Statistical Methods Fall 2020 Answers to Problem Set 2

Jake Muff
Student number: 015361763
22/09/2020

1. Exercise 1

   (a) For the uncertainty in the measurements of the amount of carpet and wallpapers needed. For the amount of carpet needed this is simply just the bottom rectangle of the room and therefore the area of carpet needed is just the length multiplied by the width, therefore the uncertainty is the amount of carpet needed is
   Area is

$$A_c = l \pm \sigma_l \times w \pm \sigma_w$$

Variance is then

$$\sigma_{A_c}^2 = A_c^2 \cdot \left[\left(\frac{\sigma_l}{l} \right)^2 + \left(\frac{\sigma_w}{w} \right)^2 + \frac{2\sigma_{lw}}{lw} \right]$$

With standard deviation or uncertainty in the Area

$$\sigma_{A_c} = |A_c| \cdot \sqrt{\left(\frac{\sigma_l}{l} \right)^2 + \left(\frac{\sigma_w}{w} \right)^2 + 2 \left(\frac{\sigma_{lw}}{lw} \right)} \tag{1}$$

For the wallpaper we have 4 surfaces. The rectangle for the room has 6 surfaces. 4 that need to be covered by wallpaper, 1 for the ceiling and 1 for the floor for the carpet. So the area of wallpaper needed is

$$A_w = (h \pm \sigma_h \times l \pm \sigma_l) \times 2 + 2 \times (h \pm \sigma_h \times w \pm \sigma_w)$$

So to calculate the total uncertainty we calculate uncertainty caused by multiplication in $h \times l = C$ and in $h \times w = D$ so that

$$\sigma_C = |C| \cdot \sqrt{\left(\frac{\sigma_h}{h} \right)^2 + \left(\frac{\sigma_l}{l} \right)^2 + 2 \left(\frac{\sigma_{hl}}{hl} \right)} \tag{2}$$

And for $h \times w = D$

$$\sigma_D = |D| \cdot \sqrt{\left(\frac{\sigma_h}{h} \right)^2 + \left(\frac{\sigma_w}{w} \right)^2 + 2 \left(\frac{\sigma_{hw}}{hw} \right)} \tag{3}$$

The uncertainty change caused by multiplication by a constant means that the uncertainty is multiplied by the modulus of that constant. The uncertainty change caused by the addition of the 2 multiplication parts of the equation means that the uncertainty in the amount of wallpaper needed is

$$\sigma_{A_w} = \sqrt{4 \times (\sigma_C)^2 + 4 \times (\sigma_D)^2 + 8 \times (\sigma_{CD})} \tag{4}$$

To show that the uncertainty in the amount of carpet needed and the amount of wallpaper needed is correlated we can calculate the covariance matrix where the diagonal terms are the variance for the carpet and wallpaper respectively and the off diagonal terms are the covariances between them.

$$cov(\sigma_{A_c}, \sigma_{A_w}) = \begin{pmatrix} \sigma_{A_c} & \sigma_{A_c A_w} \\ \sigma_{A_w A_c} & \sigma_{A_w} \end{pmatrix}$$

The (Pearson) correlation coefficient is calculated through

$$r = \frac{cov(\sigma_{A_c}, \sigma_{A_w})}{\sigma_{A_c} \cdot \sigma_{A_w}}$$

I now simulated this and put it into python, making use of the *np.cov* and *pearsonr* functions for *NumPy*. The uncertainties in the lenght, width and height were equalised and the length and width were simulated as 10 random integers between 2 and 10m with the height as 2.40m
However, I struggled to understand what the question was asking. A plot of the length or width of the room against the correlation coefficient will simply lead to a straight line plot as the correlation coefficient is a constant between the two random variables, it does not change as a function of the length or width.

2. Exercise 2

   (a) i. Probability to observe the Aurora Borealis every night?
   Assuming the probability of Northern lights being observable and probability of cloudless sky to be independent we can simply multiply them. So the probability of these two independent events happening every night for 3 nights is

   $$P(NL) = \frac{1}{2} \;;\; P(cloudless) = \frac{3}{5}$$

   Chance of getting aurora borealis every <u>one</u> night is

   $$0.5 \times 0.6 = 0.3$$

   ii. Probability to not observe the Northern Lights at all in 3 days
   A binomial distribution can be used as the problem can be viewed as having

two discrete outcomes, either you observe the Northern lights or you don't. We can use this to calculate the probability of observing a specific number of successful events, such as observing the Northern Lights.

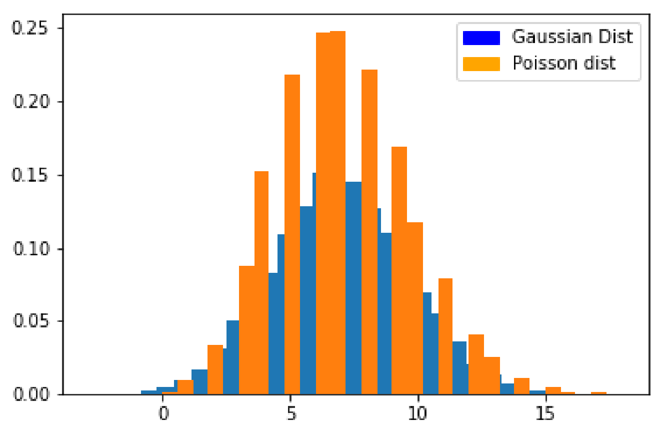
$$f(n; N, P) = \frac{N!}{n!(N-n)!} p^n (1-P)^{N-n}$$

Here, N is the number of observations or days in our case. n is the number of successes we are looking for, so in the case of observing the Northern Lights at least once in 3 days $n = 1$. The probability of success or observing the northern lights is denoted by P.
So the probability of not observing the Northern lights in 3 days is

$$\frac{3!}{0!(3-0)!} 0.3^0 (1-0.3)^3$$

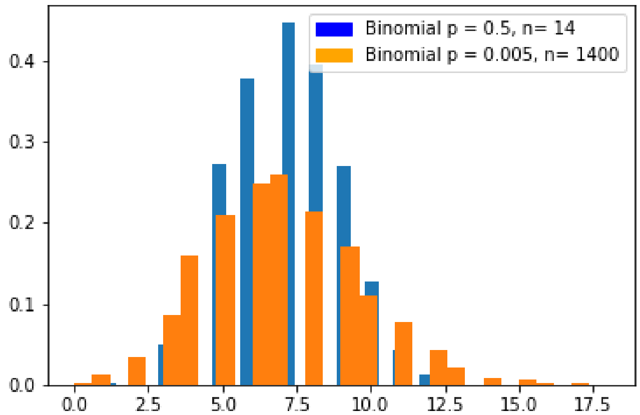
$$= 0.343$$

(b) i. Compare a Poisson distribution with a mean of 7 with the corresponding Gaussian distribution (the one having the same mean and variance as the Poisson distribution) by plotting them on top of each other in the same histogram.
This was done using python and pyplot with the plot shown for plots in the appendix and the source code as well.



Kuva 1: Plot of a gaussian distribution vs a poisson distribution

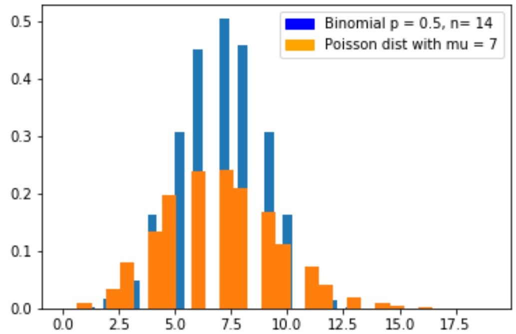
ii. Do the same for two binomial distributions each at a time: one with p = 0.5 and 14 trials and another one with p = 0.005 and 1400 trials. What are the discrepancies between the two on-top-of-each- other plotted distributions?

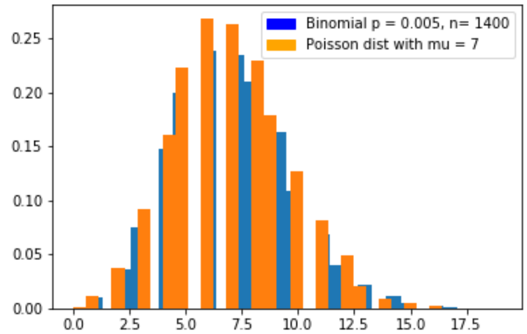


Kuva 2: Plot of 2 binomial distributions with different properties

Which one of the binomial distributions is approximated better by the Gaussian distribution?
Discrepancies between the Gaussian and the Poisson distribution are that the gaussian distribution is continuous which is seen by the fact that there are no gaps between the bins in the histogram. The Poisson and Binomial distribution are discrete and bounded at 0 hence the histogram stopping at 0 for them but not for the Gaussian. The Gaussian is also symmetric whereas the Poisson is not.
The binomial distribution with p = 0.005 and n=1400 (orange) is better approximated by the gaussian distirbution as the histogram is more bell curved with an equal spread. This is due to having a higher n number and the probability being lower. Having more trials increases its approximation by a gaussian curve.

iii. Poisson Distrbution vs Binomial Distribution
As we can see the second binomial distribution with p = 0.005 and n = 1400 resembles the poisson distribution more.





Kuva 3: Plot of a binomial dist with p=0.5, n =14 vs a poisson dist with mu = 7

Kuva 4: Plot of a binomial dist with p=0.005, n =1400 vs a poisson dist with mu = 7

3. Appendix

(a) 2b(i)

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

N = 10000
mu = 7
sigma = np.sqrt(mu)

G = np.random.normal(mu, sigma, N)
P = np.random.poisson(mu, N)

count, bins, ignored = plt.hist(G, 30, density=True)

count, bins, ignored = plt.hist(P, 30, density=True)
blue_patch = mpatches.Patch(color='blue', label='Gaussian Dist')
orange_patch = mpatches.Patch(color='orange', label='Poisson dist')
plt.legend(handles=[blue_patch, orange_patch])
plt.savefig('GVsP.png')
plt.show()
```

(b) 2b(ii)

```
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.patches as mpatches


n_1 = 14 #number of trials
p_1 = 0.5 #probability
```

```
n_2 = 1400 #number of trials
p_2 = 0.005 #probability


B_1 = np.random.binomial(n_1, p_1, N)
B_2 = np.random.binomial(n_2, p_2, N)
count, bins, ignored = plt.hist(B_1, 30, density=True)

count, bins, ignored = plt.hist(B_2, 30, density=True)
blue_patch = mpatches.Patch(color='blue', label='Binomial p = 0.5, n= 14')
orange_patch = mpatches.Patch(color='orange', label='Binomial p = 0.005, n= 1400'
plt.legend(handles=[blue_patch, orange_patch])
plt.savefig('BVsB.png')
plt.show()
```

(c) Binomial Dist vs Gaussian Dists

```
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.patches as mpatches


n_1 = 14 #number of trials
p_1 = 0.5 #probability

N = 10000
mu = 7
sigma = np.sqrt(mu)

B_1 = np.random.binomial(n_1, p_1, N)
P = np.random.poisson(mu, N)

count, bins, ignored = plt.hist(B_1, 30, density=True)
count, bins, ignored = plt.hist(P, 30, density=True)
blue_patch = mpatches.Patch(color='blue', label='Binomial p = 0.5, n= 14')
orange_patch = mpatches.Patch(color='orange', label='Poisson dist with mu = 7')
plt.legend(handles=[blue_patch, orange_patch])
plt.savefig('B1vsP.png')
plt.show()

import matplotlib.pyplot as plt
import numpy as np
import matplotlib.patches as mpatches


n_2 = 1400 #number of trials
p_2 = 0.005 #probability

N = 10000
```

```
mu = 7
sigma = np.sqrt(mu)

B_2 = np.random.binomial(n_2, p_2, N)
P = np.random.poisson(mu, N)

count, bins, ignored = plt.hist(B_2, 30, density=True)
count, bins, ignored = plt.hist(P, 30, density=True)
blue_patch = mpatches.Patch(color='blue', label='Binomial p = 0.005, n= 1400')
orange_patch = mpatches.Patch(color='orange', label='Poisson dist with mu = 7')
plt.legend(handles=[blue_patch, orange_patch])
plt.savefig('B2vsP.png')
plt.show()
```

(d) Python Code for Question 1

```
import matplotlib.pyplot as plt
import numpy as np
from numpy.random import rand
from numpy.random import seed
import random
from scipy.stats import pearsonr

# seed random number generator
seed(np.random.randint(1,100))

n=10

L = np.random.randint(1,10,n) #random integer between 1 and 10 for length
W = np.random.randint(1,10,n) #randm integer between 1 and 10 for width
#L = np.array([6,9,6,1,1,2,8,7,3,5])
#W = np.array([6,3,5,3,5,8,8,2,8,1])
H = np.full((1,n), 2.40) #height (and length and width) in m. array of 2.40s
carpet = L * W #area of carpet needed
wallpaper = 2*(H*L) + 2*(H*W) #area of wallpaper needed

#print(carpet)
#print(wallpaper)
#assuming uncertainties to be equal
sigmaL = 0.01 #uncertainty in length
sigmaW= 0.01 #uncertainty in width
sigmaH=0.01 #uncertainty in height

covLW = np.cov(L,W) #covariance between length and width
covHL = np.cov(H,L) #covariance between height and lenght
covHW = np.cov(H,W) #covariance between ehight and width

sigmaCarpet = carpet * np.sqrt((sigmaL/L)**2 + (sigmaW/W)**2 + (2*(abs(covLW[
sigmaHL = (H*L) * np.sqrt((sigmaH/H)**2 + (sigmaL/L)**2 + (2*(abs(covHL[0][1]/
sigmaHW = (H*W) * np.sqrt((sigmaH/H)**2 + (sigmaW/W)**2 + (2*(abs(covHW[0][1]/
```

```
covCD = np.cov(H*L, H*W) #covariance between C and D or the height length and

sigmaWallpaper = np.sqrt((4 * sigmaHL**2) + (4* sigmaHW**2)+ (8*abs(covCD[0][

print(len)

print(len(sigmaHL))
print(len(sigmaHW))

print(len(sigmaWallpaper))
print(len(sigmaCarpet))

covCaWa= np.cov(sigmaCarpet, sigmaWallpaper)   #covariance between carpet and
corr2,_ = pearsonr(sigmaCarpet,sigmaWallpaper)

#np.std(L) #std dev for L
#np.std(W) #std dev for W

#calculating the covariance between the uncertainties in the carpet and wallp
sigmaC_s = sigmaCarpet**2
sigmaWa_s = sigmaWallpaper**2
sumsigmaC = np.sum(sigmaCarpet)
sumsigmaWa = np.sum(sigmaWallpaper)
sumC2 = np.sum(sigmaC_s)
sumWa2 = np.sum(sigmaWa_s)

covCW = n*(np.sum(sigmaCarpet*sigmaWallpaper))-(np.sum(sigmaCarpet)*(np.sum(si

rCW = covCW / (np.sqrt((n*sumC2 - sumsigmaC**2)*(n*sumWa2 - sumsigmaWa**2)))


L_s = L**2
W_s = W**2
sumL = np.sum(L)
sumW = np.sum(W)
sumL2 = np.sum(L_s)
sumW2 = np.sum(W_s)

cov = n*(np.sum(L*W))-(np.sum(L)*(np.sum(W)))

r = cov / (np.sqrt((n*sumL2 - sumL**2)*(n*sumW2 - sumW**2)))

corr2,_ = pearsonr(L,W)


print(L)
print(W)
print(H)
```