Markov chain Monte Carlo

For this problem, you will be implementing a hierarchical Bayesian model and using a Markov chain Monte Carlo technique (Gibbs sampling) to approximate its predictions. In particular, you will implement the Beta-binomial model with an exponential prior used in Kemp et al. (2007) to understand how people learn {\em overhypotheses} --- beliefs about how properties tend to be distributed over different situations. So, the Beta distribution is prominent in their model, which is a probability distribution over values between 0 and 1. The generative model for Beta-Binomial model with an exponential prior is

$$\begin{aligned} \alpha &\sim \text{Exponential}(1) & \beta &\sim \text{Beta}(1,1) \\ \theta_i|\alpha,\beta &\sim \text{Beta}\left(\alpha\beta, \alpha(1-\beta)\right) & y_i|n_i,\theta_i &\sim \text{Binomial}(\theta_i; n_i) \end{aligned}$$

Tip: Most programming languages provide functions for generating Exponential and Beta distributed random variables.

Remember their cover story. Imagine that you come across a box filled with bags of marbles. Each bag has a lot of marbles. Each marble is either black or white. You get to observe n_i marbles from bag i and write down that y_i of them are white. The proportion of white marbles in bag i is θ_i (and because there are lots and lots of marbles in each bag, we can safely assume that y_i is Binomial distributed with parameters θ_i and n_i).

2a. Beta distributions

First, you will estimate the posterior distribution of a simplified version of the model: The posterior probability of the proportion of white marbles after seeing some marbles from a single bag without the hierarchical priors. So, the generative process for this part is: $\theta_i|\alpha,\beta \sim \text{Beta}\left(\alpha\beta, \alpha(1-\beta)\right)$ and $y_i|n_i,\theta_i \sim \text{Binomial}(\theta_i; n_i)$.

You can look up information about the Beta distribution on the Conjugate Prior page of Wikipedia.

Note that the notation here follows the notation used by Kemp et al. (2007). Wikipedia uses a different notation, which is actually the more standard one in statistics.}.

The posterior distribution for a Beta-Binomial model is $\text{Beta}\left(\alpha\beta + y_i, \alpha(1-\beta) + (n_i - y_i)\right)$. This is the equation you need to complete this task.

For each α and β value, please graph:

1. the prior distribution, (i.e. , no marbles observed) the posterior after observing one white marble (and zero black marbles)
2. the posterior distribution after observing 5 white and 5 black marbles,
3. and the posterior distribution after observing 9 white and 1 black marble all on the same plot.

I provide the a and b values as well (which is usually what most programming languages expect) and correspond to the standard/Wikipedia parameterization of the Beta distribution).

Please explore the following Beta distributions:

1. $\alpha = 0.1$ and $\beta = 0.5$ ($a = 0.05, b = 0.05$)

2. $\alpha = 0.1$ and $\beta = 0.9$ $(a = 0.09, b = 0.01)$
3. $\alpha = 1$ and $\beta = 0.5$ $(a = 0.5, b = 0.5)$
4. $\alpha = 1$ and $\beta = 0.9$ $(a = 0.9, b = 0.1)$
5. $\alpha = 5$ and $\beta = 0.5$ $(a = 2.5, b = 2.5)$
6. $\alpha = 5$ and $\beta = 0.9$ $(a = 4.5, b = 0.5)$

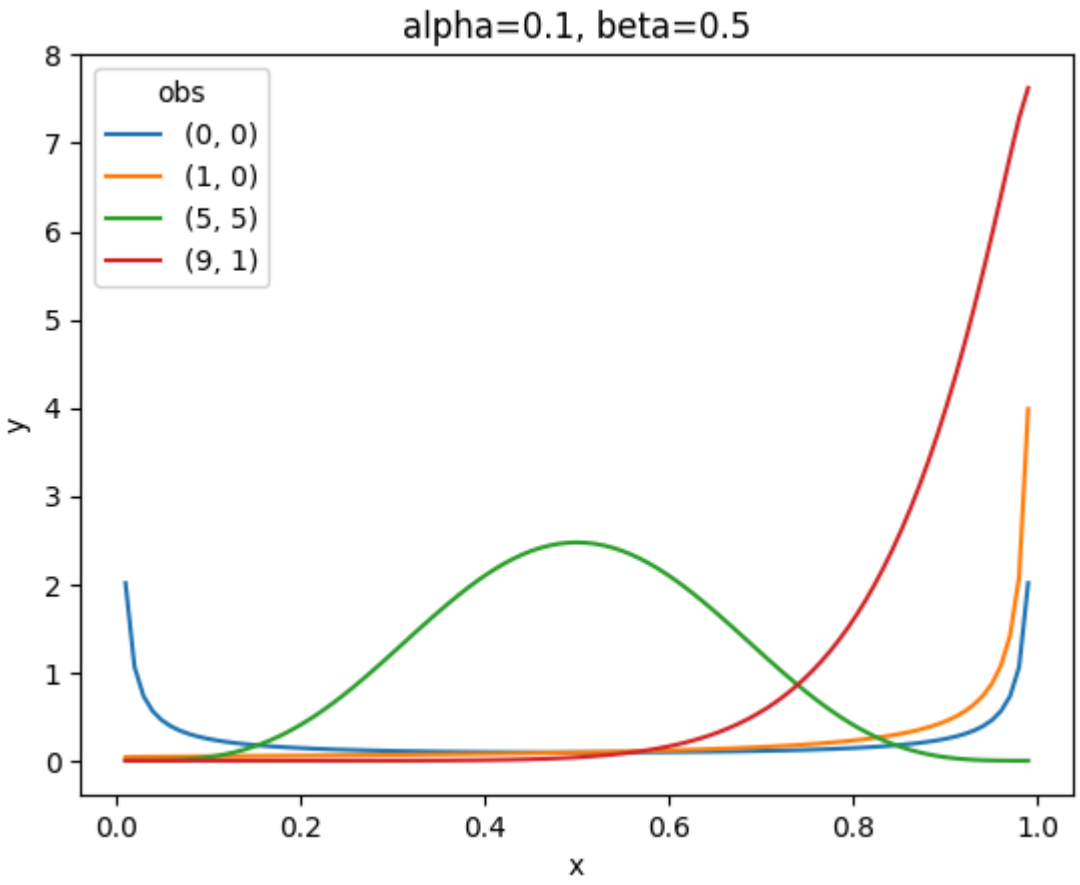
Although I expect you to execute and think about each of these, you are only required to turn in one of the graphs. Please turn in make sure that the graph is visible and each distribution is distinct and clearly labelled. Please write a paragraph describing how changing α and β affect the posterior update.

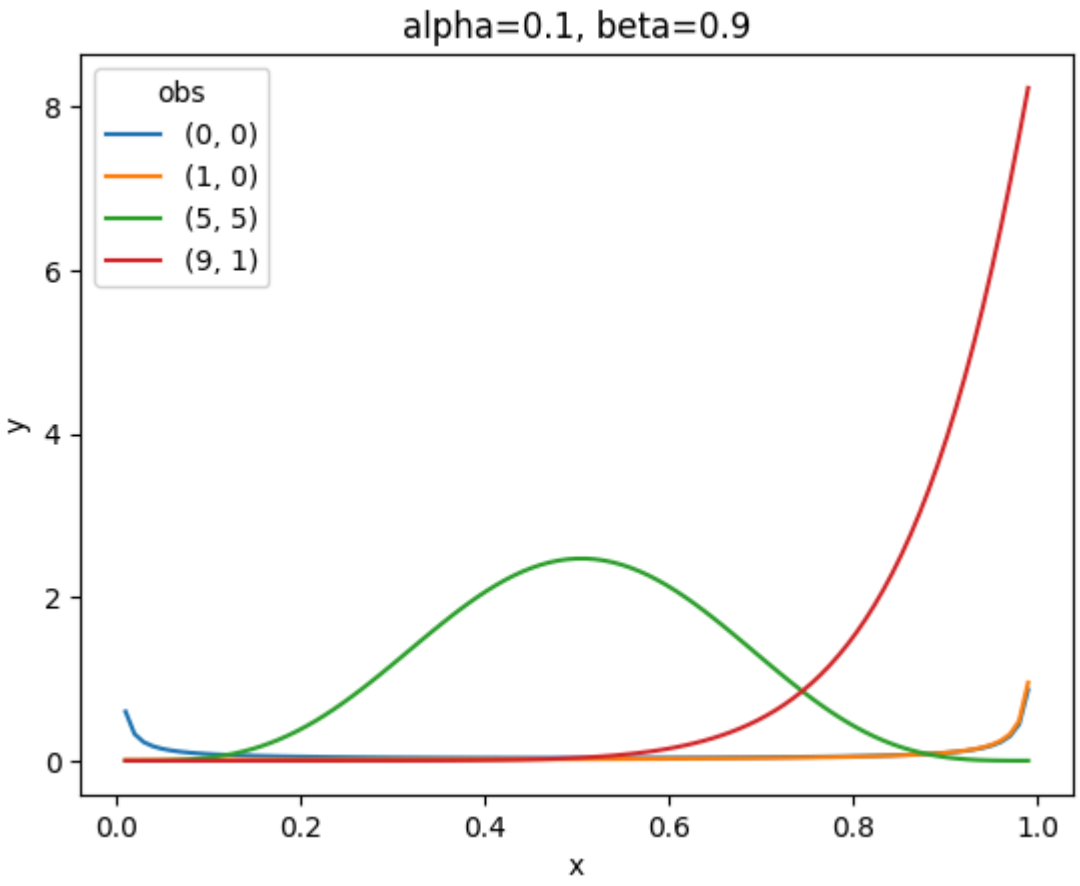
In [1]:
```
# distributions when parametrizing with alpha and beta
distributions_orig = [
    (0.1, 0.5), # alpha = 0.1, beta = 0.5
    (0.1, 0.9),
    (1.0, 0.5),
    (1.0, 0.9),
    (5.0, 0.5),
    (5.0, 0.9),
]

# distributions when parametrizing with a and b (standard)
distributions_wiki = [
    (0.05, 0.05), # a = 0.05, b = 0.05
    (0.09, 0.01),
    (0.50, 0.50),
    (0.90, 0.10),
    (2.50, 2.50),
    (4.50, 0.50),
]

num_observed = [
    (0, 0), # 0 white, 0 black <-- prior distribution!
    (1, 0), # 1 white, 0 black
    (5, 5),
    (9, 1),
]
```

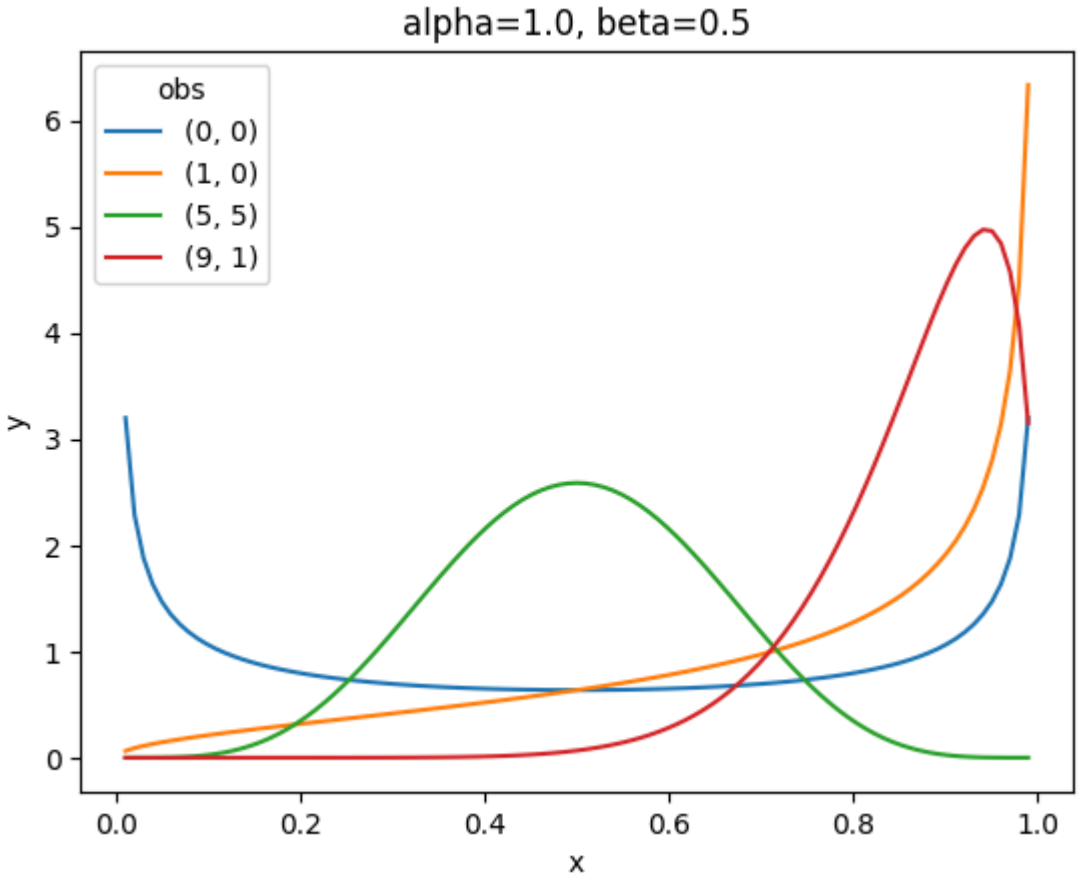
In [2]:
```
import scipy.stats
from scipy.stats import beta
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sb
import pandas as pd
```

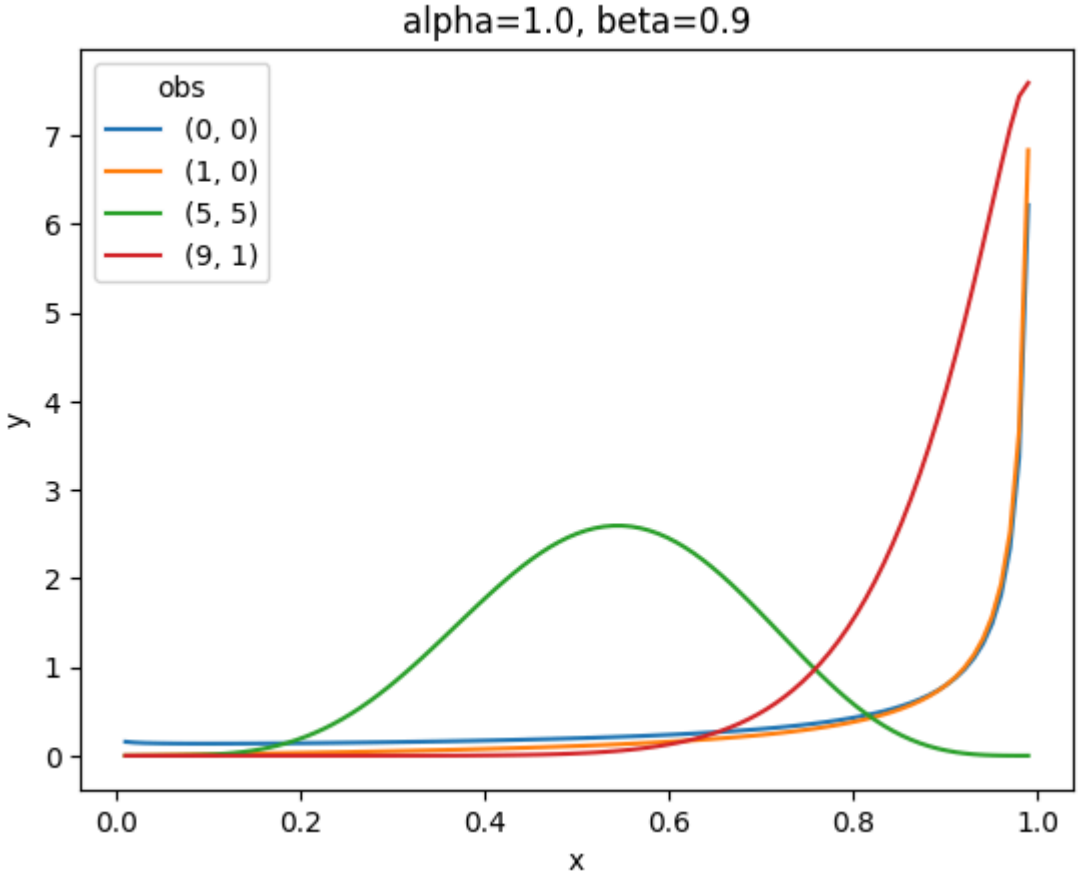
In [3]:
```
for (Alpha, Beta), (a, b) in zip(distributions_orig, distributions_wiki):
    # Beta roughly corresponds to mean
    # Alpha is the "sample size"

    dfs = []
    x = np.linspace(start=0.01, stop=1-0.01, num=101)
    for observed in num_observed:
        obs_wht, obs_blk = observed
        n = obs_wht + obs_blk
        a_obs = a + obs_wht
        b_obs = b + (n - obs_wht)
        probs = beta.pdf(x, a_obs, b_obs)
        df = pd.DataFrame(data={'x': x, 'y': probs, 'obs': str(observed)})
        dfs.append(df)
        pass
```

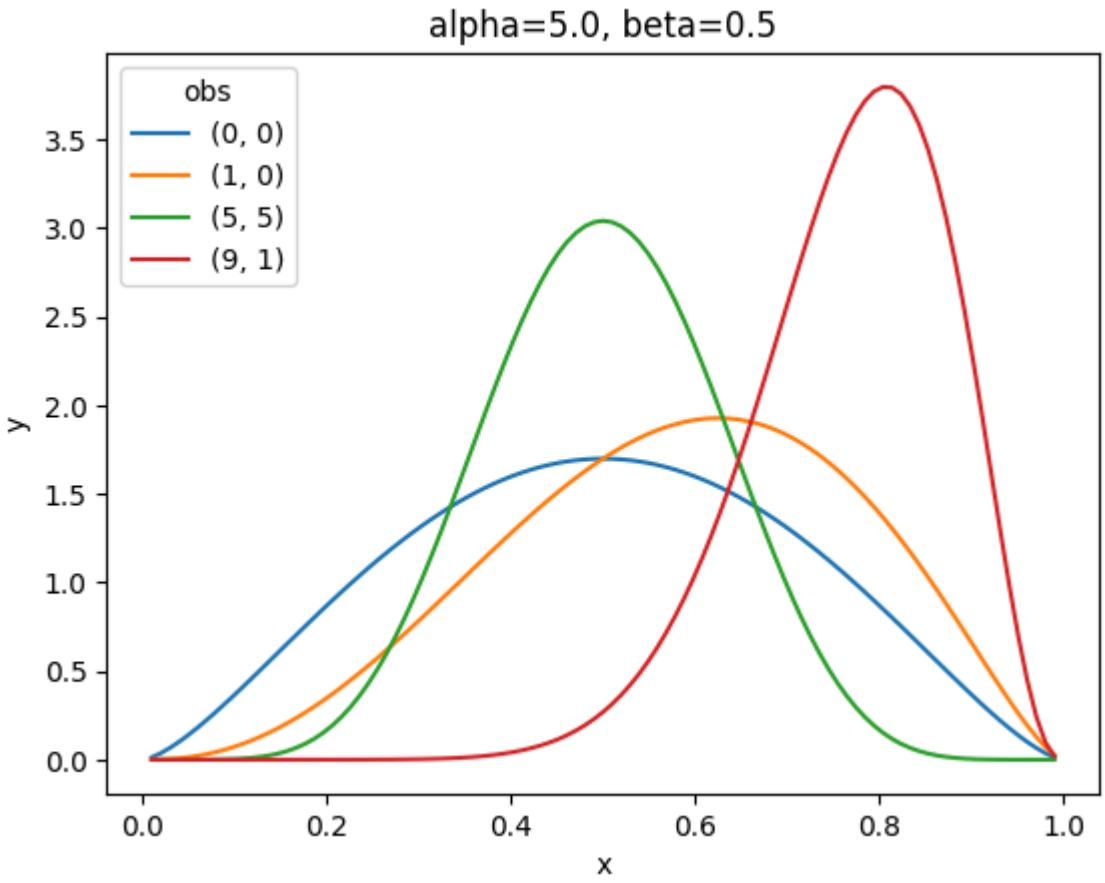
```
    dfs_concat = pd.concat(dfs)
    sb.lineplot(data=dfs_concat, x='x', y='y', hue='obs')
    plt.title(f'alpha={Alpha}, beta={Beta}')
    plt.figure()
```

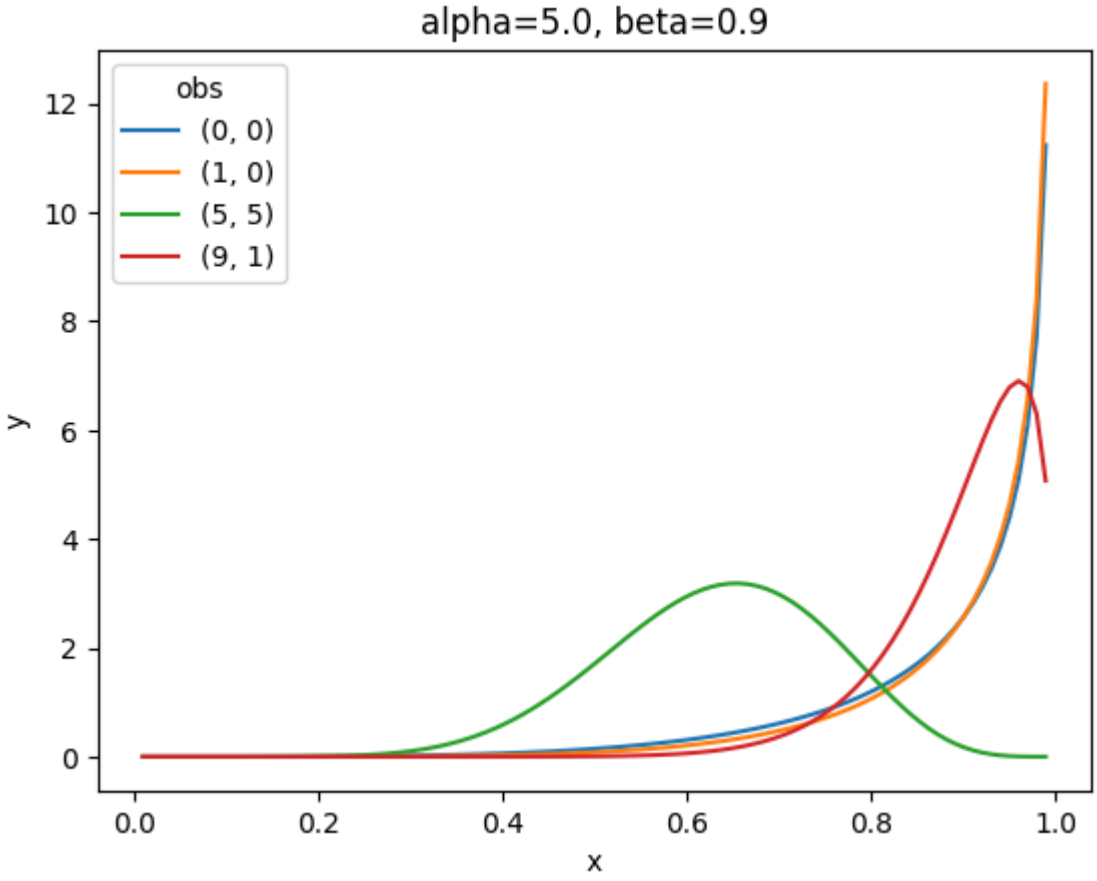





alpha=1.0, beta=0.5
obs
(0, 0)
(1, 0)
(5, 5)
(9, 1)
y
x
0
1
2
3
4
5
6
0.0
0.2
0.4
0.6
0.8
1.0


alpha=1.0, beta=0.9
obs
(0, 0)
(1, 0)
(5, 5)
(9, 1)
y
x
0
1
2
3
4
5
6
7
0.0
0.2
0.4
0.6
0.8
1.0






```
<Figure size 640x480 with 0 Axes>
```

The Beta parameter places a prior on the mean of the curve.

Alpha, as described on Wikipedia, is the "sample size" parameter. The higher it is, the more samples are required to more the probability mass of the distribution away from the mean.

2b. Implementing a MCMC sampler

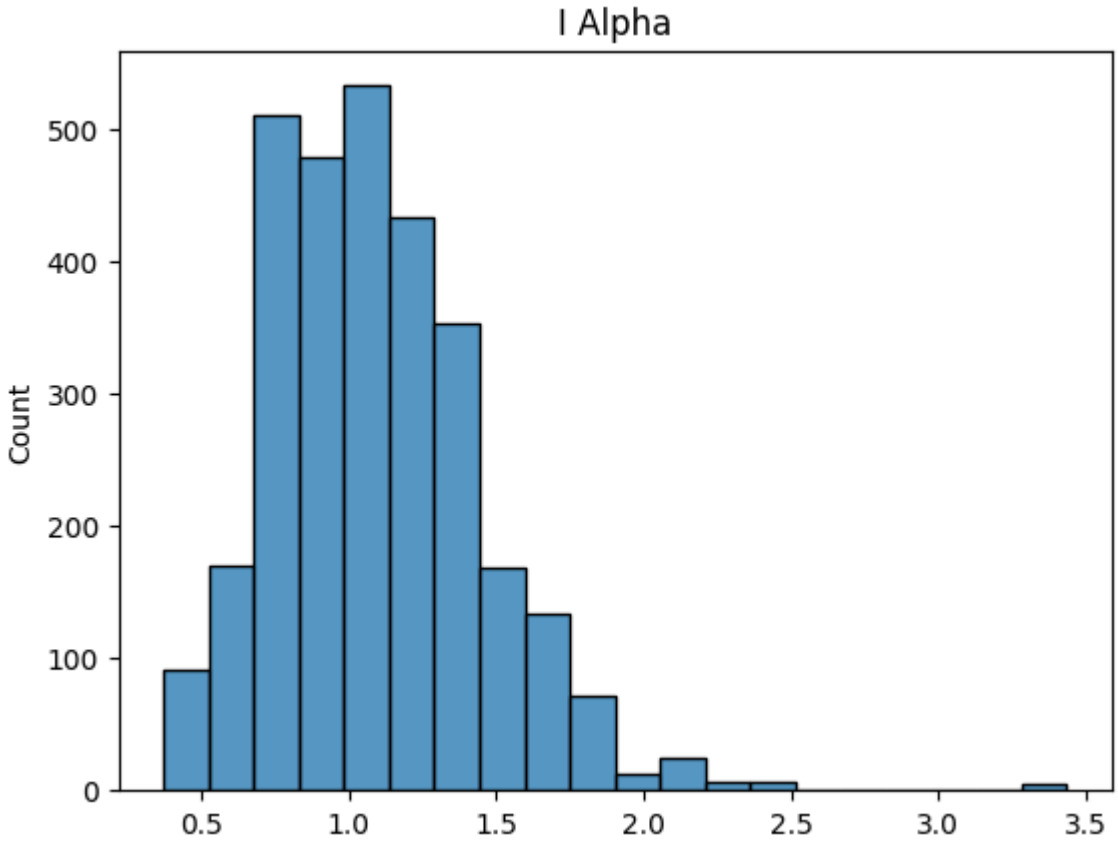
Now you will implement samplers to estimate the posterior distribution on α, β, and $\theta_1, \ldots \theta_M$ after observing some marbles from M bags (i.e., given $y_1, \ldots, y_M, n_1, \ldots, n_M$) and the predictive probability for the probability of white marbles in new bags after observing some marbles from some bags (i.e., given $y_1, \ldots, y_M, n_1, \ldots, n_M$). We will be approximating the posterior distribution of $\theta_1 \ldots, \theta_M$ using Gibbs sampling steps and interleaving these steps with Metropolis-Hastings Markov chain Monte Carlo (MH MCMC) sampling for α and β.

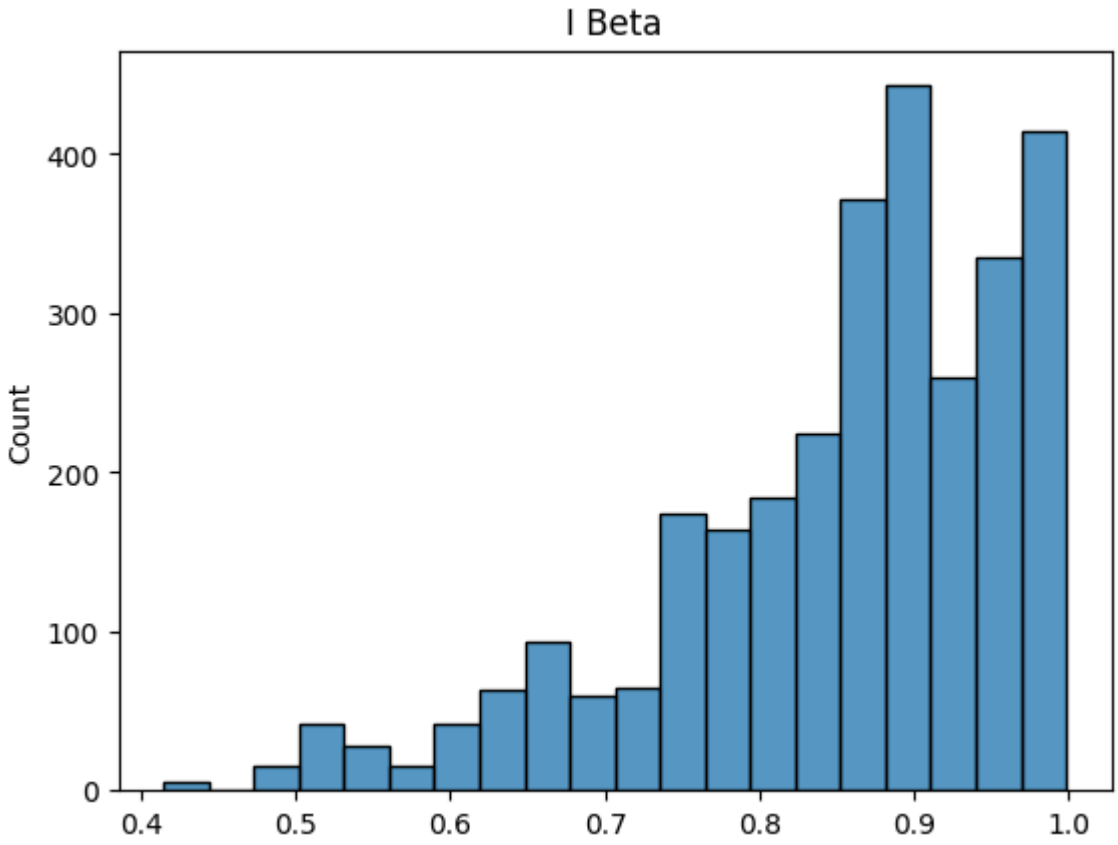
Here is an overview of the process:

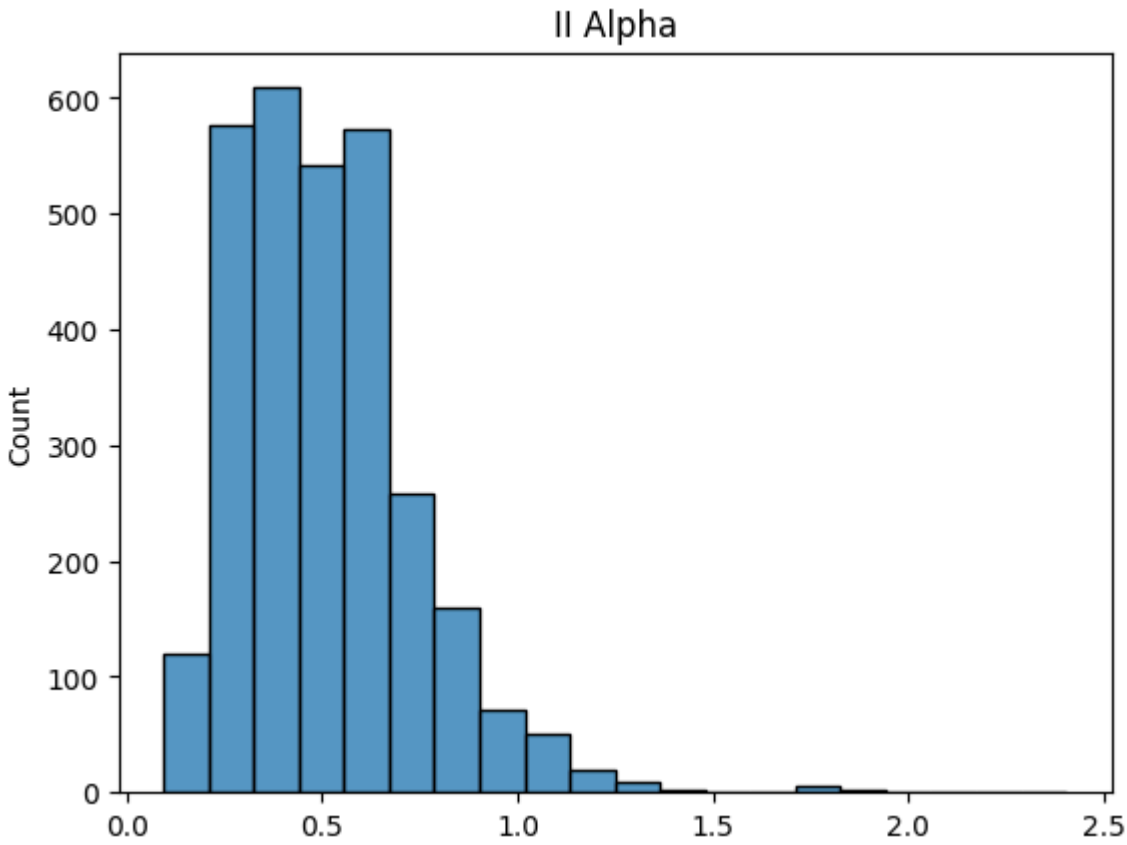
1. Initialize α and β by sampling them from their priors (Exponential(1) and Beta(1,1), respectively).
2. Sample a value for each θ_m of the M bags based on current α and β.
3. Sample new proposal values for α and β given the M sampled θ values via an equation provided in the detailed procedure below.
4. Accept the proposed α and β, or to keep the previous values.
5. Repeat from 2 until convergence

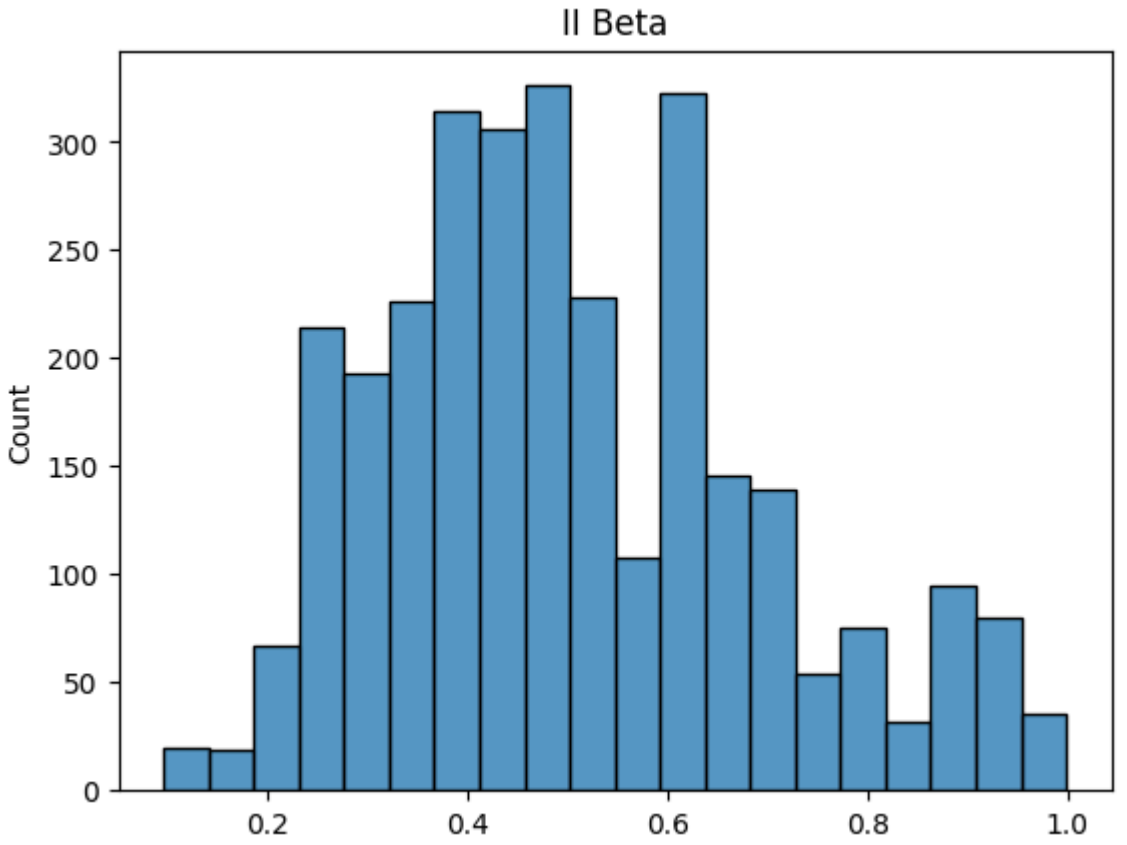
In [11]:
```
def pdf_Theta_given_Alpha_Beta(Theta_new, Alpha_cur, Beta_cur):
    return scipy.stats.beta.pdf(Theta_new, Alpha_cur, Beta_cur)

def pdf_A(Alpha_new):
    return scipy.stats.expon.pdf(Alpha_new)

def pdf_B_given_B(Beta_new, Beta_cur):
    return scipy.stats.beta.pdf(Beta_new, 1 + Beta_cur, 2 - Beta_cur)

def pdf_B(Beta_new):
    return scipy.stats.beta.pdf(Beta_new, 1, 1)

def sampler(bag_contents, T=3000):
    # 0 initialize alpha and beta
    generator = np.random.default_rng(seed=42)
    a = generator.exponential(scale=1)

    b = generator.beta(a=1, b=1)

    samples = []
    for idx_sweep in range(T):
        # 1 sample a Theta_m for each of the M bags based on the current Alpha and Be
        obs_wht = bag_contents[:,0]
        obs_blk = bag_contents[:,1]
        num_observed = obs_wht + obs_blk
        # display(obs_wht, num_observed)
        a_obs = a + obs_wht
        b_obs = b + (num_observed - obs_wht)
        M = 10
        Thetas = generator.beta(a=a_obs, b=b_obs, size=M)
        # display(Thetas)

        # 2. Generate a proposed next state for the sampler using the current state.
        a_cand = a + generator.standard_t(df=1)
        b_cand = generator.beta(a=1+b, b=2-b)
        # display(Alpha_candidate, Beta_candidate)

        # 3. Calculate the probability that we accept the candidate Alpha and Beta as
        pa = pdf_A(a)
        pb = pdf_B(b)
        pa_new = pdf_A(a_cand)
        pb_new = pdf_B(b_cand)
        # print(f'Alpha,Beta={(a, b)}')
```

```
        # print(f'pa,pb={(pa,pb)}, pa_new,pb_new={pa_new, pb_new}')

        p_theta_new = np.prod(pdf_Theta_given_Alpha_Beta(Thetas, a_cand, b_cand))
        numerator = pdf_B_given_B(b, b_cand) * pa_new * pb_new * p_theta_new

        p_theta_cur = np.prod(pdf_Theta_given_Alpha_Beta(Thetas, a, b))
        denominator = pdf_B_given_B(b_cand, b) * pa * pb * p_theta_cur

        # print(p_theta_new, p_theta_cur)

        c = numerator / denominator

        Y = generator.random()
        # print(Y, c)
        if Y <= c:
            a, b = a_cand, b_cand
        samples.append((a, b))
    return np.array(samples)

# The contents of the bags as specified in I and II
contents_I = np.empty(shape=(10,2))
contents_I[:] = np.array([9, 11])

contents_II = np.empty(shape=(10,2))
contents_II[:5] = np.array([1, 19])
contents_II[5:] = np.array([19, 1])

contents = [contents_I, contents_II]

# Finally, run the samper on each of the contents
samples_mult = []
for desc, content in zip(['I', 'II'], contents):
    samples = sampler(bag_contents=content, T=3000)
    samples_mult.append(samples)

    for idx, typ in [(0, 'Alpha'), (1, 'Beta')]:
        plt.figure()
        sb.histplot(x=samples[:,idx], bins=20)
        plt.title(f'{desc} {typ}')
```


I Alpha
Count
0
100
200
300
400
500
0.5
1.0
1.5
2.0
2.5
3.0
3.5


I Beta
Count
0
100
200
300
400
0.4
0.5
0.6
0.7
0.8
0.9
1.0





In [14]:
```
for typ, samples in zip(['I', 'II'], samples_mult):
    a_mean, b_mean = np.mean(samples, axis=0)
    expected_theta_new = (a_mean * b_mean) / (a_mean * b_mean + a_mean*(1-b_mean))
    print(f'For section {typ},')
    print(f'Alpha mean = {a_mean}, Beta mean = {b_mean}, Expected Theta = {expected_t
```

```
For section I,
Alpha mean = 1.0928421697490411, Beta mean = 0.851012274303348, Expected Theta = 0.851
012274303348
For section II,
Alpha mean = 0.5104616701891956, Beta mean = 0.5057442713703126, Expected Theta = 0.50
57442713703126
```

Paragraphs

I know something went wrong because the mean for the first set of marble bags, the homogeneous ones, should have been close to 9/20 with tight variance. I was not able to pinpoint the bug in the code, unfortunately. I write my predictions here for what I would expect to see from correctly-running code.

For the homogeneous set of marble bags, Beta tended toward one end. Particularly, Beta hovered around 9/20, since that is the Theta most commonly exhibited by the bags in the set. Moreover, Alpha was low. This makes intuitive sense because there was little variance in the makeup of marble bags, hence the Theta underlying the bags would not be very different from each other.

For the heterogeneous set of marble bags, Beta should have had observations toward both ends of the range (0 and 1), as the thetas most commonly exhibited by the marble bags were near 0 and 1. Alpha should be high, as most samples of B would be found near the ends of the range. Intuitively, bags had very different Thetas from each other.

The mean for Betas across marble bag sets should be quite close! This is because the homogenous bags ae scarcely different from a theta of 0.5 and the heterogeneous bags had much variance but the number of the two types of bag was evenly distributed relative to Beta = 0.5.

The mean for Alphas across marble bag sets should be different. The homogeneous set is, by definition, homogeneous. Variance is naturally low. The heterogeneous set will be best fit by a higher Alpha, something which will not change by being averaged.

End of paragraphs

To implement the Gibbs sampling portion, we iterate through the bags and for each bag, we resample the current value for the proportion of white marbles in bag i ($\theta_i^{(t)}$) given the value of all other variables (so, given $\theta_1^{(t-1)}, \ldots, \theta_{i-1}^{(t-1)}, \ldots, \theta_{i+1}^{(t-1)}, \ldots, \theta_M^{(t-1)}, \alpha^{(t)}, \beta^{(t)}, \mathbf{y}, \mathbf{n}$). Given α and β all of the bags are independent of each other (if you're not sure, use the conditional independence rules for graphical models we discussed in class), and so, we sample $\theta_i^{(t)}|\alpha^{(t)}, \beta^{(t)}, y_i, n_i$. Fortunately, we already know this from the last subproblem! It's a Beta-Binomial model and so

$$\theta_i^{(t)}|\alpha^{(t)}, \beta^{(t)}, y_i, n_i \sim \text{Beta}\left(\alpha^{(t)}\beta^{(t)} + y_i, \alpha^{(t)}(1-\beta^{(t)}) + (n_i - y_i)\right)$$

The sampler for α is a bit more tricky because an explicit solution for the posterior distribution of $\alpha|\theta_1, \ldots, \theta_M$ is unknown. To implement the MH MCMC, we successively generate new proposed values for α and β given its previous value. For α, we will use the Student's t distribution with one-degree of freedom and a mean centered at the current sample for α, $\alpha^{(t)}$ (under the constraint that $\alpha^{(t+1)} > 0$).

We do so that we normally propose to change α to values near the current one, but also occasionally propose very different values, which can help the Markov chain ``mix'' better. Student's t probability distribution with one degree of freedom is %next year make this clearer

$$f(x) = \frac{1}{\pi(1+x^2)}$$

So, our proposal distribution $q(\alpha'|\alpha^{(t)}) = (\alpha^{(t)} + X^{(t)})I(\alpha^{(t)} + X^{(t)} > 0)$, where $X \sim \text{Student's } t(1)$ (Note the $'$ is used to denote the proposed new value of the corresponding variable -- in this case α). Choosing a proposal distribution for β is simpler. It will be a Beta distribution whose mean is β_t and has a bit of variance. So, $q(\beta'|\beta^{(t)})$ is $\text{Beta}(1+\beta^{(t)}, 2-\beta^{(t)})$.

To implement MH MCMC, we first initialize the Markov chain by a random sample from the prior ($\alpha^{(1)} \sim \text{Exponential}(1)$ and $\beta^{(1)} \sim \text{Beta}(1,1)$) and then for $t = 1, \ldots, T$ follow the following steps:

1. Do a ``sweep'' (resample each $\theta_i^{(t)}$ once) of Gibbs sampling. You should sample the M bags in a random order each sweep.So, resample each θ_i:
   $\theta_i^{(t)}|\alpha^{(t)}, \beta^{(t)}, y_i, n_i \sim \text{Beta}\left(\alpha^{(t)}\beta^{(t)} + y_i, \alpha^{(t)}(1-\beta^{(t)}) + (n_i - y_i)\right)$
2. Generate a proposed next state for the sampler using the current state. To do so, sample α' by adding $X \sim \text{Student's } t(1)$ to $\alpha^{(t)}$ and sample $\beta'|\beta^{(t)} \sim \text{Beta}(1+\beta^{(t)}, 2-\beta^{(t)})$.
3. Calculate the probability c that we accept α' and β' as the next state. Because the proposal distribution for α is symmetric, we do not need to include it in our acceptance term. However, the proposal distribution for β is not symmetric, and so the MH requires a correction term to ensure that the probability that we move back and forth between states. So, the acceptance probability is the product of two probabilities: (1) the ratio of the probability of transitioning from the proposal back to the current state to the probability of the reverse transition and (2) the ratio of the posterior probability of the proposal to the posterior probability of the previous sample. Don't worry. I've done the derivation for you (it is straightforward, so it's a good exercise to do if you are curious!). It is

   $$\begin{aligned} c &= \frac{q(\beta^{(t)}|\beta')p(\alpha', \beta'|\theta_1^{(t)}, \ldots, \theta_M^{(t)})}{q(\beta'|\beta^{(t)})p(\alpha^{(t)}, \beta^{(t)}|\theta_1^{(t)}, \ldots, \theta_M^{(t)})} \\ &= \frac{\texttt{betapdf}(\beta^{(t)}; 1+\beta', 2-\beta')p(\alpha')p(\beta')\prod_i p\left(\theta_i^{(t)}|\alpha', \beta'\right)}{\texttt{betapdf}(\beta'; 1+\beta^{(t)}, 2-\beta^{(t)})p(\alpha^{(t)})p(\beta^{(t)})\prod_i p\left(\theta_i^{(t)}|\alpha^{(t)}, \beta^{(t)}\right)} \end{aligned}$$

   where `betapdf(x;y,z)` is the amount of density at x given by a Beta function with parameters y and z.
4. Sample $Y \sim U(0,1)$. If $Y \leq c$, $\left(\alpha^{(t+1)}, \beta^{(t+1)}\right) = (\alpha', \beta')$. Otherwise, reject and the next sample is the same as t: $\left(\alpha^{(t+1)}, \beta^{(t+1)}\right) = \left(\alpha^{(t)}, \beta^{(t)}\right)$

Now we use the sampler to explore the model's behavior given different sets of marble bags. Please use the following sets of marble bags:

1. 10 bags where each bag has 9 white and 11 black marbles.
2. 5 bags with 1 white and 19 black marbles, and 5 bags with 19 white and 1 black marble.

For each of these, please run the sampler for 3000 sweeps ($T = 3000$). Please include (two separate) histograms of the α samples and β samples for both I. and II. Also, calculate the average α and β value, and the expected value for a new bag of marbles (θ_{M+1}) after observing the bags of marbles for each (note that this is $\frac{\bar{\alpha}\bar{\beta}}{\bar{\alpha}\bar{\beta}+\bar{\alpha}(1-\bar{\beta})}$, where $\bar{\cdot}$ is the average of that variable).

Please write two paragraphs. In the first paragraph, describe the histograms for α and β given each set of marble bags and explain what their values mean. In the second paragraph, compare the histograms and the values you were asked to calculate. What is equal and unequal? Explain why.