sheet1

October 26, 2023

1 Exercise 1

Berkay Günes & Niklas Knöll

1.1 1 Set theory

From the lecture we already know $\mathrm{P}(A \cup B) = \mathrm{P}(A) + \mathrm{P}(B) - \mathrm{P}(A \cap B)$. With this it follows:

$$
\begin{aligned}
&\mathrm{P}(A \cup B \cup C) \\
=&\mathrm{P}(A \cup (B \cup C)) \\
=&\mathrm{P}(A) + \mathrm{P}(B \cup C) - \mathrm{P}(A \cap (B \cup C)) \\
=&\mathrm{P}(A) + \mathrm{P}(B) + \mathrm{P}(C) - \mathrm{P}(B \cap C) - \mathrm{P}(A \cap (B \cup C)) && \text{|use distributivity} \\
=&\mathrm{P}(A) + \mathrm{P}(B) + \mathrm{P}(C) - \mathrm{P}(B \cap C) - \mathrm{P}((A \cap B) \cup (A \cap C)) \\
=&\mathrm{P}(A) + \mathrm{P}(B) + \mathrm{P}(C) - \mathrm{P}(B \cap C) - \mathrm{P}(A \cap B) - \mathrm{P}(A \cap C) + \mathrm{P}((A \cap B) \cap (A \cap C)) \\
=&\mathrm{P}(A) + \mathrm{P}(B) + \mathrm{P}(C) - \mathrm{P}(B \cap C) - \mathrm{P}(A \cap B) - \mathrm{P}(A \cap C) + \mathrm{P}(A \cap B \cap C)
\end{aligned}
$$

1.2 2 Conditional probabilities

Two events, and , are such that ( ) = 0.5, ( ) = 0.3, and ( ) = 0.1. (a) $P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{0.1}{0.3} = \frac{1}{3}$ (b) $P(A|A \cup B) = \frac{P(A \cap (A \cup B))}{P(A \cup B)} = \frac{P(A)}{P(A)+P(B)-P(A \cap B)} = \frac{5}{7}$ (c) $P(A|A \cap B) = \frac{P(A \cap (A \cap B))}{P(A \cap B)} = \frac{P(A \cap B)}{P(A \cap B)} = 1$ (d) $P(\overline{A \cap B}) = 1 - P(A \cap B) = 0.9$

1.3 3 Event composition

In a group of 100 students, 40 are taking a math class, 30 are taking a physics class, and 20 are taking both math and physics. You randomly select a student. Calculate the probability that the selected student is taking a math class, but not a physics class.

$n = 1, m = 0.4, p = 0.3, m \cap p = 0.2$

$P = P(m) - P(m \cap p) = \frac{0.4-0.2}{1} = 0.2$

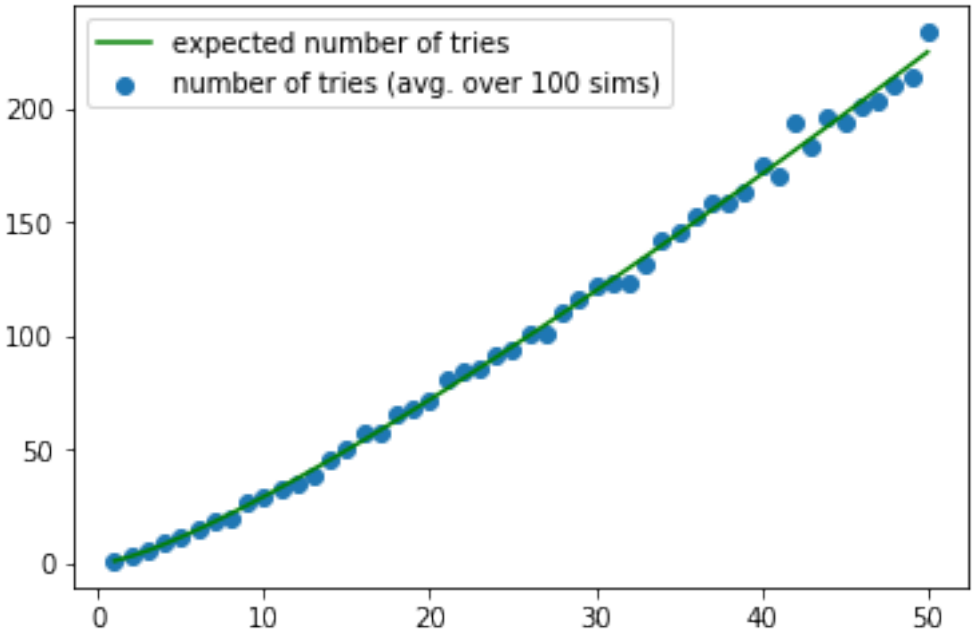
1.4 4 Coupon collector's problem

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.special import erfinv
from scipy.special import zeta
from scipy.stats import norm
```

```
import random
from math import factorial
import mpmath as mp
```

[ ]:
```
N=np.linspace(1,50,50)

def tries(N,runs):
    results=[]
    for i in range(runs):
        coupons = 0
        N_tries = 0
        while coupons < N:
            N_tries += 1
            r = np.random.uniform(0,1,1)
            if r < (N - coupons)/N:
                coupons += 1
        results.append(N_tries)
    return results

def avg_tries(N,runs):
    return np.mean(tries(N,runs))

simulations=[]
for i in N:
    simulations.append(avg_tries(i,100))

def expected(N):
    t=0
    for i in range(N):
        i=int(i)
        t+=1/(i+1)
    return t*N

exp_values=[]
for i in N:
    exp_values.append(expected(int(i)))

plt.scatter(N,simulations,label="number of tries (avg. over 100 sims)")
plt.plot(N,exp_values, label="expected number of tries",color="g")
plt.legend()
```

[ ]:
```
<matplotlib.legend.Legend at 0x7fd37f10fcf8>
```


expected number of tries
number of tries (avg. over 100 sims)
200
150
100
50
0
0
10
20
30
40
50