P1

October 23, 2017

1 Problem 1

The likelihood for a coin flipping experiment will be defined by the binomial

$$\mathcal{L}(p) = \binom{n}{Y} p^Y (1-p)^{n-Y} \tag{1}$$

where - n: total number of flips - Y: Observed number of Heads - p: Probability of each flip being Heads

The test statistic, λ, is defined as

$$\lambda = \frac{\mathcal{L}(p_0)}{\mathcal{L}(\hat{p})} \tag{2}$$

Where $\hat{p}$ is the value of p that maximizes $\mathcal{L}$. It can be simply found by setting $\frac{d}{dp}\mathcal{L}(p)|_{p=\hat{p}} = 0$ that $\hat{p} = \frac{Y}{n}$. From this it follows that

$$\lambda = \frac{1}{2^n}\left(\frac{n}{Y}\right)^Y \left(\frac{n}{n-Y}\right)^{n-Y} \tag{3}$$

The likelihood ratio test then is given (for some α), by

$$\lambda < k_\alpha \tag{4}$$

$$\frac{1}{2^n}\left(\frac{n}{Y}\right)^Y \left(\frac{n}{n-Y}\right)^{n-Y} < k_\alpha \tag{5}$$

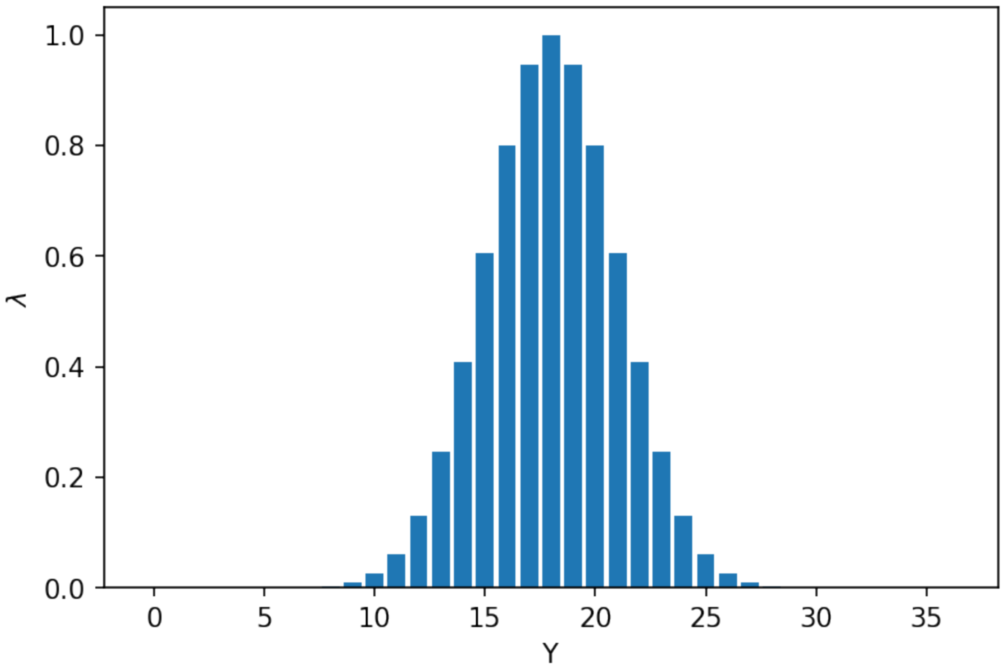
In [1]:
```
import numpy as np
from scipy.stats import binom
import matplotlib.pyplot as plt
%matplotlib inline
plt.rcParams['figure.dpi']=150
```

In [2]:
```
Nflips = 36
def lambda_(n, Y):
    return 2.0**-n * (n/Y)**Y * (n/(n-Y))**(n-Y)

xs = np.array(range(Nflips+1))
```

```
plt.bar(xs, lambda_(Nflips, xs))
plt.xlabel('Y')
plt.ylabel(r'$\lambda$')
```

```
/usr/lib/python3.6/site-packages/ipykernel/__main__.py:3: RuntimeWarning: divide by zero encou
  app.launch_new_instance()
```

```
Out[2]: Text(0,0.5,'$\\lambda$')
```



a) We have as a given that the exclusion region is $|Y - 18| \geq 4$. This corresponds to $k_\alpha = 0.4080...$, so now let's find the α value by running some MC experiments.

```
In [3]: k = lambda_(36, 14)
        Nexp = 1000000

        Ys = binom.rvs(Nflips, 0.5, size=Nexp)
        lambdas = lambda_(Nflips, Ys)

        Npass = np.sum(lambdas < k)
        print(f'alpha = {Npass/Nexp}')
```

```
alpha = 0.132401
```

So, for the given acceptance criterion, $\alpha \approx 0.13$.

---

b) Now, suppose that $p = 0.7$. What is the value of *beta*?

```
In [4]: Ys = binom.rvs(Nflips, 0.7, size=Nexp)
        lambdas = lambda_(Nflips, Ys)

        Npass = np.sum(lambdas >= k)
        print(f'beta = {Npass/Nexp}')
```

```
beta = 0.162632
```

```
/usr/lib/python3.6/site-packages/ipykernel/__main__.py:3: RuntimeWarning: divide by zero encou
  app.launch_new_instance()
```

So, for the chosen value of p, $\beta \approx 0.16$.

---

c) False, it is the probability of our test rejecting H_0 when it is true.
d) False, it is the probability of our test failing to reject H_0 when $H_a(p = 0.7)$ is true.
e) Correct, It was assumed that $H_a(p = 0.7)$ was true.
f) True, since the alternative hypotheses is more similar to the null hypotheses, they are more difficult to distinguish, which will lead to a larger β value.
g) False, α is the probably rejecting H_0, when it is true.
h) True, the rejection region growing from $|y - 18| \geq 4$ to $|y - 18| \geq 2$ would lead to more incorrect rejections.
i) True.
j) False, it would be smaller because a less stringent rejection criteria would make it easier to reject H_0.

P3

October 23, 2017

An experiment yields two samples x_1, and x_2 that are drawn from a Cauchy distribution with unkown mean, μ. The null hypotheses is $\mu = 0.5$, and the alternative is $\mu = 0.5$. If we apply a likelihood ratio test, our test statistic is

$$\lambda = \frac{\mathcal{L}(\mu = 0.5)}{\mathcal{L}(\mu = 1.5) + \mathcal{L}(\mu = 0.5)} \tag{1}$$

where,

$$\mathcal{L}(\mu) = \frac{1}{\pi^2} \frac{1}{(1 + (x_1 - \mu)^2)(1 + (x_2 - \mu)^2)} \tag{2}$$

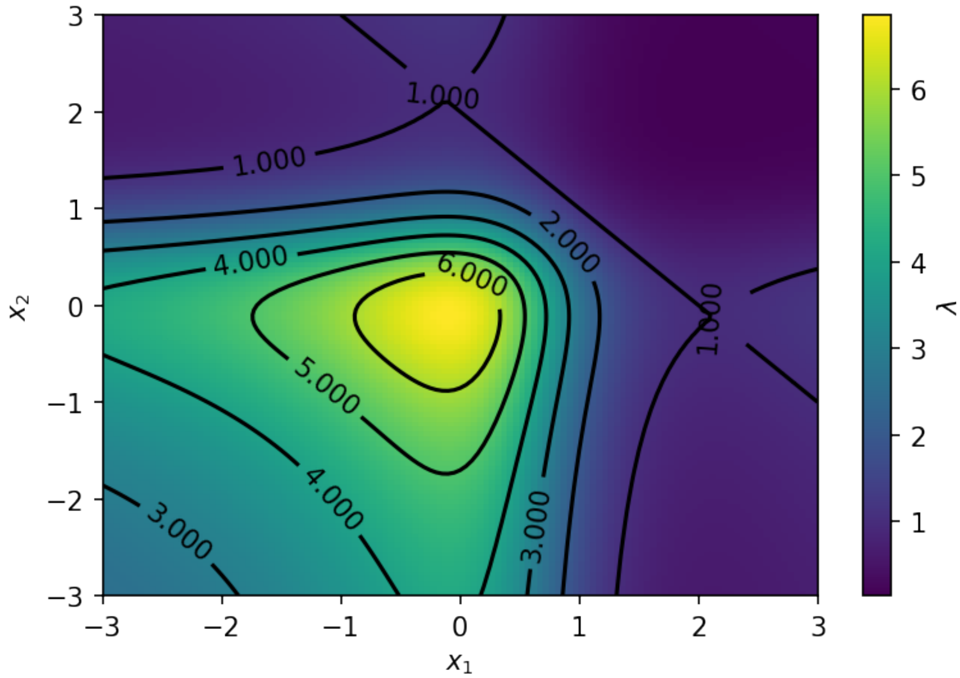
In [1]:
```
import numpy as np
from scipy.stats import cauchy
import matplotlib.pyplot as plt
%matplotlib inline
plt.rcParams['figure.dpi']=150
```

In [2]:
```
def lambda_(x1, x2, mu_0, mu_a, type_=1):
    L = cauchy.pdf
    L_0 = L(x1, mu_0)*L(x2, mu_0)
    L_a = L(x1, mu_a)*L(x2, mu_a)
    return L_0 / L_a
```

In [3]:
```
dx = 0.05
bound = 3
x1s, x2s = np.mgrid[slice(-bound,bound+dx, dx),
                    slice(-bound,bound+dx, dx)]

lambdas = lambda_(x1s, x2s, 0.5, 1.5)
```

In [4]:
```
levels = list(range(7))
plt.pcolormesh(x1s, x2s, lambdas)
plt.colorbar().set_label(r'$\lambda$')
CS = plt.contour(x1s, x2s, lambdas, levels=levels, colors='k')
plt.clabel(CS, inline=1)
plt.xlabel(r'$x_1$')
plt.ylabel(r'$x_2$')
max_idx = np.unravel_index(np.argmax(lambdas), lambdas.shape)
print(f'Maximum level of lambda: {lambdas[max_idx]:0.3f} @ x1={x1s[max_idx]:0.2f} x2={
```

```
Maximum level of lambda: 6.852 @ x1=-0.10 x2=-0.10
```

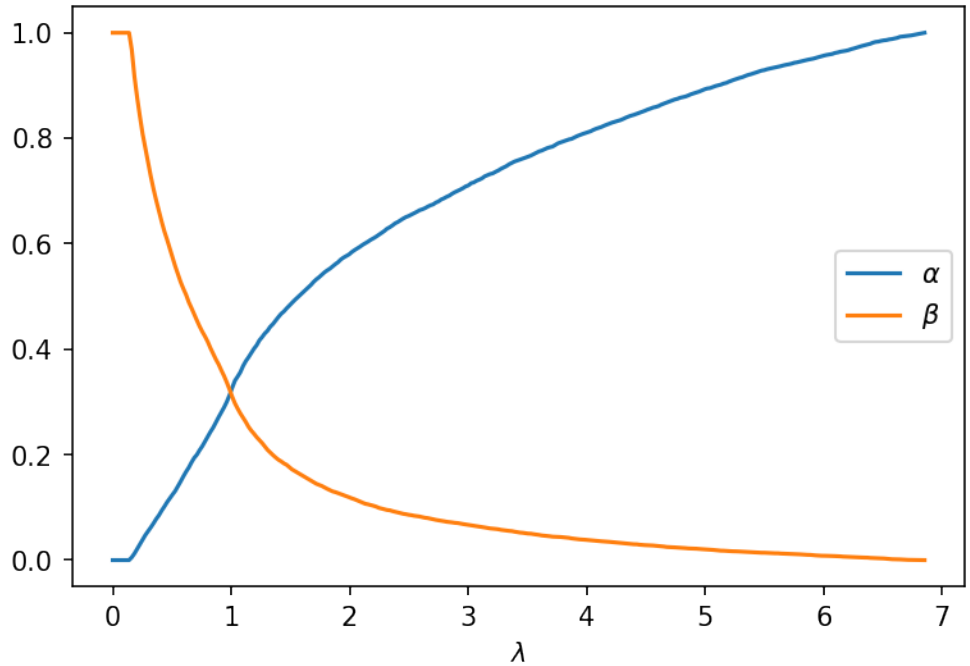


Now, what are the α and β values for various values of λ?

```
In [5]: Nlambdas = 300   # Test Nlambdas values of lambda
        Nexps = 10000   # Run Nexps MC experiments

        test_lambdas = np.linspace(0, np.max(lambdas), Nlambdas)

        def mk_sample(mu):
            return cauchy.rvs(mu, size=Nexps)

        sample_lambdas = lambda_(mk_sample(0.5), mk_sample(0.5), 0.5, 1.5)
        alphas = np.zeros(Nlambdas)
        for i, test_lambda in enumerate(test_lambdas):
            Npass = np.sum(sample_lambdas < test_lambda)
            alphas[i] = Npass / Nexps

        sample_lambdas = lambda_(mk_sample(1.5), mk_sample(1.5), 0.5, 1.5)
        betas = np.zeros(Nlambdas)
        for i, test_lambda in enumerate(test_lambdas):
            Npass = np.sum(sample_lambdas >= test_lambda)
            betas[i] = Npass / Nexps
```

```
plt.plot(test_lambdas, alphas, label=r'$\alpha$')
plt.plot(test_lambdas, betas, label=r'$\beta$')
plt.xlabel(r'$\lambda$')
plt.legend()
```

Out[5]: <matplotlib.legend.Legend at 0x7f3b5f1bfeb8>



Suppose we want to define our rejection region with $\alpha = 0.2$. This corresponds to a rejection region of $\lambda < 0.73$, and a corresponding β value of 0.45.