Problem 1

徐逸洋 518370910020

(a).

In [1]:
```python
import math
import numpy as np
from scipy.special import gammaln

def likelihood(x,y):
    return math.exp(-1 * (x - y) * (x - y) / 2) * 1 / (math.pow(y, 2) + 1)

grid_size = 500
grid_center = []
grid_value = []
for i in range(grid_size):
    grid_center.append(1/(2 * grid_size) + (i * 1) / grid_size)

for i in range(grid_size):
    grid_value.append(likelihood(0.5, grid_center[i]))

product = sum(x * y for x, y in zip(grid_center, grid_value))
total = sum(grid_value)
expectation = product / total
print(expectation)
```

```
0.4431210359963634
```

(b)

(b) Complete the following table using your grid approximation.

Grid size n	50	250	500	1000
$E[Y\|X=0.5]$	0.443	0.443	0.443	0.443

(c) In [7]:
```python
import numpy as np

grid_size = 100
sample_size = 500

y = np.linspace(0, 1, 100)
likelihood = np.array([(np.exp(-(i - 0.5) * (i - 0.5)) / (1 + i * i) / 2) for i in y])
likelihood = likelihood / (likelihood.sum())
i = np.unravel_index(np.random.choice(f.size, size=sample_size, p=likelihood), likelihood.shape)
sample = y[i] + (y[1] - y[0]) * (np.random.rand() - 0.5)
print("Sample =")
print(sample)

e = sample.sum() / sample_size
print("Expection = ",e)
```

(d)

(d) Complete the following table using your grid approximation.

Grid size n	50		250		500		1000	
Sample size k	100	1000	100	1000	100	1000	100	1000
$E[Y\|X=0.5]$	0.473	0.448	0.435	0.430	0.431	0.416	0.436	0.447

HW3-P2

November 9, 2021

[1]:
```
import numpy as np
from scipy import stats
from scipy.special import gammaln as gml
import matplotlib.pyplot as plt
%matplotlib inline
```

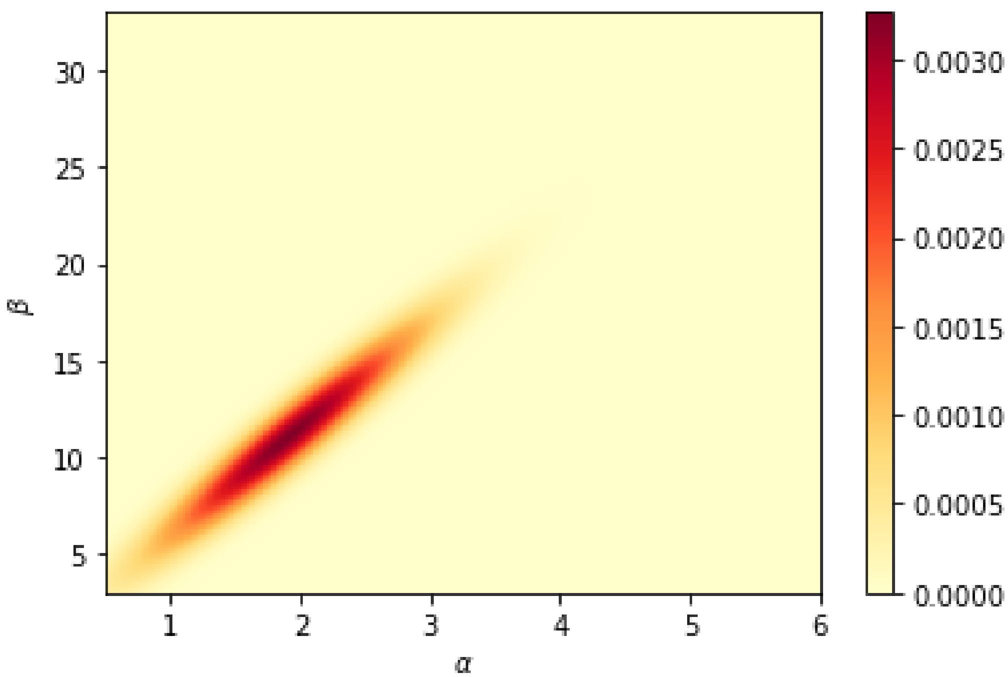
[2]:
```
# data from (BDA3, p. 102)
y = np.array([
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1,
    1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 5, 2,
    5, 3, 2, 7, 7, 3, 3, 2, 9, 10, 4, 4, 4, 4, 4, 4, 4,
    10, 4, 4, 4, 5, 11, 12, 5, 5, 6, 5, 6, 6, 6, 6, 16, 15,
    15, 9, 4
])
n = np.array([
    20, 20, 20, 20, 20, 20, 20, 19, 19, 19, 19, 18, 18, 17, 20, 20, 20,
    20, 19, 19, 18, 18, 25, 24, 23, 20, 20, 20, 20, 20, 20, 10, 49, 19,
    46, 27, 17, 49, 47, 20, 20, 13, 48, 50, 20, 20, 20, 20, 20, 20, 20,
    48, 19, 19, 19, 22, 46, 49, 20, 20, 23, 19, 22, 20, 20, 20, 52, 46,
    47, 24, 14
])
```

[3]:
```
A = np.linspace(0.5, 6, 100)
B = np.linspace(3, 33, 100)

LP = (-5/2 * np.log(A + B[:,None]) + np.sum(gml(A + B[:,None]) - gml(A) -␣
↪gml(B[:,None]) + gml(A + y[:,None,None]) + gml(B[:,None] + (n - y)[:
↪,None,None]) - gml(A + B[:,None] + n[:,None,None]), axis=0))
LP -= LP.max()
p = np.exp(LP)
p /= p.sum()

# Rejection sampling
x = []
y = []
g_Mu = [11, 1.9]
g_Cov = [[16, 2.7], [2.7, 0.5]]
```

```
g = np.zeros([B.shape[0], A.shape[0]])
for j in range(B.shape[0]):
    for i in range(A.shape[0]):
        temp = [B[j], A[i]]
        x.append(B[j])
        y.append(A[i])
        g[j, i] = stats.multivariate_normal.pdf(temp, g_Mu, g_Cov)

g /= g.sum()
```

[4]:

```
plt.imshow(g, origin='lower', aspect='auto', extent=(A[0], A[-1], B[0], B[-1]),␣
 ↪cmap = 'YlOrRd')
plt.xlabel(r'$\alpha$')
plt.ylabel(r'$\beta$')
# plt.grid('off')
plt.colorbar()
plt.show()
```

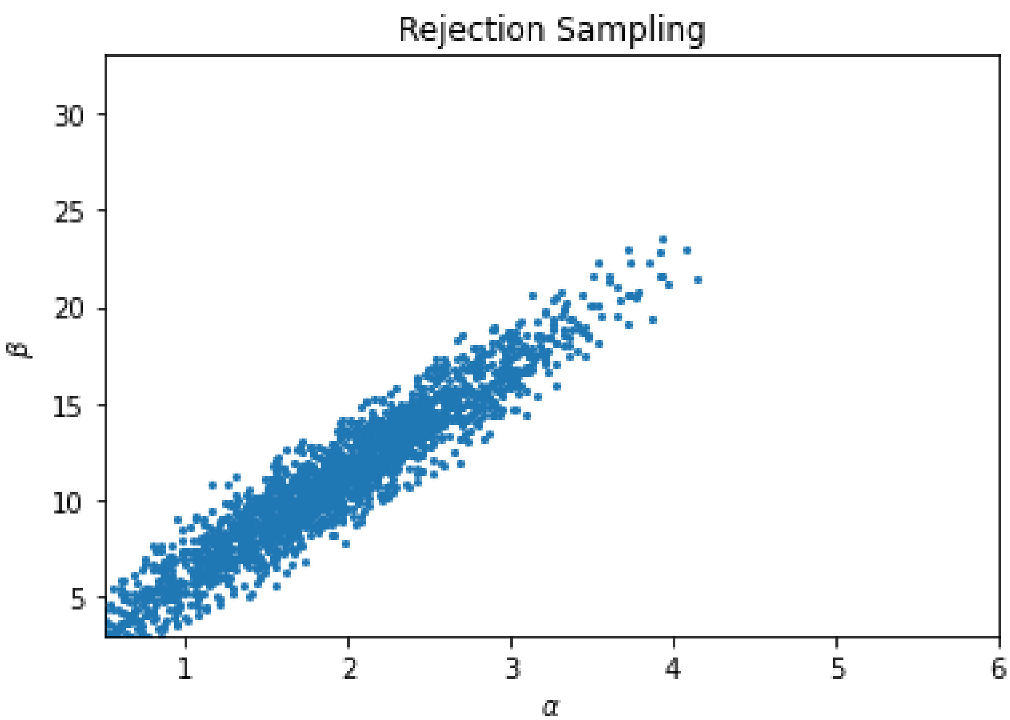


[5]:

```
M = np.max(p/g)
g *= M

sample_size = 2000
sam_x, sam_y = np.random.multivariate_normal(size=sample_size, mean=g_Mu,␣
 ↪cov=g_Cov).T
```

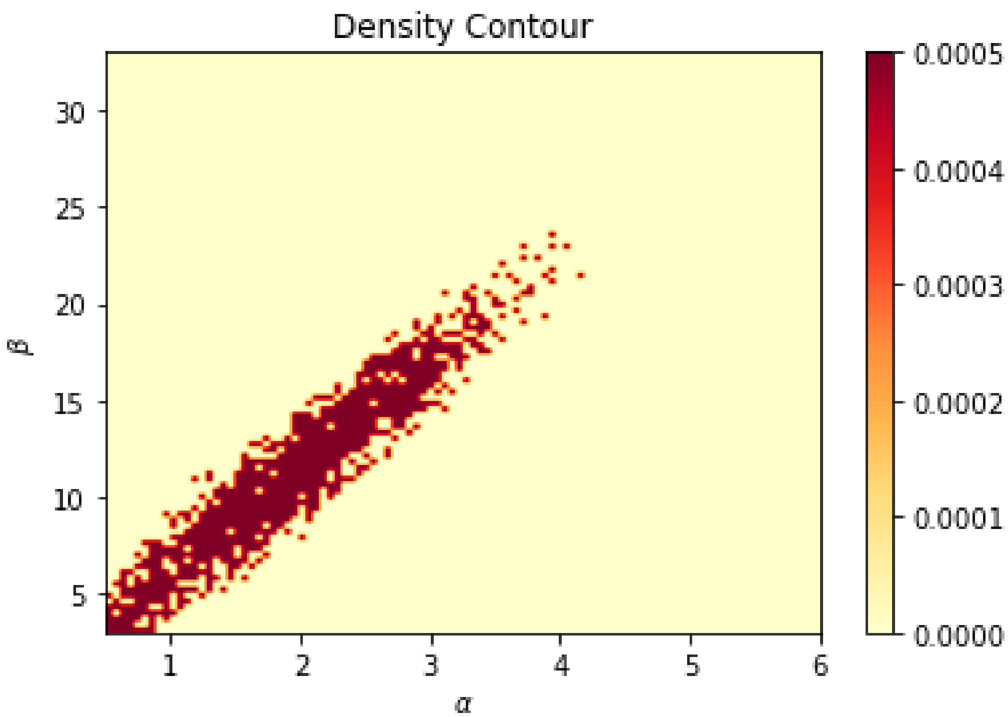
```
X = np.asarray(x)
Y = np.asarray(y)

idx = np.argmin((X[:, None] - sam_x) ** 2 + (Y[:, None] - sam_y) ** 2, axis=0)
posB, posA = divmod(idx, A.shape[0])
acc = np.random.rand(sample_size) * g[posB, posA]
acc = acc < p[posB, posA]
```

[6]:
```
sampleA = A[posA]
sampleB = B[posB]
sampleA += (np.random.rand(sample_size) - 0.5) * (A[1]-A[0])
sampleB += (np.random.rand(sample_size) - 0.5) * (B[1]-B[0])

plt.scatter(sampleA, sampleB, 10, linewidth=0)
plt.xlim([A[0], A[-1]])
plt.ylim([B[0], B[-1]])
plt.xlabel(r'$\alpha$')
plt.ylabel(r'$\beta$')
plt.title('Rejection Sampling')
plt.show()
```



[7]:
```
m = np.zeros([B.shape[0], A.shape[0]])
samp_A = A[posA]
```

```
samp_B = B[posB]
m[posB, posA] = 1
m /= len(posA)
plt.imshow(m, origin='lower', aspect='auto', extent=(A[0], A[-1], B[0], B[-1]),␣
 ↪cmap = 'YlOrRd')
plt.xlabel(r'$\alpha$')
plt.ylabel(r'$\beta$')
plt.title('Density Contour')
plt.colorbar()
plt.show()
```



Problem 3

Gibbs sampling is a special case of Metropolis-Hastings sampling in which the acceptance probability is 1.

In Metropolis-Hastings sampling,

acceptance ratio $r = \frac{P(\theta^* | y) \cdot J(\theta^{t-1} | \theta^*)}{P(\theta^{t-1} | y) \cdot J(\theta^* | \theta^{t-1})}$ where $J(\theta^* | \theta^{t-1})$ is the proposal distribution. In Gibbs sampling, $J(\theta^* | \theta^{t-1}) = P(\theta^* | y)$, meaning that the proposal distribution is the same as the conditional distribution

$$\Rightarrow r = \frac{P(\theta^* | y) \cdot P(\theta^{t-1} | y)}{P(\theta^{t-1} | y) \cdot P(\theta^* | y)} = 1 \quad \text{Acceptance probability} = \min(1, r) = 1$$

Problem 4.

(a) $P(y_1 = \text{white}, y_2 = \text{black}) = \frac{1}{2} \times \frac{1}{2} = P(y_1 = \text{black}, y_2 = \text{white})$ (i) Therefore, observations y_1 and y_2 are exchangeable. Also, $P(y_1, y_2) = P(y_1) \cdot P(y_2) = \frac{1}{2} \times \frac{1}{2}$ (ii) Therefore, y_1 and y_2 are independent.

(iii) We can act as if they are independent.

(b) (i) Suppose we have k white balls. $P(y_1 = \text{white}, y_2 = \text{black}) = \frac{k}{n} \cdot \frac{n-k}{n-1}$,

$P(y_1 = \text{black}, y_2 = \text{white}) = \frac{n-k}{n} \cdot \frac{k}{n-1} = \frac{k(n-k)}{n(n-1)} = P(y_1 = \text{white}, y_2 = \text{black})$, so y_1, y_2 are exchangeable.

(ii) $P(y_1 = \text{black}, y_2 = \text{white}) = \frac{k(n-k)}{n(n-1)} \neq P(y_1 = \text{black}) \cdot P(y_2 = \text{white}) = \frac{n-k}{n} \cdot \frac{k}{n}$, so y_1, y_2 are not independent.

(iii) When n is large enough, $n(n-1)$ is close to n^2, we can act as if they are independent. Otherwise we can't.

(C) (i) Similar to (b), $P(y_1 = \text{white}, y_2 = \text{black}) = P(y_1 = \text{black}, y_2 = \text{white})$, so y_1, y_2 are exchangeable.

(ii) $P(y_1 = \text{black}, y_2 = \text{white}) \neq P(y_1 = \text{black}) \cdot P(y_2 = \text{white})$, so y_1, y_2 are not independent.

(iii) Since there are many balls, term $n(n-1)$ can be approximated as n^2, so we can act as if they are dependent.

Problem 5.

5. Mixtures of independent distributions: suppose the distribution of $\theta = (\theta_1, \dots, \theta_J)$ can be written as a mixture of independent and identically distributed components:

$$p(\theta) = \int \prod_{j=1}^{J} p(\theta_j|\phi)p(\phi)d\phi.$$

Prove that the covariances $\text{cov}(\theta_i, \theta_j)$ are all nonnegative.

$$\text{cov}(\theta_i, \theta_j) = E(\theta_i\theta_j) - E(\theta_i)E(\theta_j)$$

$$= E[E(\theta_i\theta_j|\phi)] - E[E(\theta_i|\phi)] \cdot E[E(\theta_j|\phi)]$$

$$= E[E(\theta_i\theta_j|\phi) - E(\theta_i|\phi)E(\theta_j|\phi)] + E[E(\theta_i|\phi)E(\theta_j|\phi)] - E[E(\theta_i|\phi)]E[E(\theta_j|\phi)]$$

$$= E[\text{cov}(\theta_i, \theta_j|\phi)] + \text{cov}[E(\theta_i|\phi), E(\theta_j|\phi)]$$

Since $E(\theta_i|\phi) = E(\theta_j|\phi)$, $\text{cov}[E(\theta_i|\phi), E(\theta_j|\phi)] = \text{Var}[E(\theta_i|\phi)]$

$\text{Var}[E(\theta_i|\phi)] \geq 0$ also, since $\forall i, j$, θ_i, θ_j are independent and identical, so $E[\text{cov}(\theta_i, \theta_j|\phi)] = 0 \Rightarrow \text{cov}(\theta_i, \theta_j) = 0 + \text{Var}[E(\theta_i|\phi)] \geq 0.$