PG13 コイン投げと大数の法則

In [1]:

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

1. コイン投げのシミュレーション

発生した乱数が0.5未満なら裏(Z=0), 発生した乱数が0.5以上なら表(Z=1)と解釈することで、 公平なコイン投げのシミュレーションになる。

In [2]:

```
if np.random.rand()<0.5:
    Z=0
else:
    Z=1
Z
```

Out[2]:

```
1
```

コイン投げを繰り返し、その結果を記録する。

In [3]:

```
trial = 10000       # コイン投げの回数
Record = []       # コイン投げを記録するためのリストを準備。初期値は空
for _ in range(trial):
    if np.random.rand(1) < 0.5:
        Z=0
    else:
        Z=1
    Record.append(Z)
Coin = np.array(Record)      # コイン投げの結果(0または1)をリストからアレイに変換
Coin
```

Out[3]:

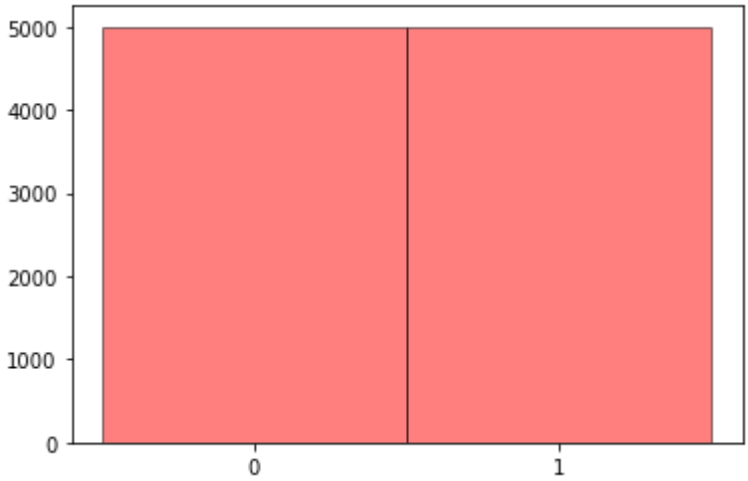
```
array([1, 0, 0, ..., 1, 1, 1])
```

In [4]:

```
# ヒストグラム
plt.hist(Coin,
         range=(-0.5,1.5),    # 幅を指定
         bins=2,              # 階級数を指定
         color='red',          # 色を指定
         alpha=0.5,            # 色の濃さ (0～1)
         ec='k')               # 棒に枠線つける (k=black)
plt.xticks([0,1])
```

Out[4]:

```
([<matplotlib.axis.XTick at 0x2806dcb6160>,
  <matplotlib.axis.XTick at 0x2806dcb6130>],
 [Text(0, 0, ''), Text(0, 0, '')])
```



2. 表の累積度数

In [5]:

```
# 表の回数の経時変化
s = 0 # 初期値
S = [s]  # 初期値をいれたアレイ
for x in Coin:
    s = s + x
    S.append(s)
S = np.array(S)      # アレイにしておくと便利
S
```

Out[5]:

```
array([   0,    1,    1, ..., 4998, 4999, 5000])
```

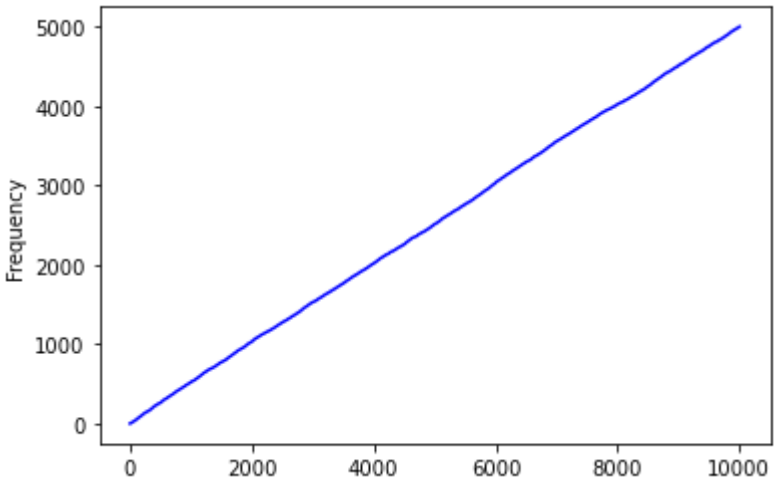
ここでは、0回目の試行までの累積度数は初期値の 0 とした。

In [6]:

```
# 表の回数の経時変化の図示
x_range = np.arange(0, trial+1)
plt.plot(x_range, S, color='blue')
plt.ylabel('Frequency')   # y 軸のラベル
```

Out[6]:

```
Text(0, 0.5, 'Frequency')
```



3. 大数の法則

前節では累積回数 S を数えたが、相対度数(relative frequency)に変えよう。既に得られている S を用いることにする。

In [7]:

```
rf = 0 # 初期値
RF = [rf]  # 初期値をいれたアレイ
for i in range(trial):
    rf = S[i+1]/(i+1)
    RF.append(rf)
RF=np.array(RF)      # アレイにしておくと便利
RF
```

Out[7]:

```
array([0.        , 1.        , 0.5       , ..., 0.49989998, 0.49994999,
       0.5       ])
```

表の相対頻度 RF の経時変化の図示

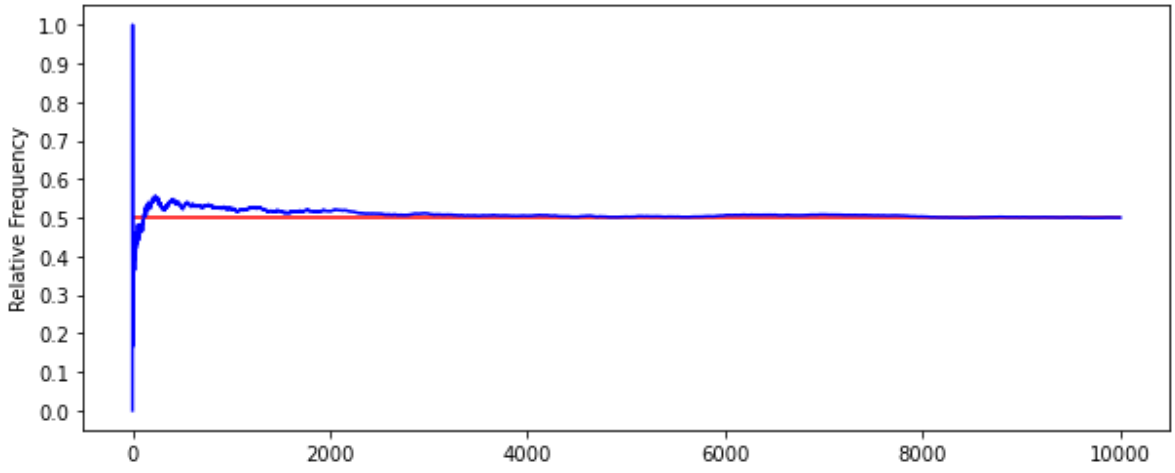
In [8]:

```
plt.figure(figsize=(10,4))
plt.plot(x_range, RF, color='blue')                    # 全部表示

plt.yticks(np.arange(0, 1.1, 0.1))          # y 軸の調整（範囲と幅）
plt.ylabel('Relative Frequency')            # y 軸のラベル
plt.hlines(0.5, 0, trial, color='red')      # y=0.5 の直線
```

Out[8]:

<matplotlib.collections.LineCollection at 0x2806e080a00>



時間軸の一部を切り出して拡大する

In [9]:

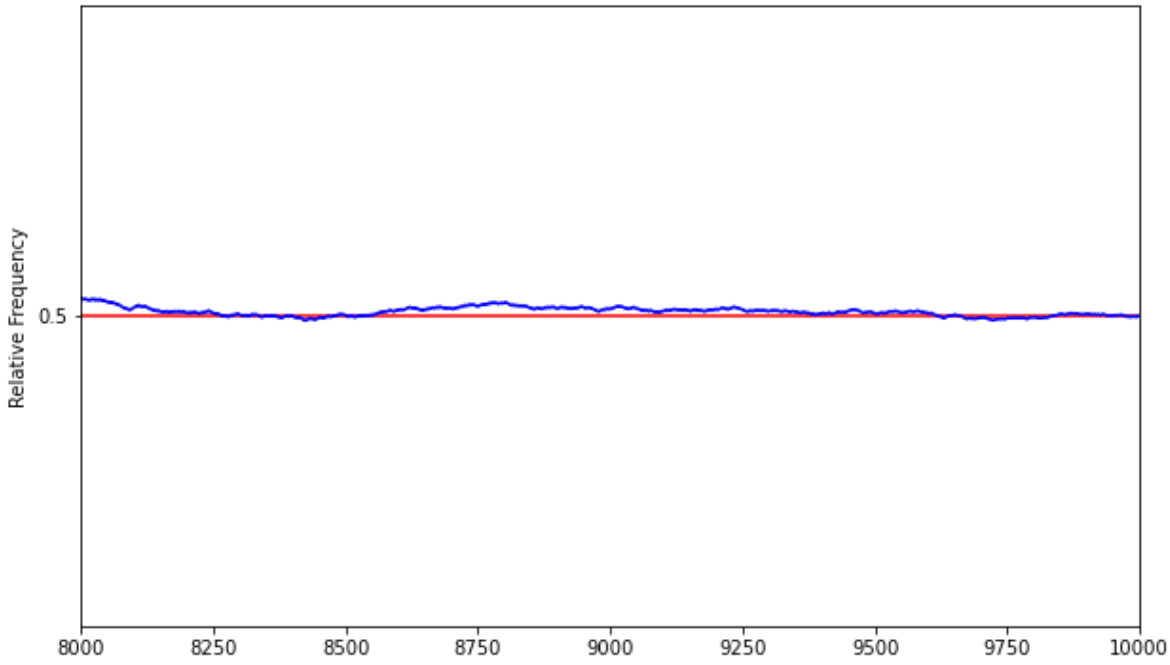
```
# 表の相対頻度の経時変化の図示（時間軸の一部を取り出す）
plt.figure(figsize=(10,6))
plt.plot(x_range, RF, color='blue')

plt.yticks(np.arange(0, 1.1, 0.1))    # y 軸の調整（範囲と幅）
plt.ylabel('Relative Frequency')      # y 軸のラベル
plt.hlines(0.5, 0, trial, color='red')

plt.axis([8000, 10000, 0.45, 0.55])               # x 軸を 8000～10000, y軸を 0.45～0.55 に制限
```

Out[9]:

```
(8000.0, 10000.0, 0.45, 0.55)
```



In [ ]:

```
```