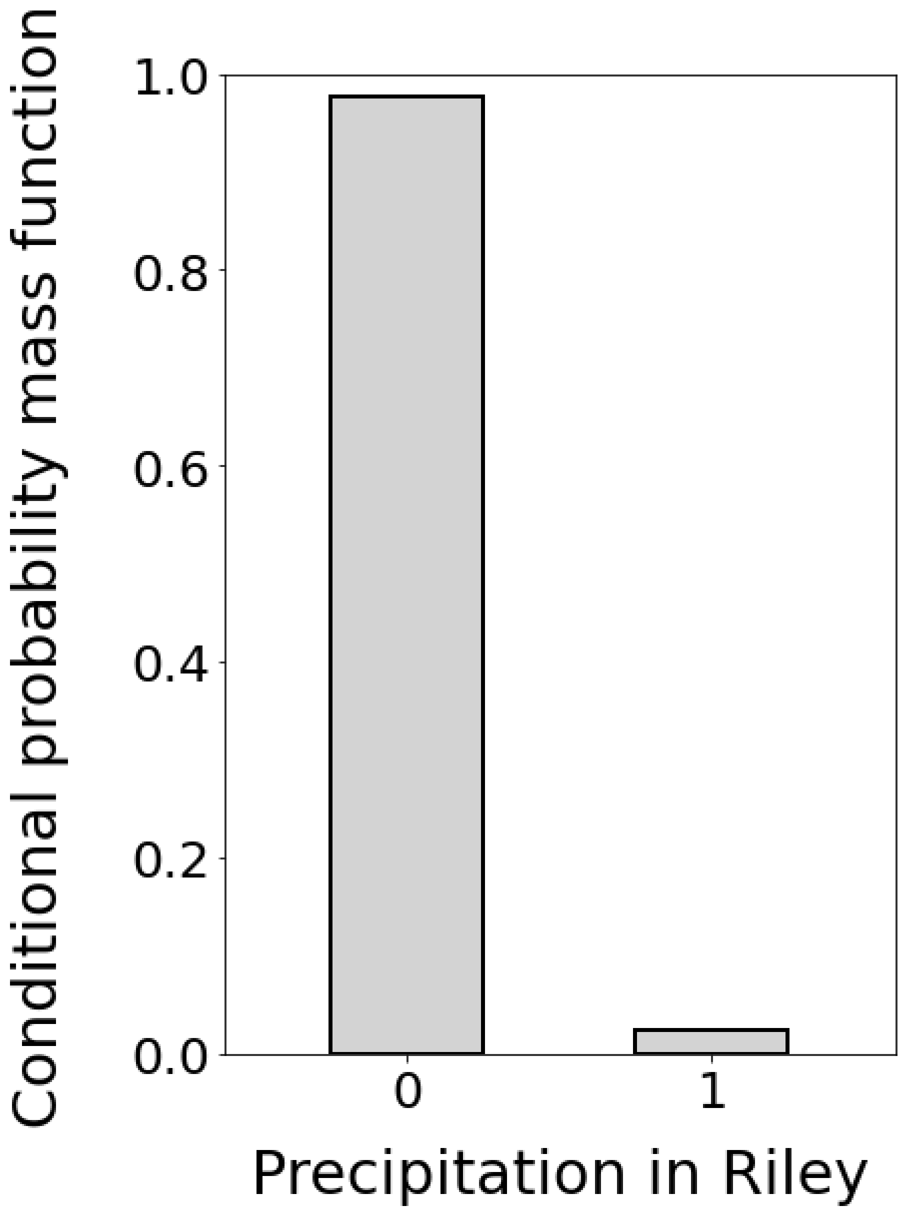

Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Bodega

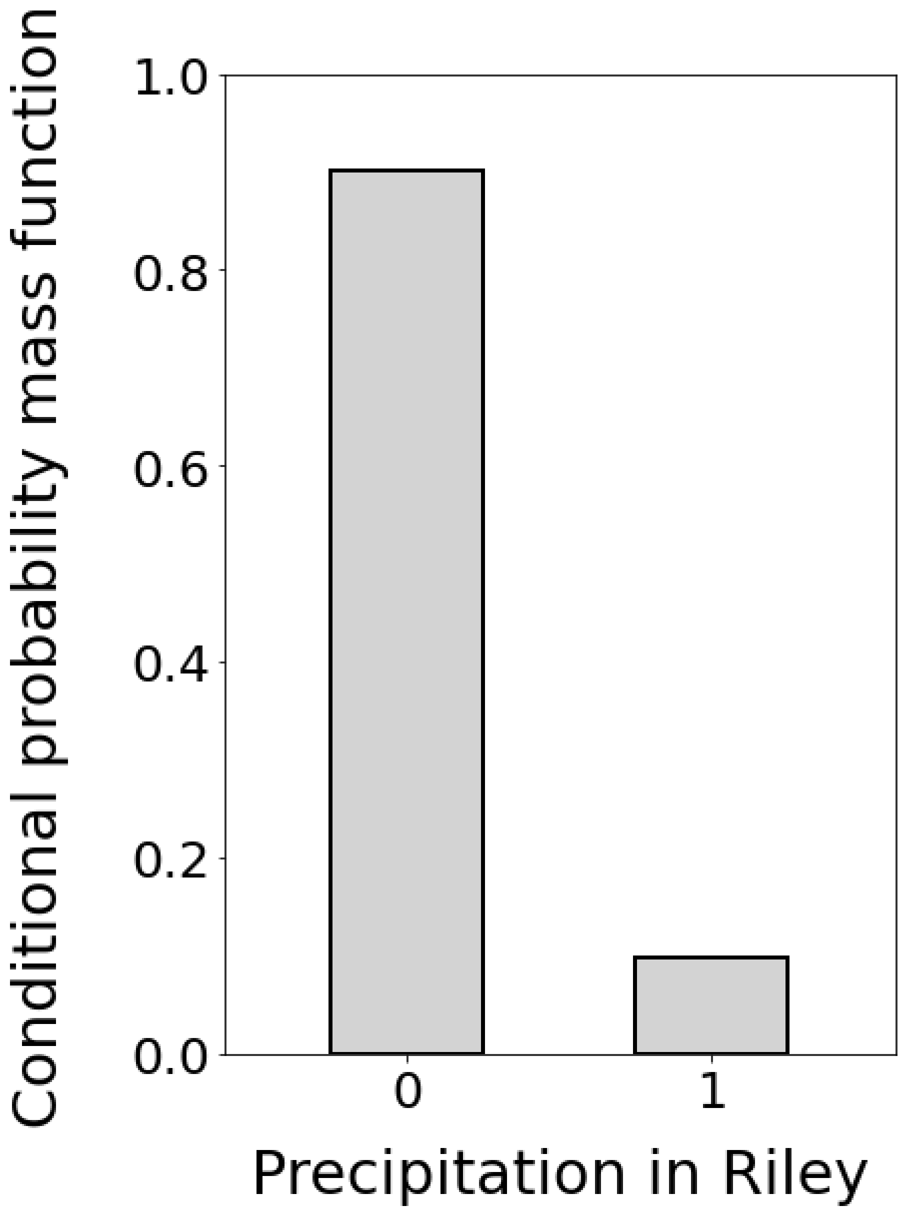

Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Coos Bay

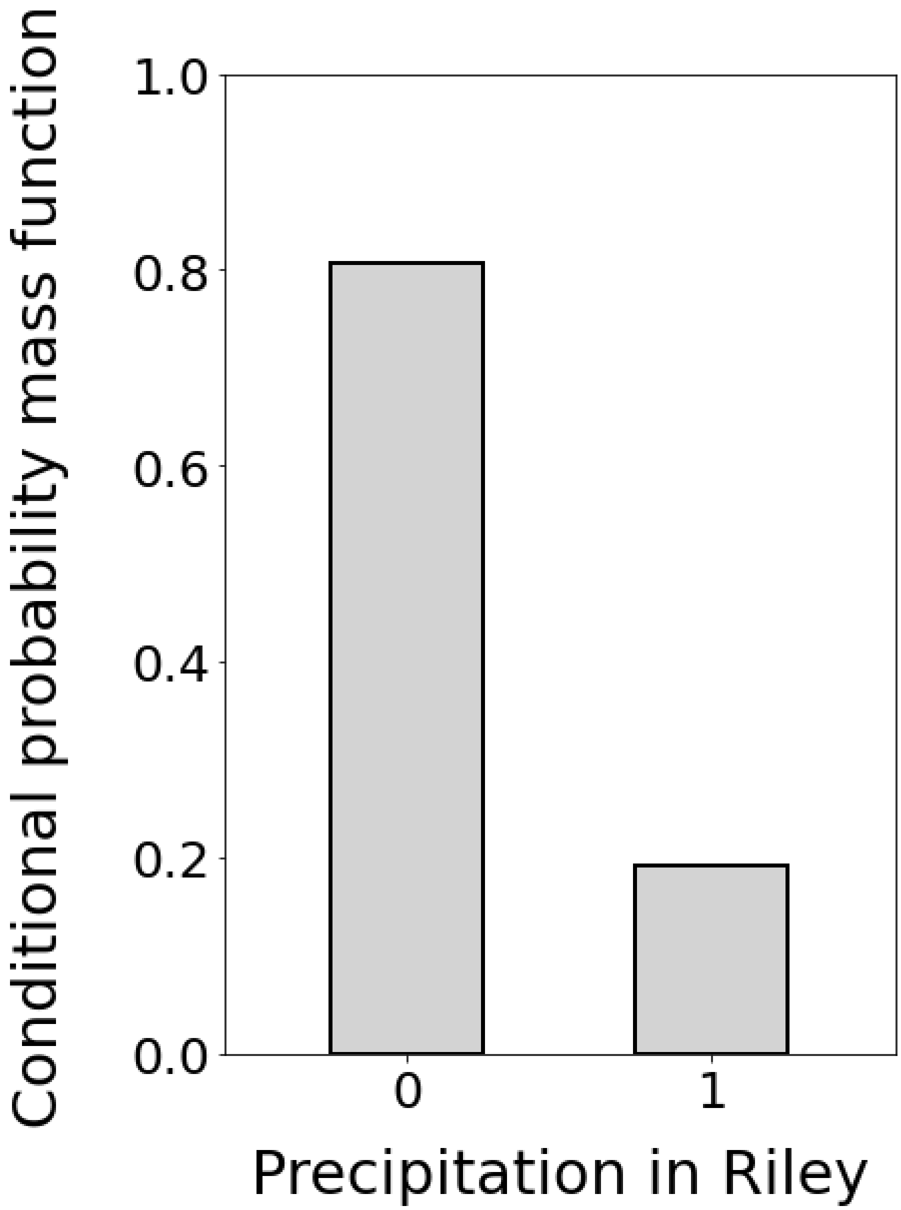

Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Coos Bay


Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Coos Bay

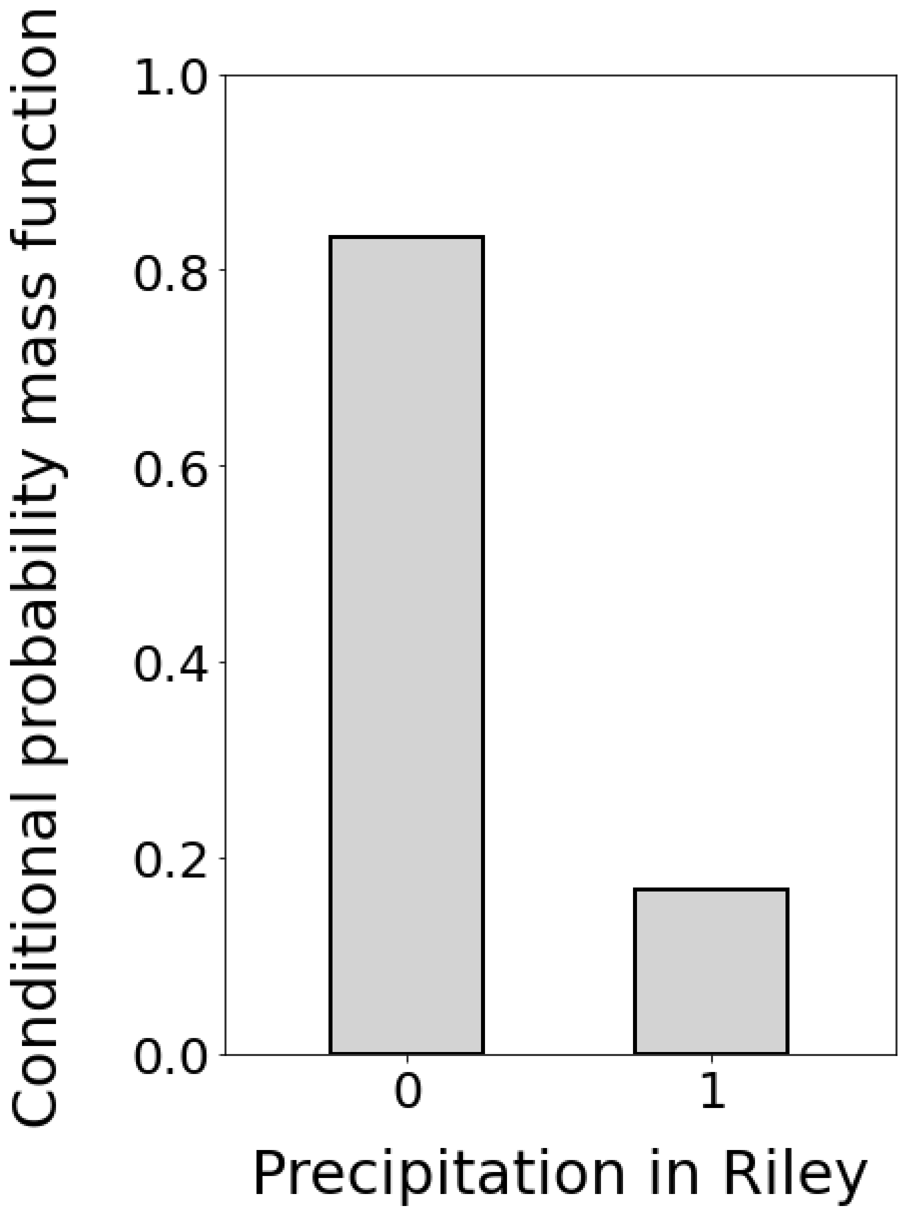

Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Coos Bay


Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Riley


Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Riley



Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Riley



In [ ]:

```
# Compute conditional joint pmf of each pair of the Bernoulli random variables g
def conditional_2_stations_given_1(joint_pmf):
    # INSERT YOUR CODE HERE

   ### STATION 1,2 | 3 ###

    # C1: s1 = 0, s2 = 0, s3 = 0
    c1_s12 = joint_pmf[0,0,0] / marginal_1_station(joint_pmf)[2][0]
    c1_s12 = np.nan_to_num(c1_s12)

    # C2: s1 = 0, s2 = 0, s3 = 1
    c2_s12 = joint_pmf[0,0,1] / marginal_1_station(joint_pmf)[2][1]
    c2_s12 = np.nan_to_num(c2_s12)

    # C3: s1 = 0, s2 = 1, s3 = 0
    c3_s12 = joint_pmf[0,1,0] / marginal_1_station(joint_pmf)[2][0]
    c3_s12 = np.nan_to_num(c3_s12)

    # C4: s1 = 0, s2 = 1, s3 = 1
    c4_s12 = joint_pmf[0,1,1] / marginal_1_station(joint_pmf)[2][1]
    c4_s12 = np.nan_to_num(c4_s12)
```

```
    # C5: s1 = 1, s2 = 0, s3 = 0
    c5_s12 = joint_pmf[1,0,0] / marginal_1_station(joint_pmf)[2][0]
    c5_s12 = np.nan_to_num(c5_s12)

    # C6: s1 = 1, s2 = 0, s3 = 1
    c6_s12 = joint_pmf[1,0,1] / marginal_1_station(joint_pmf)[2][1]
    c6_s12 = np.nan_to_num(c6_s12)

    # C7: s1 = 1, s2 = 1, s3 = 0
    c7_s12 = joint_pmf[1,1,0] / marginal_1_station(joint_pmf)[2][0]
    c7_s12 = np.nan_to_num(c7_s12)

    # C8: s1 = 1, s2 = 1, s3 = 1
    c8_s12 = joint_pmf[1,1,1] / marginal_1_station(joint_pmf)[2][1]
    c8_s12 = np.nan_to_num(c8_s12)

    c12 = [[c1_s12, c3_s12, c5_s12, c7_s12], [c2_s12, c4_s12, c6_s12, c8_s12]]


   ### STATION 1,3 | 2 ###

    # C1: s1 = 0, s2 = 0, s3 = 0
    c1_s13 = joint_pmf[0,0,0] / marginal_1_station(joint_pmf)[1][0]
    c1_s13 = np.nan_to_num(c1_s13)

    # C2: s1 = 0, s2 = 0, s3 = 1
    c2_s13 = joint_pmf[0,0,1] / marginal_1_station(joint_pmf)[1][0]
    c2_s13 = np.nan_to_num(c2_s13)

    # C3: s1 = 0, s2 = 1, s3 = 0
    c3_s13 = joint_pmf[0,1,0] / marginal_1_station(joint_pmf)[1][1]
    c3_s13 = np.nan_to_num(c3_s13)

    # C4: s1 = 0, s2 = 1, s3 = 1
    c4_s13 = joint_pmf[0,1,1] / marginal_1_station(joint_pmf)[1][1]
    c4_s13 = np.nan_to_num(c4_s13)

    # C5: s1 = 1, s2 = 0, s3 = 0
    c5_s13 = joint_pmf[1,0,0] / marginal_1_station(joint_pmf)[1][0]
    c5_s13 = np.nan_to_num(c5_s13)

    # C6: s1 = 1, s2 = 0, s3 = 1
    c6_s13 = joint_pmf[1,0,1] / marginal_1_station(joint_pmf)[1][0]
    c6_s13 = np.nan_to_num(c6_s13)

    # C7: s1 = 1, s2 = 1, s3 = 0
    c7_s13 = joint_pmf[1,1,0] / marginal_1_station(joint_pmf)[1][1]
    c7_s13 = np.nan_to_num(c7_s13)

    # C8: s1 = 1, s2 = 1, s3 = 1
    c8_s13 = joint_pmf[1,1,1] / marginal_1_station(joint_pmf)[1][1]
    c8_s13 = np.nan_to_num(c8_s13)

    c13 = [[c1_s13, c2_s13, c5_s13, c6_s13], [c3_s13, c4_s13, c7_s13, c8_s13]]


   ### STATION 2,3 | 1 ###

    # C1: s1 = 0, s2 = 0, s3 = 0
    c1_s23 = joint_pmf[0,0,0] / marginal_1_station(joint_pmf)[0][0]
```

```
    c1_s23 = np.nan_to_num(c1_s23)

    # C2: s1 = 0, s2 = 0, s3 = 1
    c2_s23 = joint_pmf[0,0,1] / marginal_1_station(joint_pmf)[0][0]
    c2_s23 = np.nan_to_num(c2_s23)

    # C3: s1 = 0, s2 = 1, s3 = 0
    c3_s23 = joint_pmf[0,1,0] / marginal_1_station(joint_pmf)[0][0]
    c3_s23 = np.nan_to_num(c3_s23)

    # C4: s1 = 0, s2 = 1, s3 = 1
    c4_s23 = joint_pmf[0,1,1] / marginal_1_station(joint_pmf)[0][0]
    c4_s23 = np.nan_to_num(c4_s23)

    # C5: s1 = 1, s2 = 0, s3 = 0
    c5_s23 = joint_pmf[1,0,0] / marginal_1_station(joint_pmf)[0][1]
    c5_s23 = np.nan_to_num(c5_s23)

    # C6: s1 = 1, s2 = 0, s3 = 1
    c6_s23 = joint_pmf[1,0,1] / marginal_1_station(joint_pmf)[0][1]
    c6_s23 = np.nan_to_num(c6_s23)

    # C7: s1 = 1, s2 = 1, s3 = 0
    c7_s23 = joint_pmf[1,1,0] / marginal_1_station(joint_pmf)[0][1]
    c7_s23 = np.nan_to_num(c7_s23)

    # C8: s1 = 1, s2 = 1, s3 = 1
    c8_s23 = joint_pmf[1,1,1] / marginal_1_station(joint_pmf)[0][1]
    c8_s23 = np.nan_to_num(c8_s23)

    c23 = [[c1_s23, c2_s23, c3_s23, c4_s23], [c5_s23, c6_s23, c7_s23, c8_s23]]

    return c12, c13, c23


cond_12_given_3,cond_13_given_2,cond_23_given_1 = conditional_2_stations_given_1

for ind in range(2):
    #print(process_name(file_name_list[stations[0]+1]) + " and " + process_name(
    #         + " conditioned on " + process_name(file_name_list[stations[2]+1])
    print(cond_12_given_3[ind])
for ind in range(2):
    #print(process_name(file_name_list[stations[0]+1]) + " and " + process_name(
    #         + " conditioned on " + process_name(file_name_list[stations[1]+1])
    print(cond_13_given_2[ind])
for ind in range(2):
    #print(process_name(file_name_list[stations[1]+1]) + " and " + process_name(
    #         + " conditioned on " + process_name(file_name_list[stations[0]+1])
    print(cond_23_given_1[ind])
```

```
[0.8843650136110782, 0.09634276245709551, 0.01041543377914546, 0.008876790152680
79]
[0.5980707395498391, 0.28617363344051444, 0.06752411575562701, 0.048231511254019
29]
[0.9620187974765031, 0.02394747006566242, 0.011329985837517704, 0.00270374662031
6725]
[0.8197381671701913, 0.08962739174219536, 0.0755287009063444, 0.0151057401812688
81]
[0.8727952342016119, 0.02172643382782385, 0.09508235019273449, 0.010395981777829
693]
```

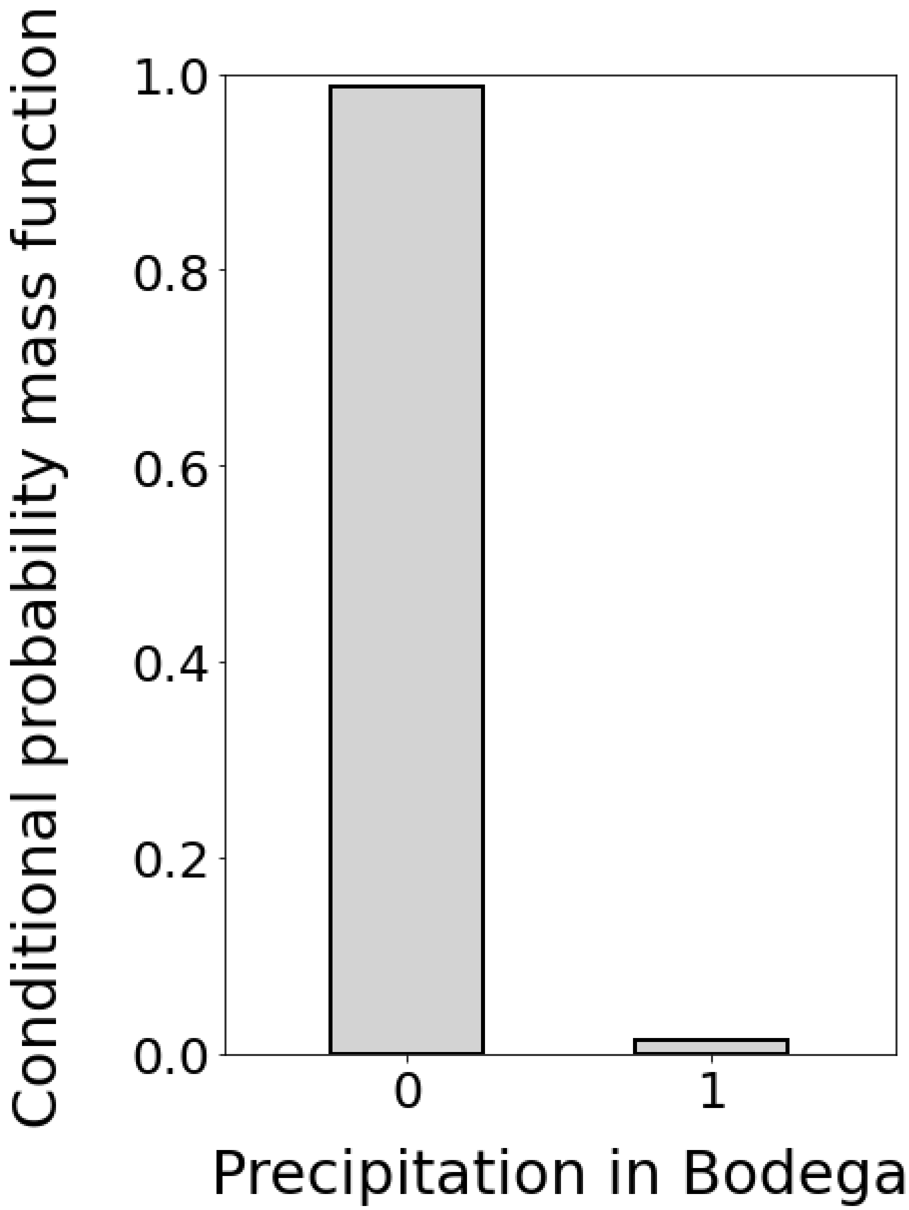
```
[0.44221105527638194, 0.10552763819095479, 0.3768844221105528, 0.075376884422110
56]
```

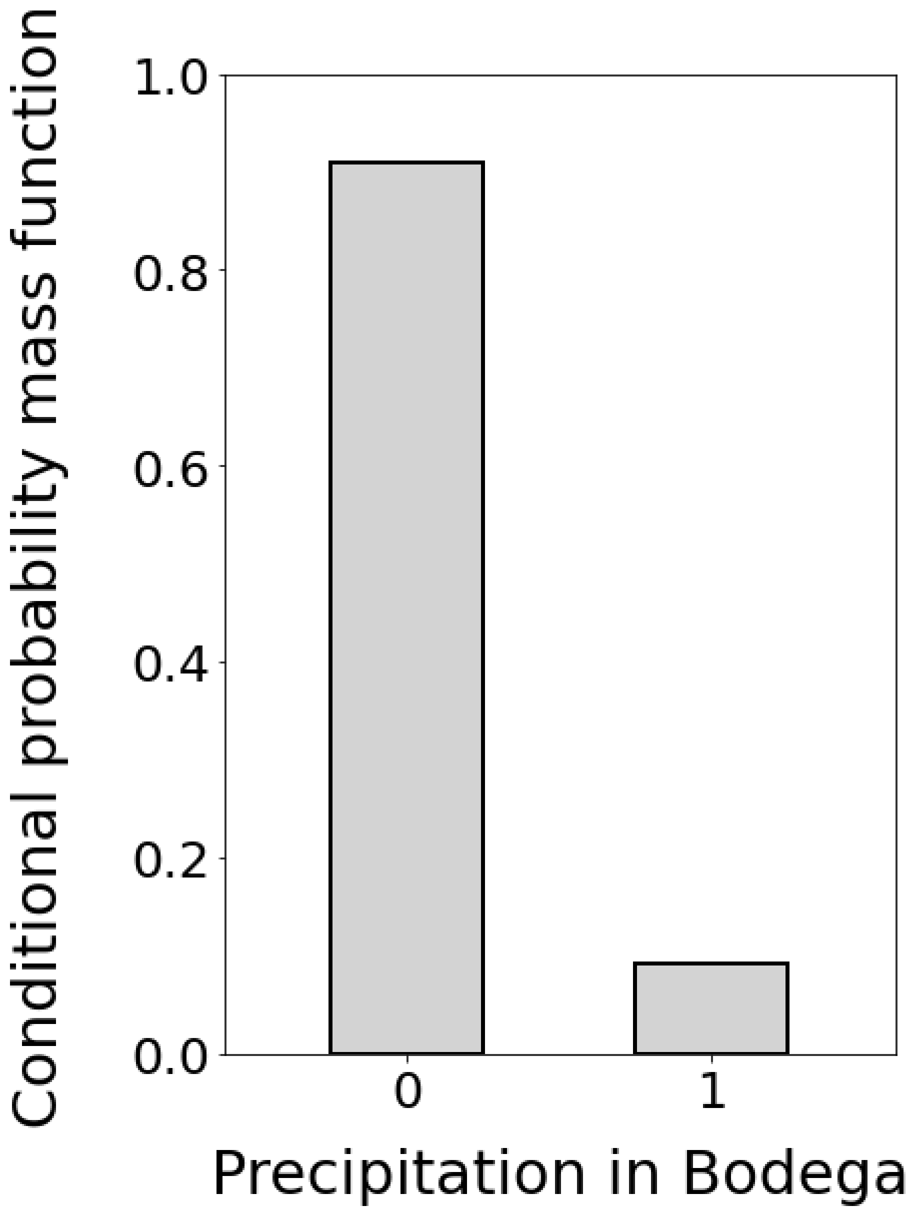
In [ ]:

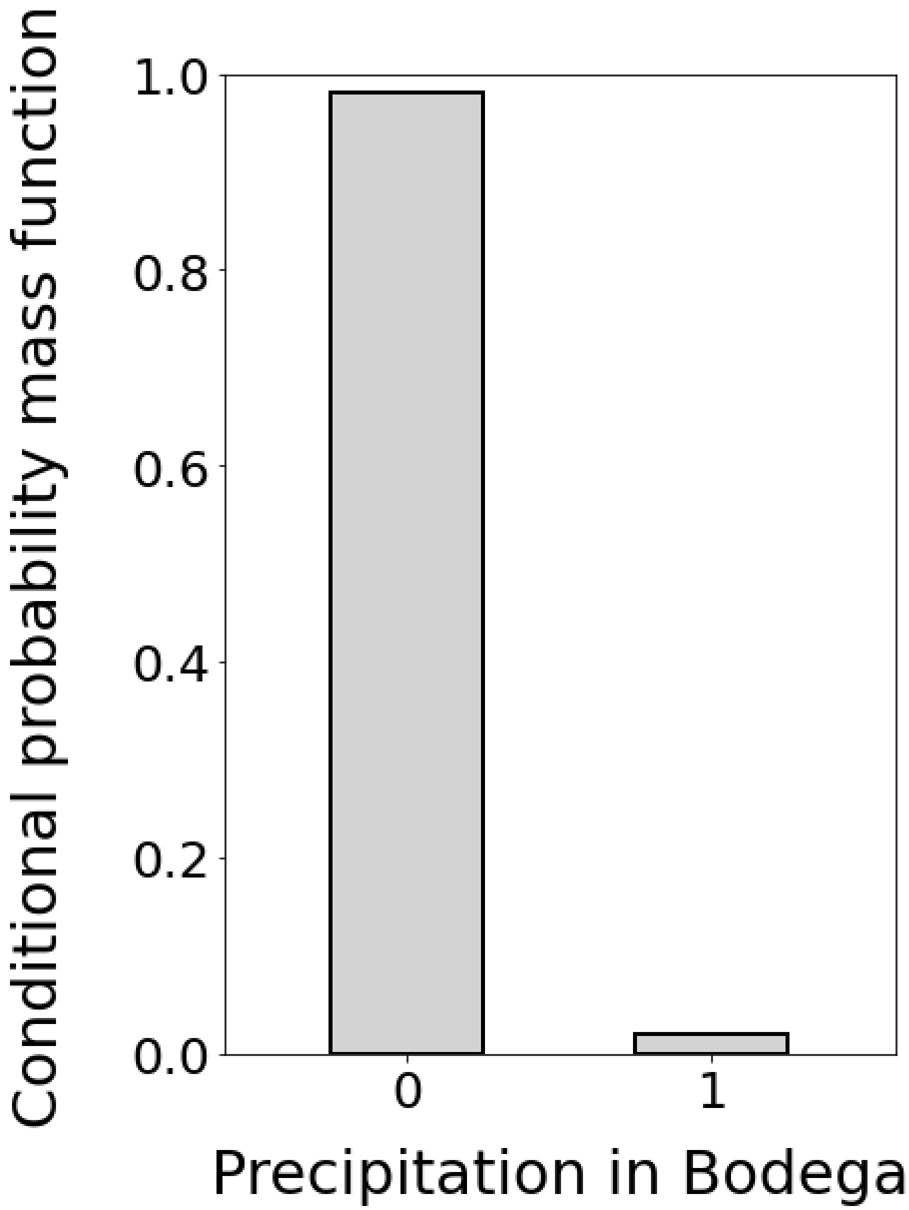
```python
# Compute conditional pmf of each Bernoulli random variable given each of the ot
# (i.e. Bodega just conditioned on Coos Bay, Bodega just conditioned on Riley, e
# Use a dictionary to save the conditional pmfs, for example cond_1["2"] should
# first random variable (Bodega)
def conditional_1_station_given_1_station(joint_pmf):
    # INSERT YOUR CODE HERE

    m1, m2, m3 =  marginal_1_station(joint_pmf)


    ### s1 GIVEN s2 ###

    ## s1 = 0 | s2 = 0
    s1_s2_1 = (joint_pmf[0,0,1] + joint_pmf[0,0,0]) / m2[0]

    ## s1 = 0 | s2 = 1
    s1_s2_2 = (joint_pmf[0,1,1] + joint_pmf[0,1,0]) / m2[1]

    ## s1 = 1 | s2 = 0
    s1_s2_3 = (joint_pmf[1,0,1] + joint_pmf[1,0,0]) / m2[0]

    ## s1 = 1 | s2 = 1
    s1_s2_4 = (joint_pmf[1,1,1] + joint_pmf[1,1,0]) / m2[1]

    s1Dict = {}
    s1Dict["2"] = [[s1_s2_1, s1_s2_3], [s1_s2_2, s1_s2_4]]


    ### s1 GIVEN s3 ###

    ## s1 = 0 | s3 = 0
    s1_s3_1 = (joint_pmf[0,0,0] + joint_pmf[0,1,0]) / m3[0]

    ## s1 = 0 | s3 = 1
    s1_s3_2 = (joint_pmf[0,0,1] + joint_pmf[0,1,1]) / m3[1]

    ## s1 = 1 | s3 = 0
    s1_s3_3 = (joint_pmf[1,0,0] + joint_pmf[1,1,0]) / m3[0]

    ## s1 = 1 | s3 = 1
    s1_s3_4 = (joint_pmf[1,0,1] + joint_pmf[1,1,1]) / m3[1]

    s1Dict["3"] = [[s1_s3_1, s1_s3_3], [s1_s3_2, s1_s3_4]]


    ### s2 GIVEN s1 ###

    ## s2 = 0 | s1 = 0
    s2_s1_1 = (joint_pmf[0,0,0] + joint_pmf[0,0,1]) / m1[0]

    ## s2 = 0 | s1 = 1
    s2_s1_2 = (joint_pmf[1,0,0] + joint_pmf[1,0,1]) / m1[1]

    ## s2 = 1 | s1 = 0
    s2_s1_3 = (joint_pmf[0,1,0] + joint_pmf[0,1,1]) / m1[0]

    ## s2 = 1 | s1 = 1
```

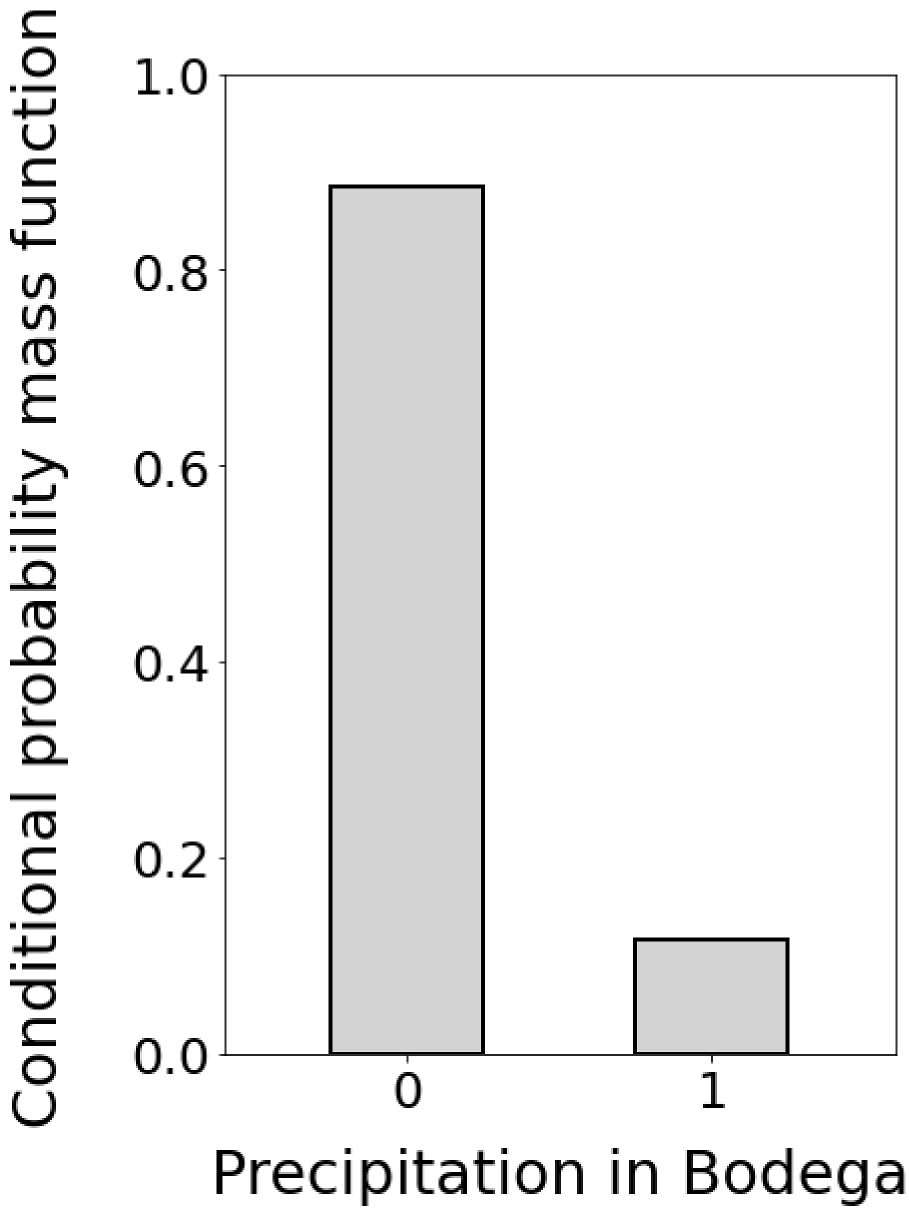
```
    s2_s1_4 = (joint_pmf[1,1,0] + joint_pmf[1,1,1]) / m1[1]

    s2Dict = {}
    s2Dict["1"] = [[s2_s1_1, s2_s1_3], [s2_s1_2, s2_s1_4]]

    ### s2 GIVEN s3 ###

    ## s2 = 0 | s3 = 0
    s2_s3_1 = (joint_pmf[0,0,0] + joint_pmf[1,0,0]) / m3[0]

    ## s2 = 0 | s3 = 1
    s2_s3_2 = (joint_pmf[0,0,1] + joint_pmf[1,0,1]) / m3[1]

    ## s2 = 1 | s3 = 0
    s2_s3_3 = (joint_pmf[0,1,0] + joint_pmf[1,1,0]) / m3[0]

    ## s2= 1 | s3 = 1
    s2_s3_4 = (joint_pmf[0,1,1] + joint_pmf[1,1,1]) / m3[1]

    s2Dict["3"] = [[s2_s3_1, s2_s3_3], [s2_s3_2, s2_s3_4]]


    ### s3 GIVEN s1 ###

    ## s3 = 0 | s1 = 0
    s3_s1_1 = (joint_pmf[0,0,0] + joint_pmf[0,1,0]) / m1[0]

    ## s3 = 0 | s1 = 1
    s3_s1_2 = (joint_pmf[1,0,0] + joint_pmf[1,1,0]) / m1[1]

    ## s3 = 1 | s1 = 0
    s3_s1_3 = (joint_pmf[0,0,1] + joint_pmf[0,1,1]) / m1[0]

    ## s3 = 1 | s1 = 1
    s3_s1_4 = (joint_pmf[1,0,1] + joint_pmf[1,1,1]) / m1[1]

    s3Dict = {}
    s3Dict["1"] = [[s3_s1_1, s3_s1_3], [s3_s1_2, s3_s1_4]]

    ### s3 GIVEN s2 ###

    ## s3 = 0 | s2 = 0
    s3_s2_1 = (joint_pmf[0,0,0] + joint_pmf[1,0,0]) / m2[0]

    ## s3 = 0 | s2 = 1
    s3_s2_2 = (joint_pmf[0,1,0] + joint_pmf[1,1,0]) / m2[1]

    ## s3 = 1 | s2 = 0
    s3_s2_3 = (joint_pmf[0,0,1] + joint_pmf[1,0,1]) / m2[0]

    ## s3 = 1 | s2 = 1
    s3_s2_4 = (joint_pmf[0,1,1] + joint_pmf[1,1,1]) / m2[1]

    s3Dict["2"] = [[s3_s2_1, s3_s2_3], [s3_s2_2, s3_s2_4]]


    return s1Dict, s2Dict, s3Dict


cond_1,cond_2,cond_3 = conditional_1_station_given_1_station(joint_pmf)
```

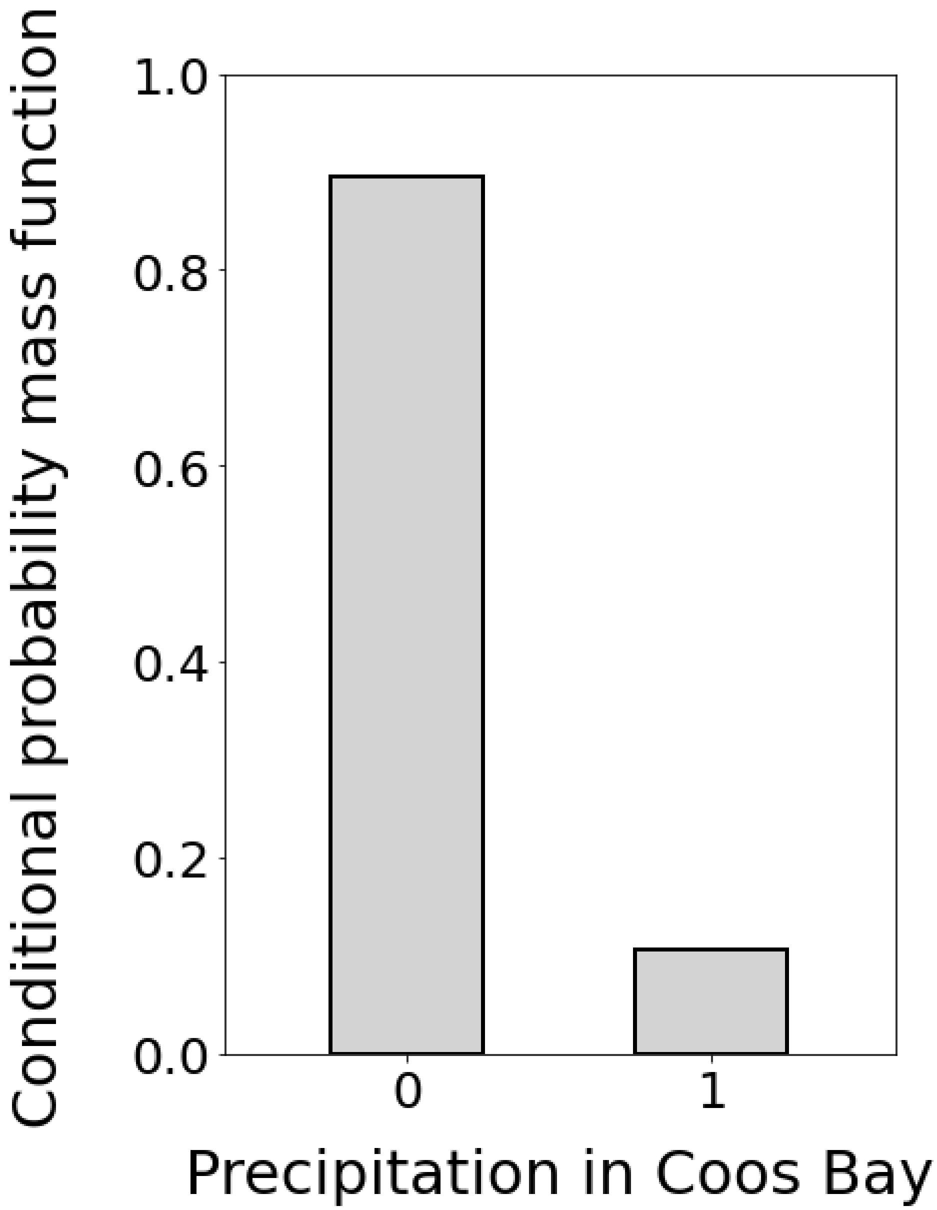
```
for given in ["2","3"]:
    for ind in range(2):
        plt.figure(figsize=(6,9))
        plt.bar(vals,cond_1[given][ind], width = 0.5, color = "lightgray", edgec
        plt.xticks(np.arange(0, 1+1, 1))
        plt.xticks(fontsize=font_size_ticks)
        plt.yticks(fontsize=font_size_ticks)
        plt.ylim([0,ymax])
        plt.xlim([xmin,xmax])
        plt.ylabel("Conditional probability mass function",fontsize=font_size,la
        plt.xlabel("Precipitation in Bodega",fontsize=font_size,labelpad = 15)
        #plt.savefig('plots/precipitation_cond_pmf_1_given_'+ given + 'eq'+str(i

for given in ["1","3"]:
    for ind in range(2):
        plt.figure(figsize=(6,9))
        plt.bar(vals,cond_2[given][ind], width = 0.5, color = "lightgray", edgec
        plt.xticks(np.arange(0, 1+1, 1))
        plt.xticks(fontsize=font_size_ticks)
        plt.yticks(fontsize=font_size_ticks)
        plt.ylim([0,ymax])
        plt.xlim([xmin,xmax])
        plt.ylabel("Conditional probability mass function",fontsize=font_size,la
        plt.xlabel("Precipitation in Coos Bay",fontsize=font_size,labelpad = 15)
        #plt.savefig('plots/precipitation_cond_pmf_2_given_'+ given + 'eq'+str(i

for given in ["1","2"]:
    for ind in range(2):
        plt.figure(figsize=(6,9))
        plt.bar(vals,cond_3[given][ind], width = 0.5, color = "lightgray", edgec
        plt.xticks(np.arange(0, 1+1, 1))
        plt.xticks(fontsize=font_size_ticks)
        plt.yticks(fontsize=font_size_ticks)
        plt.ylim([0,ymax])
        plt.xlim([xmin,xmax])
        plt.ylabel("Conditional probability mass function",fontsize=font_size,la
        plt.xlabel("Precipitation in Riley",fontsize=font_size,labelpad = 15)
        #plt.savefig('plots/precipitation_cond_pmf_3_given_'+ given + 'eq'+str(i
```

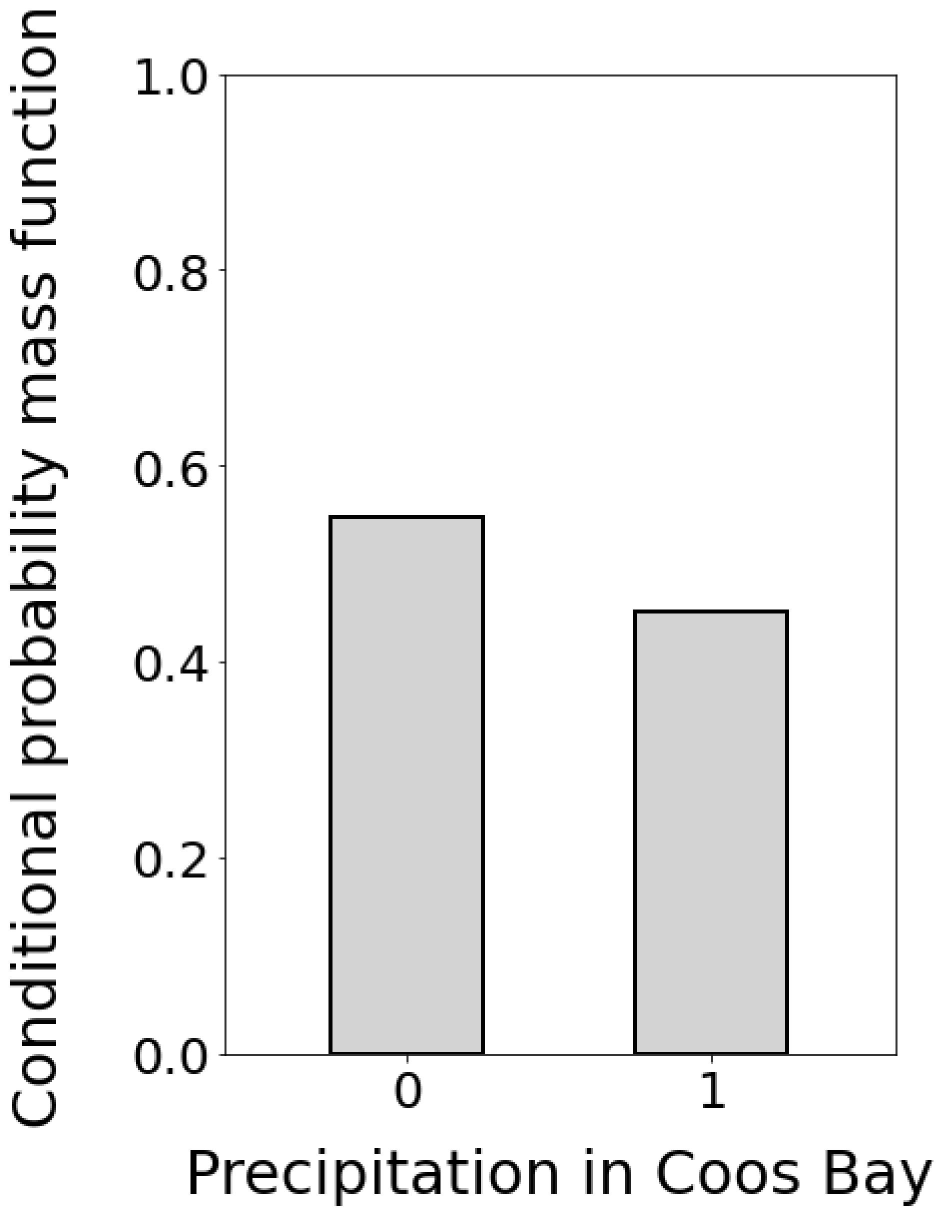

Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Bodega


Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Bodega

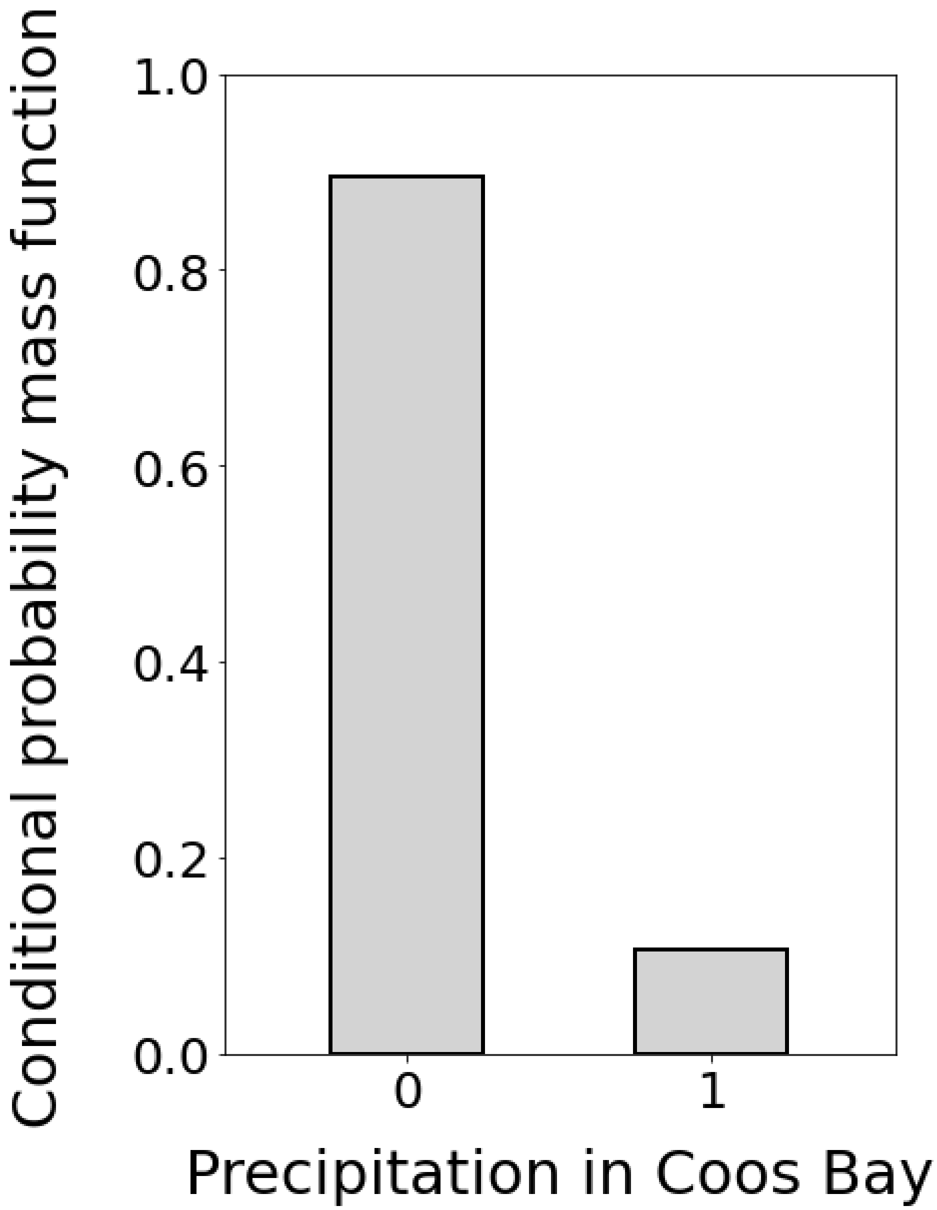

Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Bodega

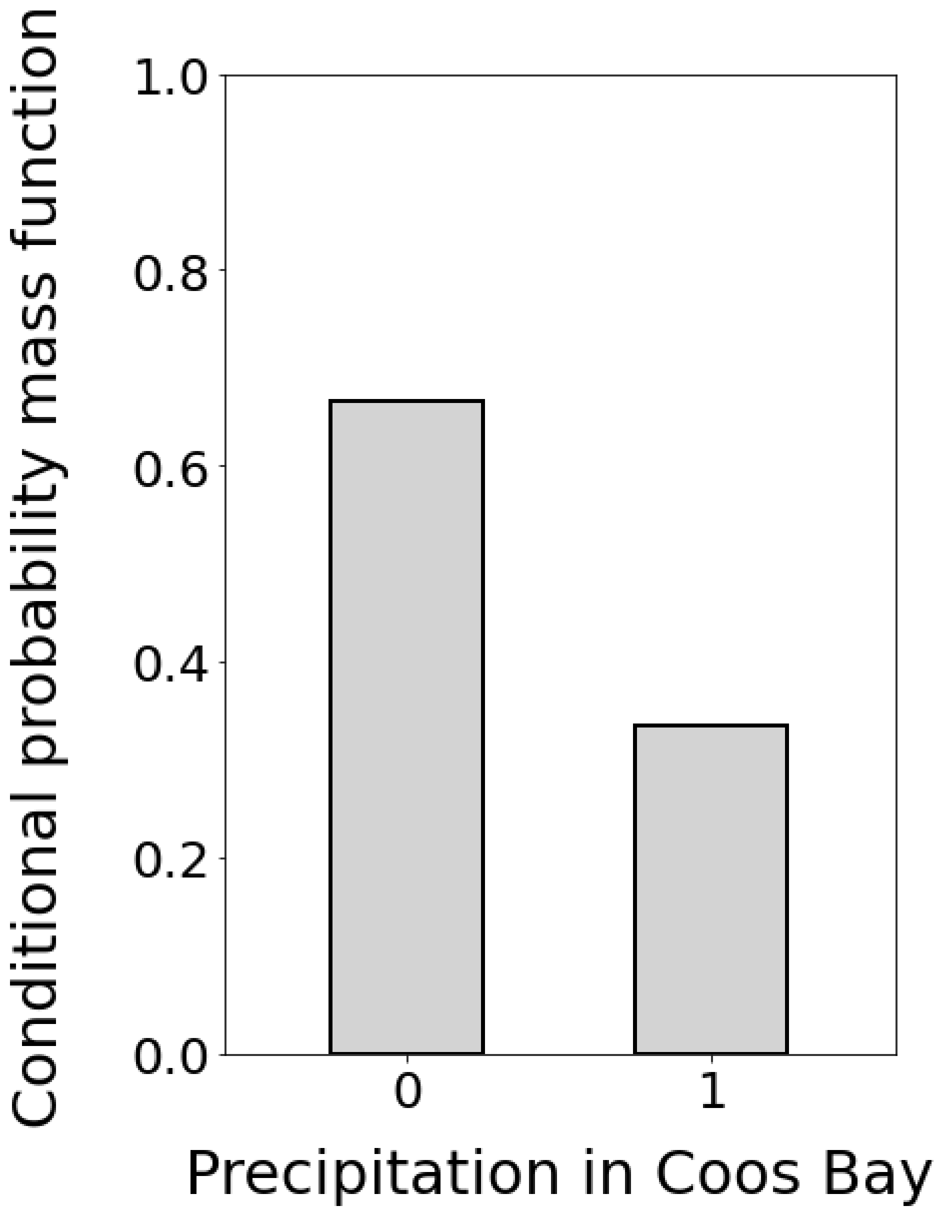

Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Bodega

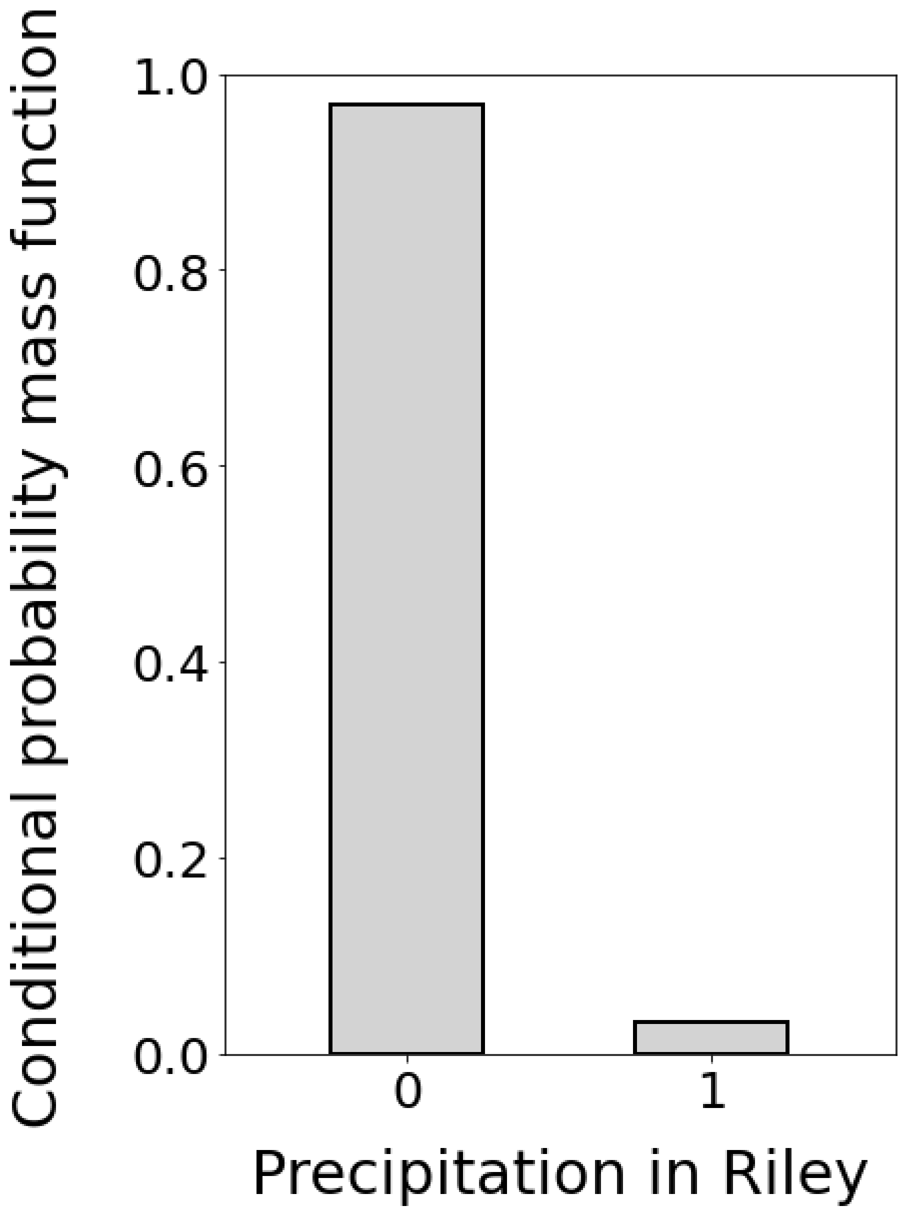

Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Coos Bay

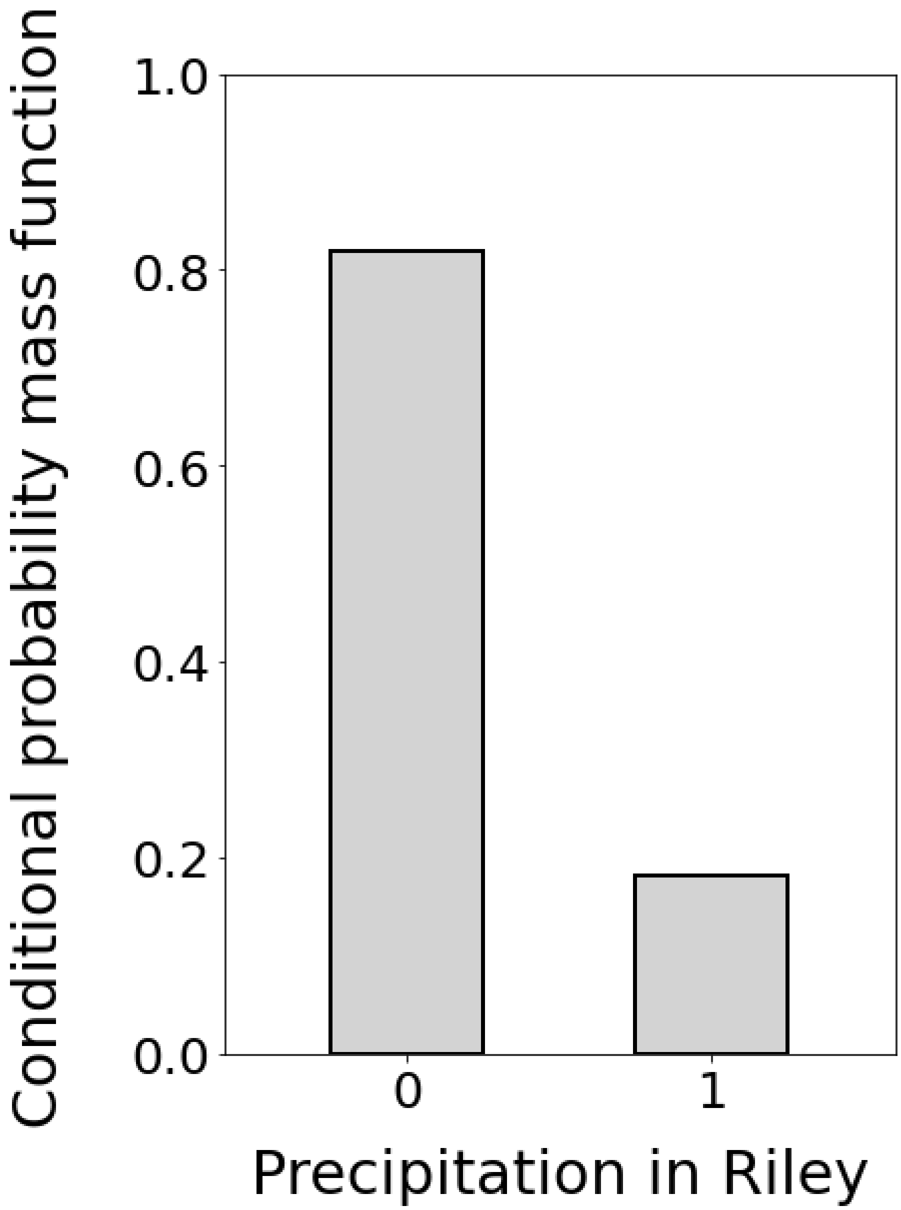


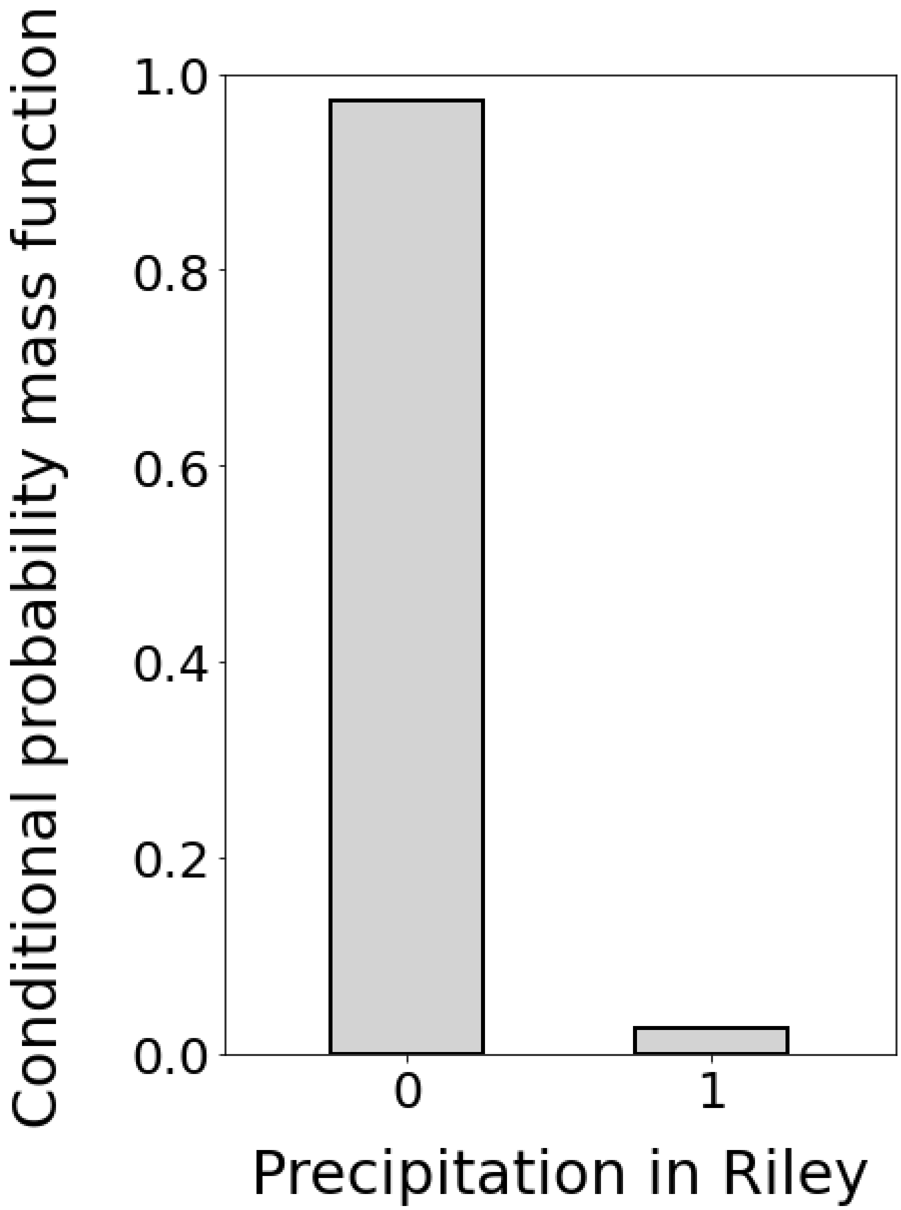
Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Coos Bay

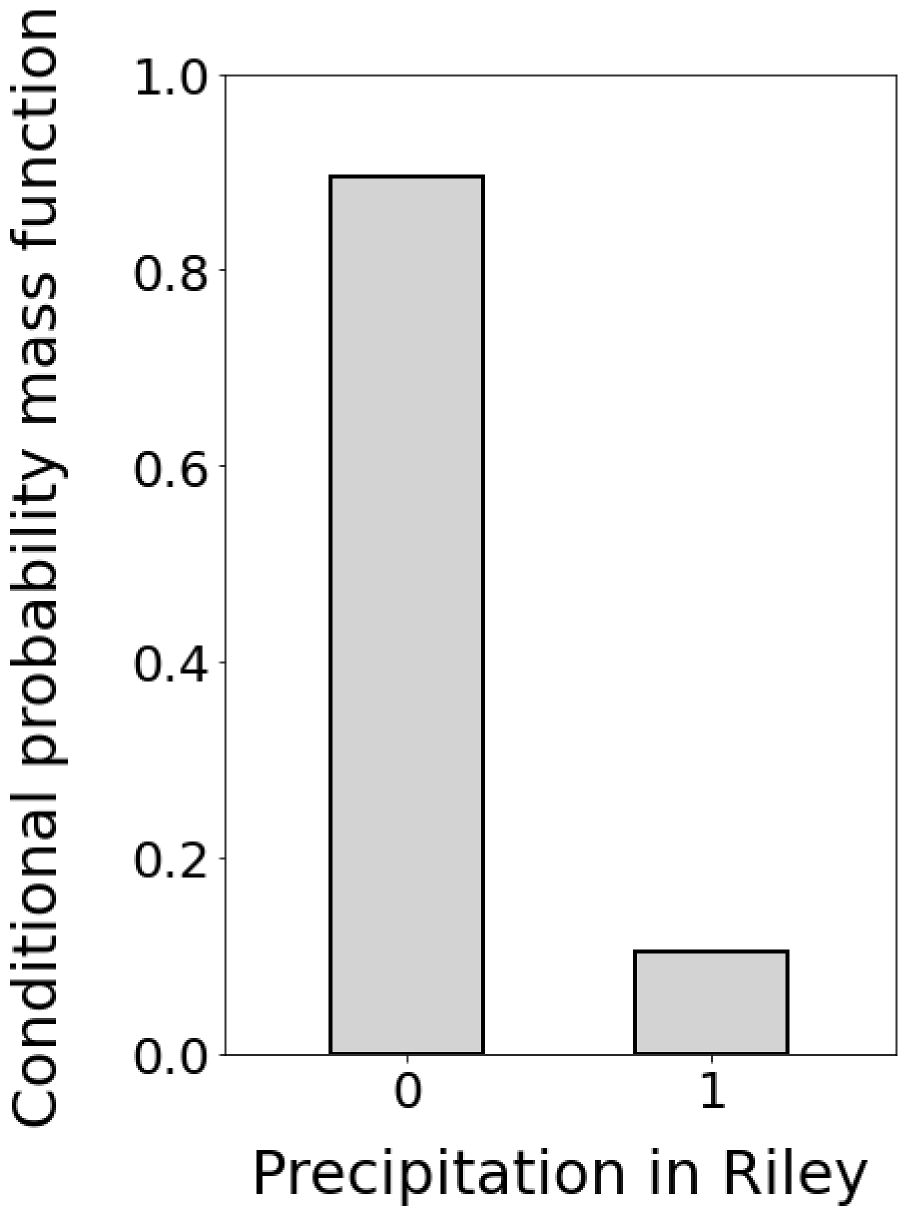

Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Coos Bay


Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Coos Bay


Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Riley


Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Riley


Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Riley


Conditional probability mass function
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Precipitation in Riley

```
In [ ]:
```

```
In [ ]:
```