Name: Ajayrangan Kasturirangan
NetID: ak10196

Complex Urban Systems
Assignment 2

NYU

Question 1

Consider a small dataset of four points D = (0, 1),(1, 0),(3, 2),(5, 4). Compute the equation of the linear regression, the total explained variance, the total variance, and the coefficient of determination. Please detail your steps.

To get the linear regression equation, we must first determine the slope and y-intercept of the line that best fits the data. To do so, we can utilize the following formulas:

$$Slope\ (m) = \frac{N\sum(xy) - \sum x \sum y}{N\sum(x^2) - (\sum x)^2} \tag{1}$$

$$Y - intercept\ (b) = \frac{\sum y - m\sum x}{N} \tag{2}$$

Substituting these values:

$$N = 4 \tag{3}$$

$$\sum x = 9 \tag{4}$$

$$\sum y = 7 \tag{5}$$

$$\sum(xy) = 26 \tag{6}$$

$$\sum(x^2) = 35 \tag{7}$$

We get Slope (m) = 0.69 and Y-intercept (b) = 0.1975
This gives the Linear Regression equation:

$$y = 0.69 * x + 0.1975 \tag{8}$$

To calculate the total explained variance, we can use:

$$Explained\ Variance = \sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{9}$$

y_i is the actual value of the i-th data point, and $\hat{y}_i$ is the predicted value of the i-th data point. For our dataset, we get:

$$Explained\ Variance = (1 - 0.1975)^2 + (0 - 0.8875)^2 + (2 - 2.2675)^2 + (4 - 3.6475)^2 \tag{10}$$

$$Explained\ Variance = 1.627475 \tag{11}$$

To calculate the total variance, we can use:

$$Total\ Variance = \sum_{i=1}^{N} (y_i - \bar{y})^2 \tag{12}$$

$\bar{y}$ is the mean. that is 1.75

$$Total\ Variance = (1 - 1.75)^2 + (0 - 1.75)^2 + (2 - 1.75)^2 + (4 - 1.75)^2 \tag{13}$$

$$Total\ Variance = 8.75 \tag{14}$$

Coefficient of determination or R^2, is given by:

$$R^2 = 1 - \frac{Explained\ Variance}{Total\ Variance} \tag{15}$$

For our dataset, we get:

$$R^2 = 1 - \frac{1.627475}{8.75} = 0.814 \tag{16}$$

Question 2

Please study the GHS Urban Centre Database from $https://ghsl.jrc.ec.europa.eu/ghs\_stat\_ucdb2015mt\_r2019a.p$ and write a 200-word summary of what the data represents. What type of questions can we answer with this data?

The GHS Urban Centre Database (GHS-UCDB) is a collection of information about cities all over the world. This data is based on different things like how many people live there and how much of the area is built up. The data is gathered from lots of sources and put together with GHSL data. The database tells us about lots of things in the cities like how big they are, where they are, what the weather is like, how is the economy, and how much pollution there is. The answer to these questions is provided using standard metrics that are mentioned in the data dictionary. The cities are measured using a 1x1 km grid, and they're classified as urban centers based on how many people live there and how much of the area is built up. The GHSL gives the input data, and the "degree of urbanization" (DEGURBA) is used to set the rules. Overall, the GHS-UCDB gives us lots of information about cities all over the world in a form that is conducive to analysis.

1. Data can assist service companies in determining the optimal boundaries for their service areas. For instance, delivery service aggregators like DoorDash heavily rely on urban areas, and by using the data, they can identify the extent of the urban centers and set their service boundaries accordingly.

2. Data can be used to find out correlations between interesting features like: a) Night time light emission and GDP b) Greenness and Flooding c) How Geography affects built up area, Economy and pollution.

3. Based on the multiple features available we can try to predict which urban centers are likely to experience or have the potential for rapid growth in the future based on current data and previous year data.

Question 3

From $https://ghsl.jrc.ec.europa.eu/ghs\_stat\_ucdb2015mt\_r2019a.php$, download the associated csv file and perform a scaling analysis between the total population in 2015 and the urban center extension for the following nations: USA, Canada, Poland, and Russia. In your analysis, please exclude any city with less than 100 km2 in area. Does the analysis match your expectations? Please explain your findings.

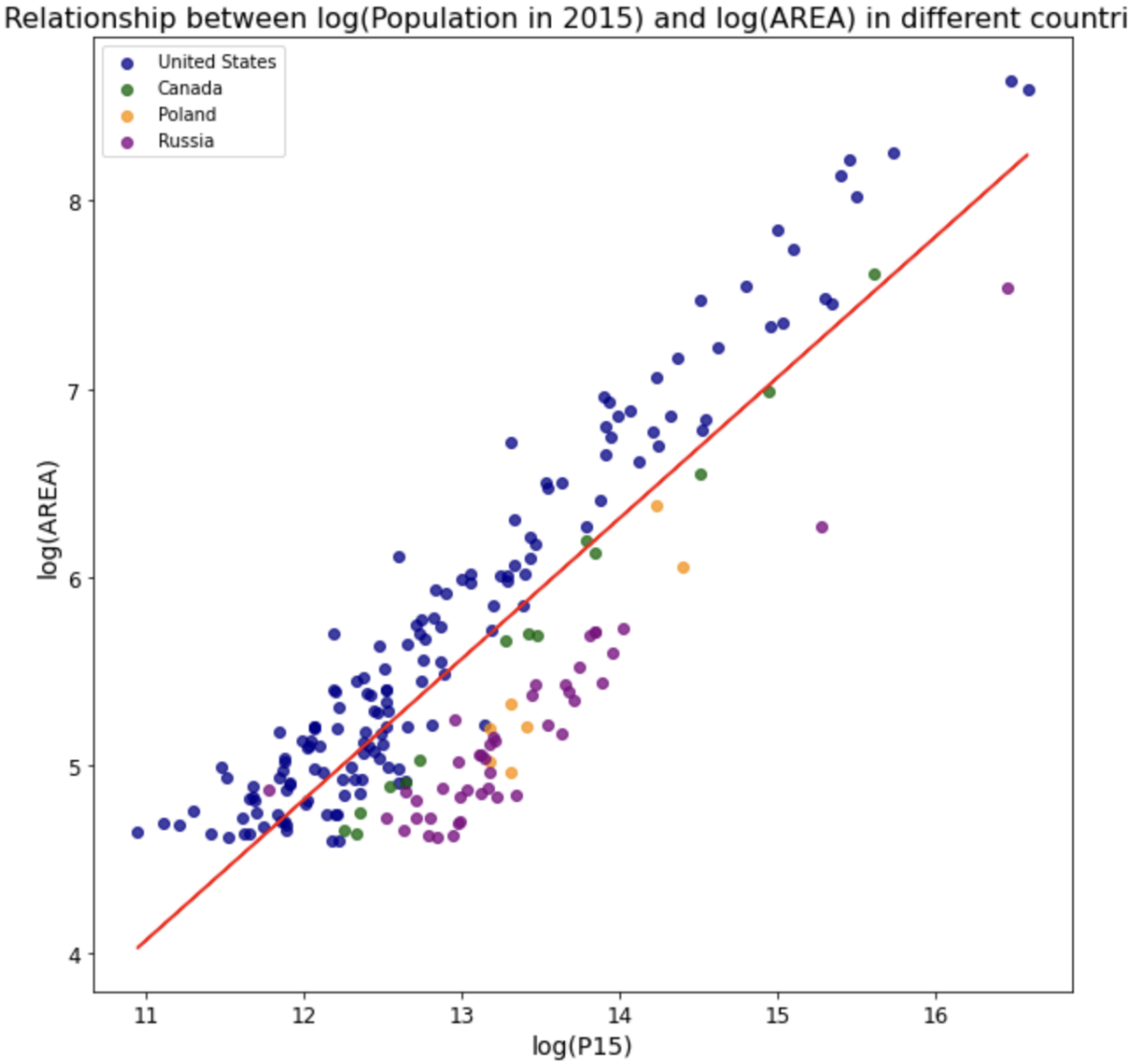
Using the code as displayed below:

```
import pandas as pd
import numpy as np


data = pd.read_csv('GHS_STAT_UCDB2015MT_GLOBE_R2019A_V1_2.csv', encoding='ISO-8859-1',
    low_memory=False)
data1 = data[['CTR_MN_NM','UC_NM_MN','P15','BBX_LATMN','BBX_LATMX','BBX_LONMN','BBX_LONMX
    ','AREA']]
countries = ['United States', 'Canada', 'Poland', 'Russia']
data1 = data1[data1['CTR_MN_NM'].isin(countries)]
data1 = data1[data1['AREA']>=100]

data1 = data1[['P15', 'AREA', 'CTR_MN_NM']].dropna()

import numpy as np
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt

for country in countries:
    data_country = data1[data1['CTR_MN_NM'] == country]
    data_log = data_country[['P15', 'AREA']].dropna().apply(np.log)
    X = data_log['P15'].values.reshape(-1, 1)
    y = data_log['AREA']
    model = LinearRegression().fit(X, y)
    print("\n\n",country)
    print('Intercept:', model.intercept_)
    print('Coefficient:', model.coef_[0])
    print('R^2:', model.score(X, y))
    if model.coef_[0] == 1:
        print('Linear relationship')
    elif model.coef_[0] < 1:
        print('Sublinear relationship')
    else:
        print('Superlinear relationship')
    plt.figure(figsize=(10, 10))
    plt.scatter(X, y, alpha=0.75, color='darkblue')
    plt.plot(X, model.predict(X), color='red')
    plt.xlabel('log(Population in 2015)', fontsize=14)
    plt.ylabel('log(AREA)', fontsize=14)
    plt.title('Relationship between log(Population in 2015) and log(AREA) in ' + country,
         fontsize=16)
    plt.tick_params(axis='both', labelsize=12)
```

```
    plt.show()


data_log = data1[['P15', 'AREA', 'CTR_MN_NM']].dropna()
data_log['P15'] = np.log(data_log['P15'])
data_log['AREA'] = np.log(data_log['AREA'])


X = data_log['P15'].values.reshape(-1, 1)
y = data_log['AREA']
model = LinearRegression().fit(X, y)
print("\n\nAll 4 Countries")
print('Intercept:', model.intercept_)
print('Coefficient:', model.coef_[0])
print('R^2:', model.score(X, y))
if model.coef_[0] == 1:
    print('Linear relationship')
elif model.coef_[0] < 1:
    print('Sublinear relationship')
else:
    print('Superlinear relationship')

countries_colors = {'United States': 'darkblue', 'Canada': 'darkgreen', 'Poland': '
    darkorange', 'Russia': 'purple'}
data_log['color'] = data_log['CTR_MN_NM'].map(countries_colors)
fig, ax = plt.subplots(figsize=(10, 10))
for country, color in countries_colors.items():
    data_country = data_log[data_log['CTR_MN_NM'] == country]
    plt.scatter(data_country['P15'], data_country['AREA'], alpha=0.75, color=color, label
        =country)
plt.plot(X, model.predict(X), color='red')
plt.xlabel('log(P15)', fontsize=14)
plt.ylabel('log(AREA)', fontsize=14)
plt.title('Relationship between log(Population in 2015) and log(AREA) in different
    countries', fontsize=16)
plt.tick_params(axis='both', labelsize=12)
plt.legend()
plt.show()
```

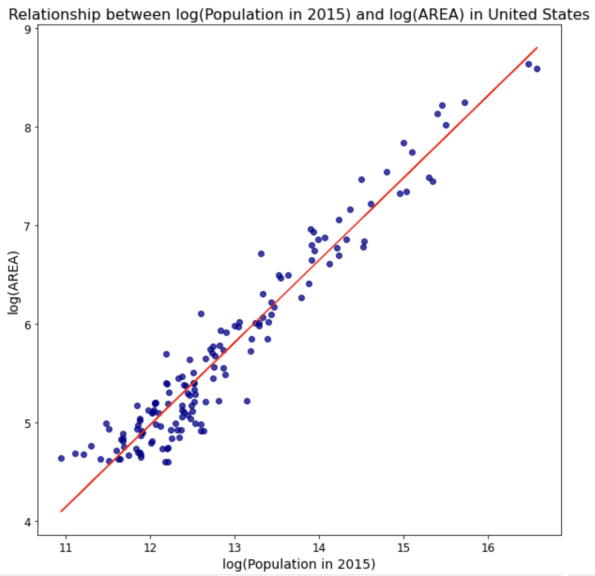
We get the following outputs as listed. Let's see the final output with all 4 countries followed by each individual country.

```
All 4 Countries
Intercept: -4.143808264706207
Coefficient: 0.7468159086791123
R^2: 0.7586324868929072
Sublinear relationship
```

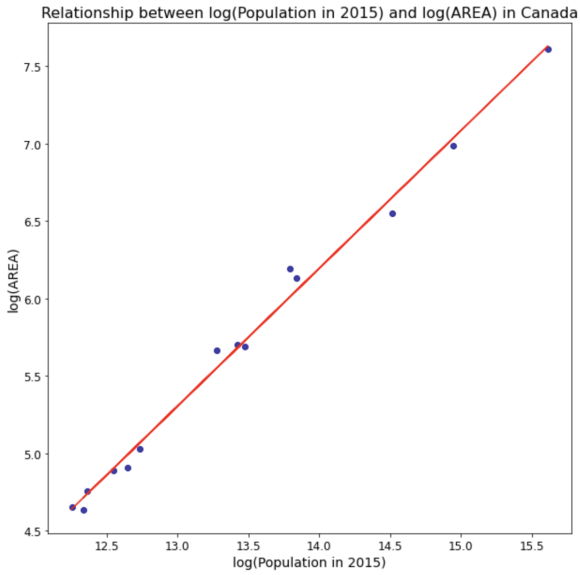


Here We can notice a sub-linear relationship as backed by theory. Following this, we have interesting results in other countries where the USA and CANADA are similar with the scaling coefficient in the range of 0.8 - 0.9 while Russia has a much lower scaling coefficient of 0.66. Indicating that cities in Russia have a much lesser increase in area as the population increases. Meanwhile, in Poland we have 0.988 indicating an almost linear increase in the area of the city along with an increase in population. The graphs for individual countries in the following page.

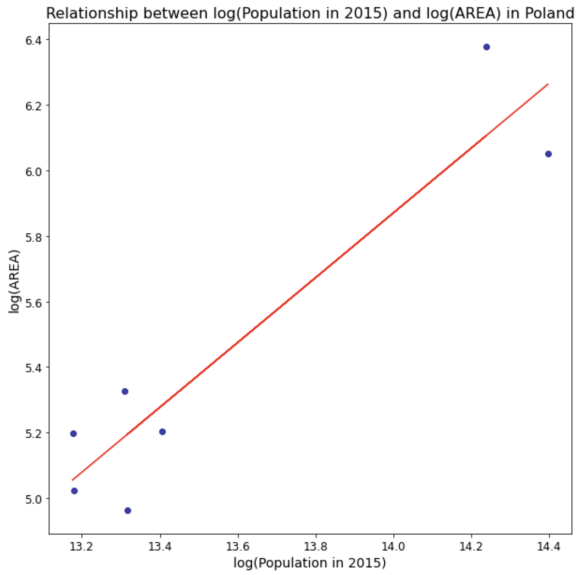
```
 United States
Intercept: -5.036636236695314
Coefficient: 0.8342868837859487
R^2: 0.9313119228493343
Sublinear relationship
```



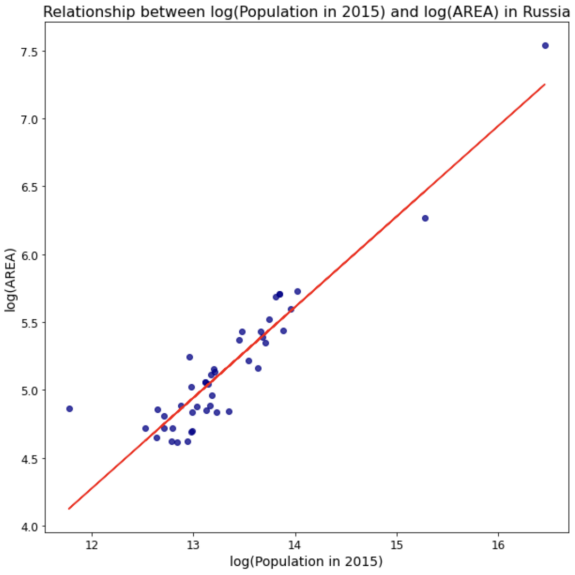

```
 Canada
Intercept: -6.25676946786586
Coefficient: 0.889359630243167
R^2: 0.9923562716968196
Sublinear relationship
```




```
 Poland
Intercept: -7.974310871947761
Coefficient: 0.988887124217453
R^2: 0.8765464831894277
Sublinear relationship
```




```
 Russia
Intercept: -3.7237621335524933
Coefficient: 0.6667213924716087
R^2: 0.8631859543474508
Sublinear relationship
```



Question 4

Consider the amorphous settlement model in Part 1 of Chapter 3. Assuming that the distance traveled by a person per day is on average 3km and that the physical distance used for defining an interaction is about 1 m, what is the expected total number of interactions per capita in Brooklyn (pull data from Wikipedia on population and area)? Does the number make sense to you based on your daily experience?

According to the amorphous model we have

$$k = \frac{N-1}{A} * a_0 * l_0 \tag{17}$$

$a_0 = 1m$ $l_0 = 3km$

From $https://en.wikipedia.org/wiki/Brooklyn$ we have $N = 2736074$ and $A = 183.4km^2$ considering only land area.

$$k = \frac{273074-1}{183.4} * \frac{1}{1000} * 3 \tag{18}$$

$$k = 4.46 \tag{19}$$

From my daily experience, I think this number is more or less correct on average. There are days when I have interactions with 0 people and days when I have interactions with 10 people but more or less can balance out.

Question 5

Consider again the amorphous settlement model in Part 1 of Chapter 3. Assume that the cost of transport is independent of any distance traveled in an imaginary city in which public transport is free for anyone and the only means of transport. How would the area of the city change as a function of the population? Please explain your results.

If the cost of transport is independent of any distance traveled, it means that the cost of transport per capita, C_t, is constant, regardless of the radius of the city. In other words, C_t does not depend on R, the radius of the city.

Given that

$$N * C_t = Y \tag{20}$$

according to the amorphous model, where Y is the total income of the population, and C_t is constant, we have

$$N \alpha Y \tag{21}$$

and

$$Y \alpha \frac{N^2}{A} \tag{22}$$

This gives

$$A \alpha N \tag{23}$$

Therefore, the area of the city is proportional to the population raised to the power of 1, which means that A is directly proportional to N. This is different from the previous case where the cost of transport per capita depended on the radius of the city and $A \alpha N^{(\frac{2}{3})}$.

Question 6

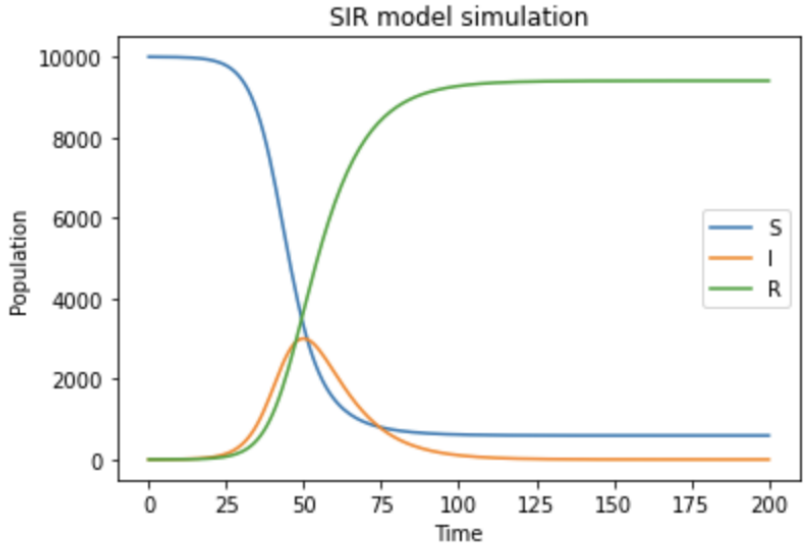
Consider the classical (compartmental) SIR model discussed in class for the following parameters $\beta = 0.3$, $\gamma = 0.1$, and N $= 10,000$. Simulate the model and plot the three variables from t $= 0$ to 200 for the case in which there is only one infected subject at the beginning of the infection and nobody is immune to the disease.

Using these system of equations, we can simulate the SIR model.

$$\frac{dS}{dt} = -\frac{\beta}{N}IS$$
$$\frac{dI}{dt} = \frac{\beta}{N}IS - \gamma I$$
$$\frac{dR}{dt} = \gamma I$$

```
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt

#Initial
S0 = 9999
I0 = 1
R0 = 0


beta = 0.3
gamma = 0.1
N = 10000


t = np.linspace(0, 200, 201)


def sir(y, t, beta, gamma, N):
    S, I, R = y
    dSdt = (-beta/N) * I * S
    dIdt = (beta/N) * I * S - gamma * I
    dRdt = gamma * I
    return [dSdt, dIdt, dRdt]


sol = odeint(sir, [S0, I0, R0], t, args=(beta, gamma, N))


plt.plot(t, sol[:, 0], label='S')
plt.plot(t, sol[:, 1], label='I')
plt.plot(t, sol[:, 2], label='R')
plt.legend()
plt.xlabel('Time')
plt.ylabel('Population')
plt.title('SIR model simulation')
plt.show()
```

```
38
39
40  print("At t=0:")
41  print("S = ", sol[0, 0])
42  print("I = ", sol[0, 1])
43  print("R = ", sol[0, 2])
44  print("")
45
46  print("At t=200:")
47  print("S = ", sol[-1, 0])
48  print("I = ", sol[-1, 1])
49  print("R = ", sol[-1, 2])
50  print("")
51
52
53  Imax = np.max(sol[:, 1])
54  print("Maximum value of I = ", Imax)
```




```
At t=0:
S =  9999.0
I =  1.0
R =  0.0

At t=200:
S =  595.1358608053775
I =  0.02864570401681712
R =  9404.835493490615

Maximum value of I =  3004.8159818370027
```

Question 7

For the same SIR considered above. Compute the reproductive number and the effective reproductive number. Would you expect the disease to be epidemic? If so, does this match the numerical results? What do you think is a disease with a similar behavior? Please compute analytically the peak number of infections and compare with simulations. Also, please compute the steady-state number of susceptible individuals by solving the associated implicit equation and compare with numerical results.

$$R_o = \frac{\beta}{\gamma} = \frac{0.3}{0.1} = 3 \tag{24}$$

$$R_e = \frac{S_o}{N} * \frac{\beta}{\gamma} = \frac{9999}{10000} * \frac{0.3}{0.1} = 2.9997 \tag{25}$$

Since R_e is very close to but slightly below 3, we would expect the disease to be epidemic. This matches the numerical results obtained by simulating the SIR model.

A disease with similar behavior to the SIR model with R0 = 3 and Re = 2.997 could be COVID-19 as inferred from $https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7657547/$.

Computing Imax analytically:

We have

$$\frac{I_{max}}{N} = 1 - \frac{1}{R_o} * (1 + ln(R_o) \tag{26}$$

$$R_o = \frac{\beta}{\gamma} = \frac{0.3}{0.1} = 3 \tag{27}$$

$$I_{max} = 3004.6 \tag{28}$$

This number matches our graph

computing the steady-state number of susceptible individuals using the equation:

$$\log\left(\frac{S_\infty}{N}\right) = R_0 \cdot (\frac{S_\infty}{N} - 1) \tag{29}$$

```python
import math
from scipy.optimize import root_scalar

def solve_for_S_inf(R0, N):
    def equation(S_inf):
        return math.log(S_inf/N) - R0 * ((S_inf-N)/N)

    sol = root_scalar(equation, bracket=[1, N/10])
    S_inf = sol.root

    return S_inf

R0 = 3
N = 10000
S_inf = solve_for_S_inf(R0, N)
print("S_inf =", S_inf)
```

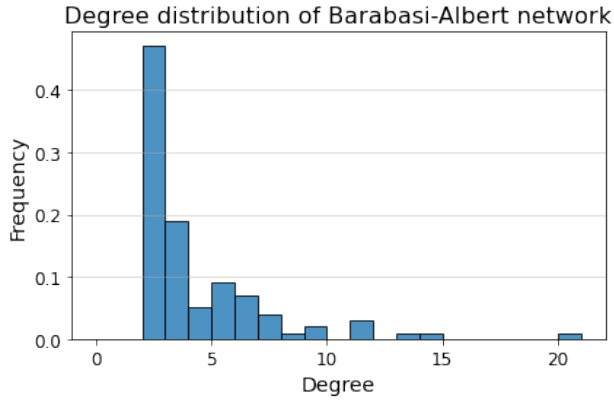
This gives a value close to the $S_{t=200}$ value we got from the simulation

$$S_{inf} = 595.20$$

Question 8

Create any scale-free network of your choice with at least 100 nodes. Plot the degree distribution and compute the mean degree $\langle k \rangle$ and the second moment of the degree $\langle k_2 \rangle$

```python
import networkx as nx
import matplotlib.pyplot as plt

ba = nx.barabasi_albert_graph(100, 2)

degrees = [ba.degree(node) for node in ba.nodes()]
plt.hist(degrees, bins=range(max(degrees)+2), density=True, alpha=0.8, edgecolor='black')
plt.xlabel('Degree', fontsize=14)
plt.ylabel('Frequency', fontsize=14)
plt.title('Degree distribution of Barabasi-Albert network', fontsize=16)
plt.tick_params(axis='both', labelsize=12)
plt.grid(axis='y', alpha=0.5)
plt.tight_layout()
plt.show()
```




```python
import numpy as np

mean_degree = np.mean(degrees)
print(f"Mean degree: {mean_degree:.2f}")

second_moment = np.mean(np.power(degrees, 2))
print(f"Second moment of degree: {second_moment:.2f}")
```

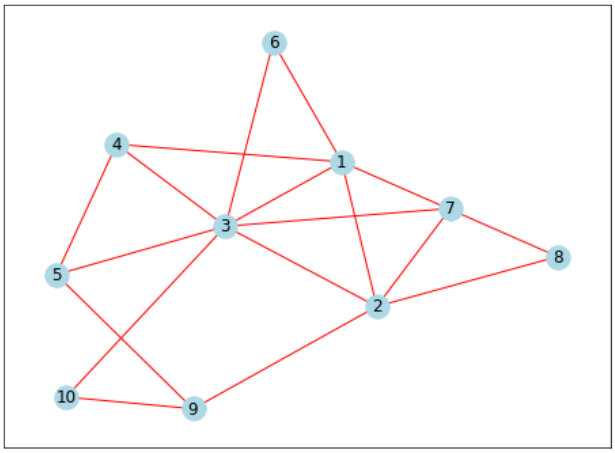
Giving an output of
Mean degree: 3.92 Second moment of degree: 24.66

Question 9

Consider the network depicted in Fig. 1, compute the searchability of the path $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$.

Given the graph, we can use the following formula to calculate the searchability of the path. Where x excludes the first and last node in the path.

```
#Plotting the network
import networkx as nx
import matplotlib.pyplot as plt

G = nx.Graph()
G.add_nodes_from([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
G.add_edges_from([(1, 2), (1, 3),(1, 4), (1, 6), (1, 8), (2, 3), (2, 7), (2, 8), (2, 9),
    (3, 4), (3, 5), (3, 6), (3, 7), (3, 10), (4, 5), (5, 9), (9, 10)])

fig, ax = plt.subplots(figsize=(8, 6))
pos = nx.spring_layout(G)
nx.draw_networkx(G, pos=pos, with_labels=True, node_color='lightblue', edge_color='red')
plt.show()
```



$$H_x = log(K_s) + \sum_{j \in \hat{x}} \log(K_j - 1)$$

Based on this we have

$$H_x = log(5) + log(5-1) + log(7-1)$$

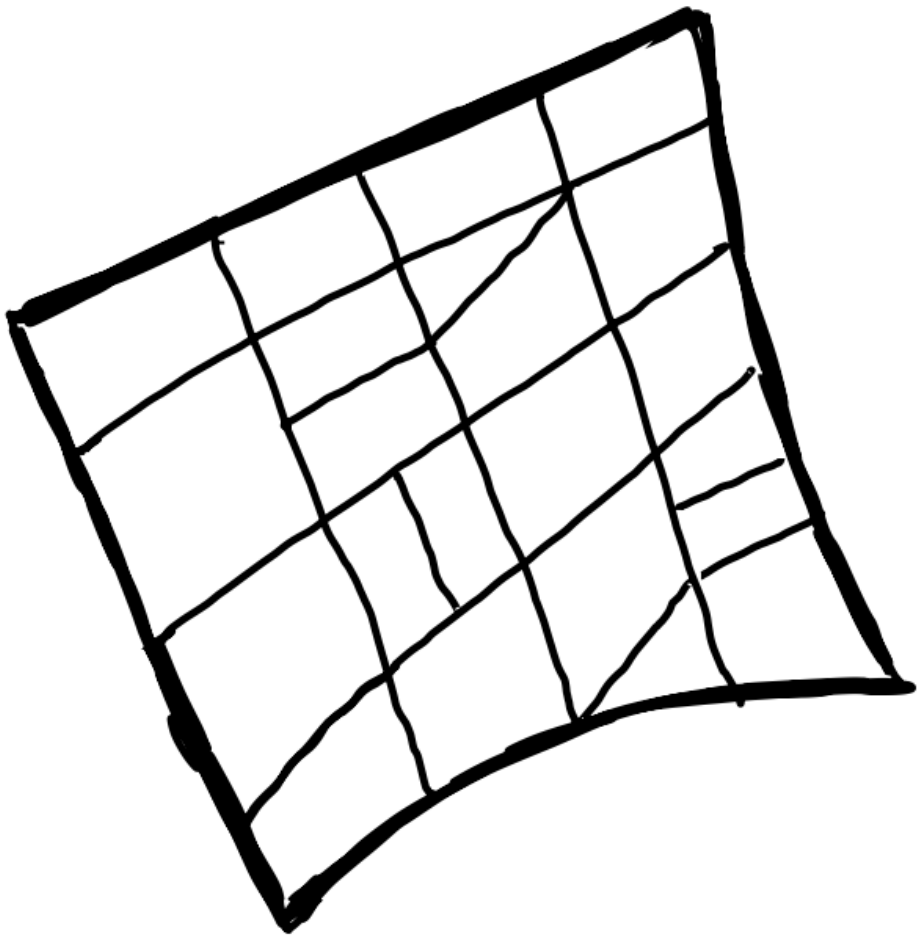
So,

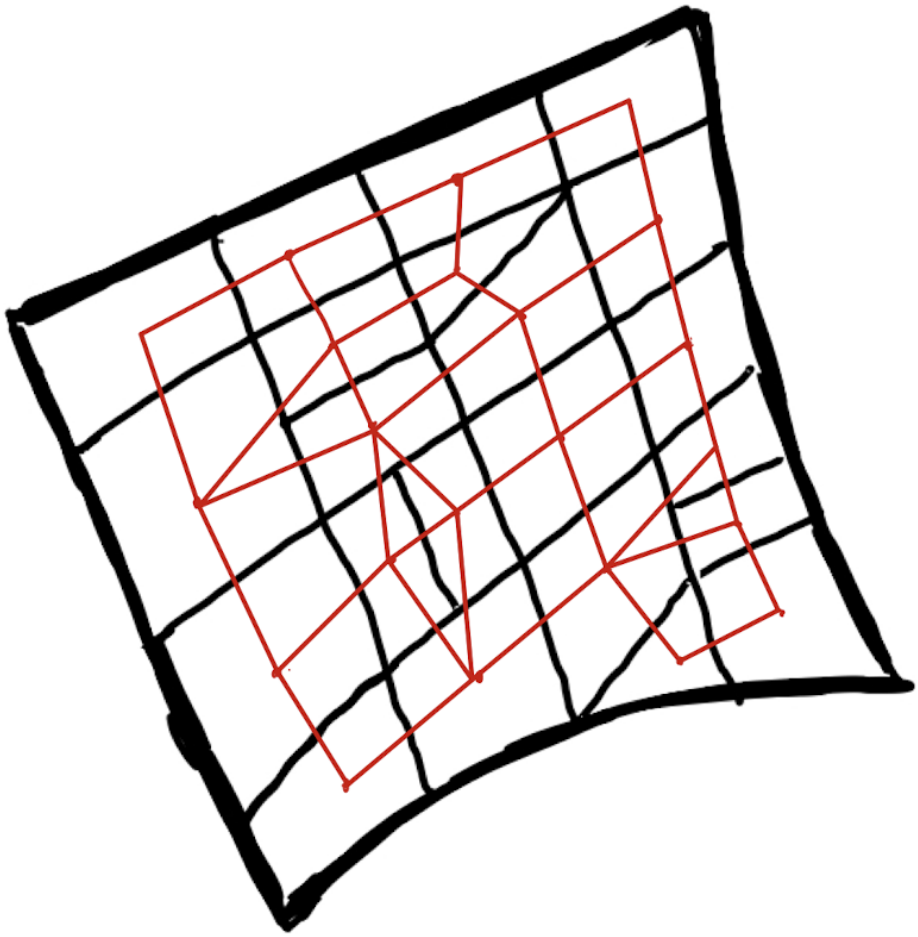
$$H_x = 4.78$$

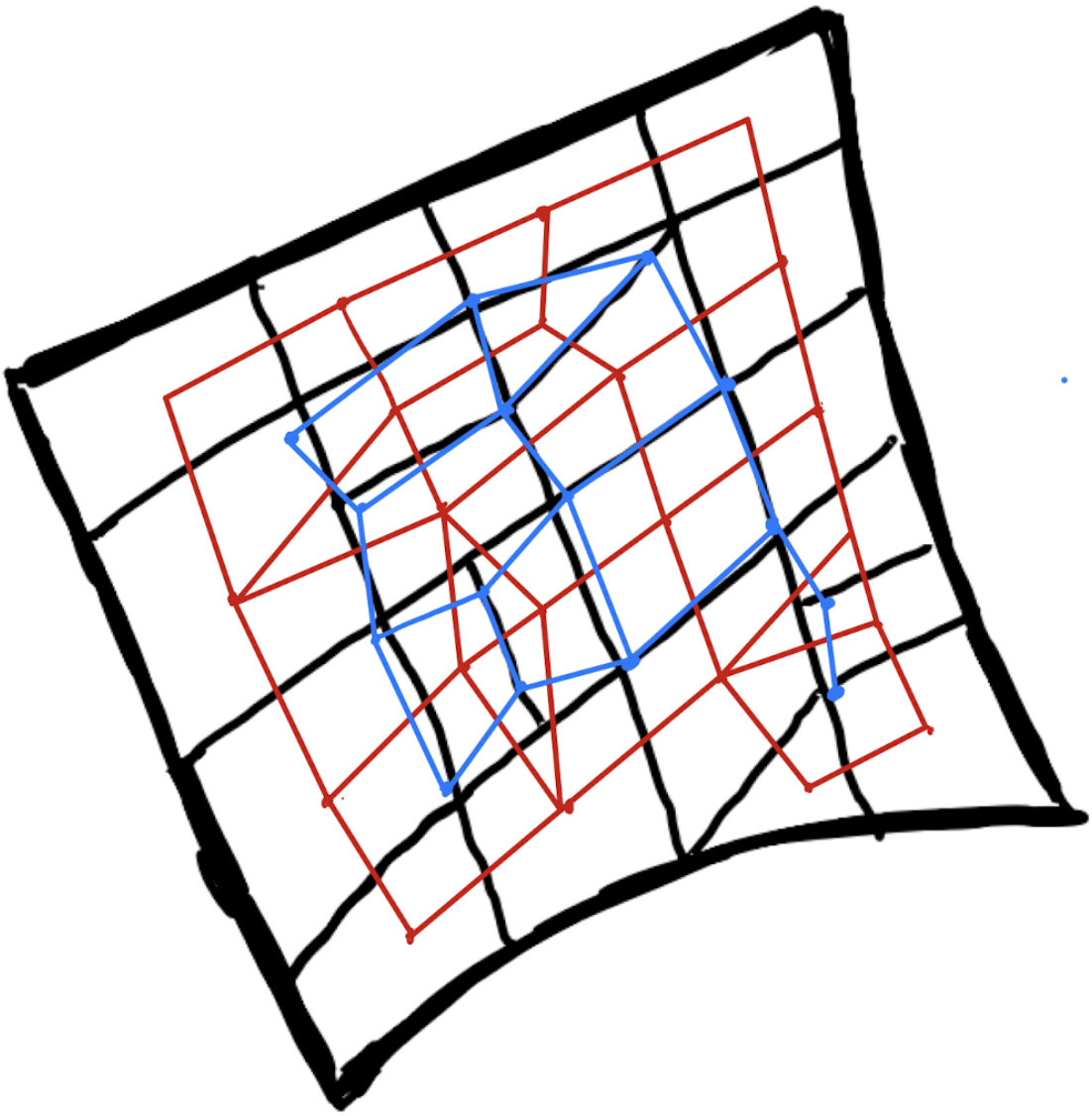
Question 10

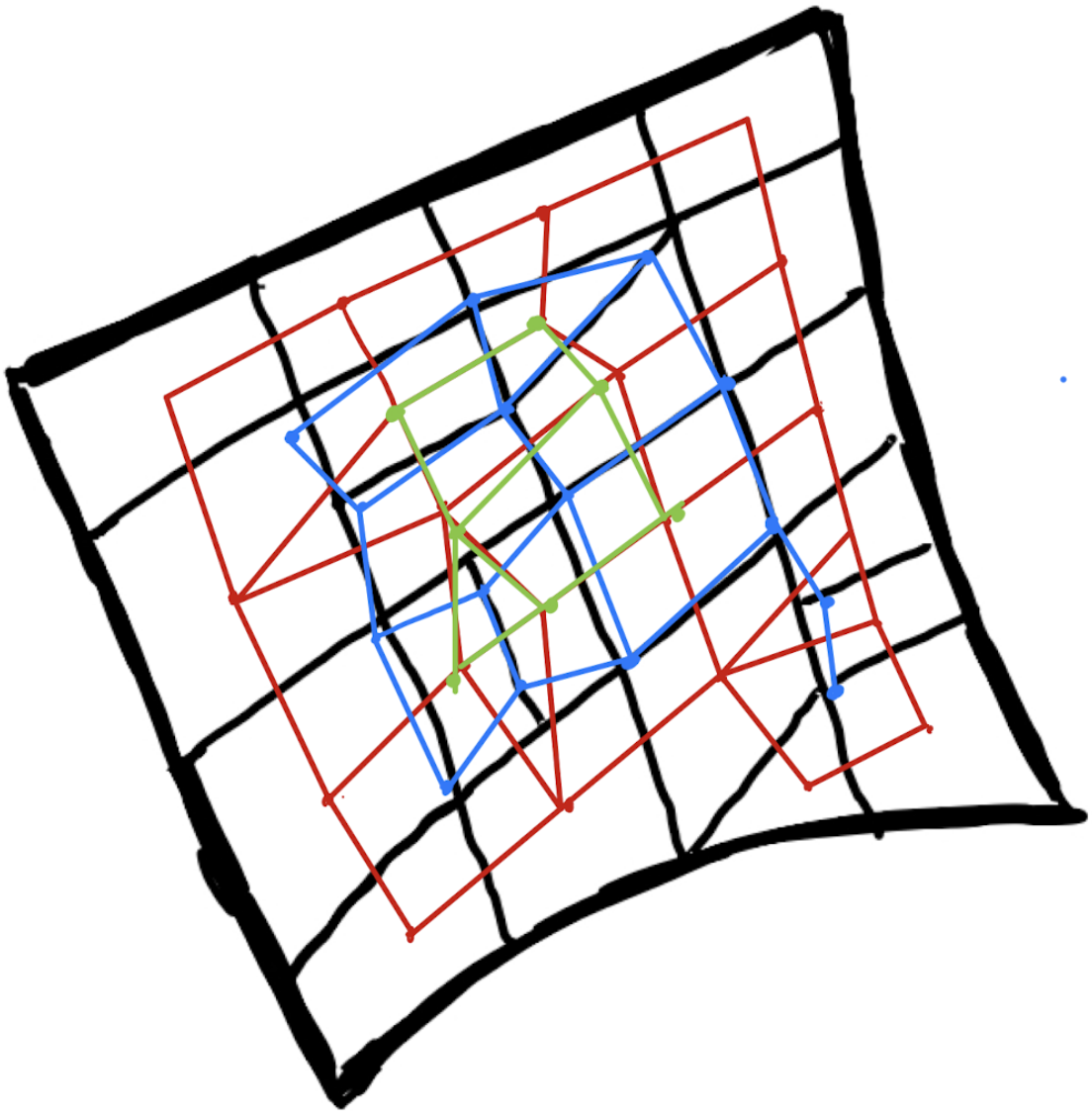
Consider the schematic of the block depicted in Fig. 2, what is the block complexity k_max?

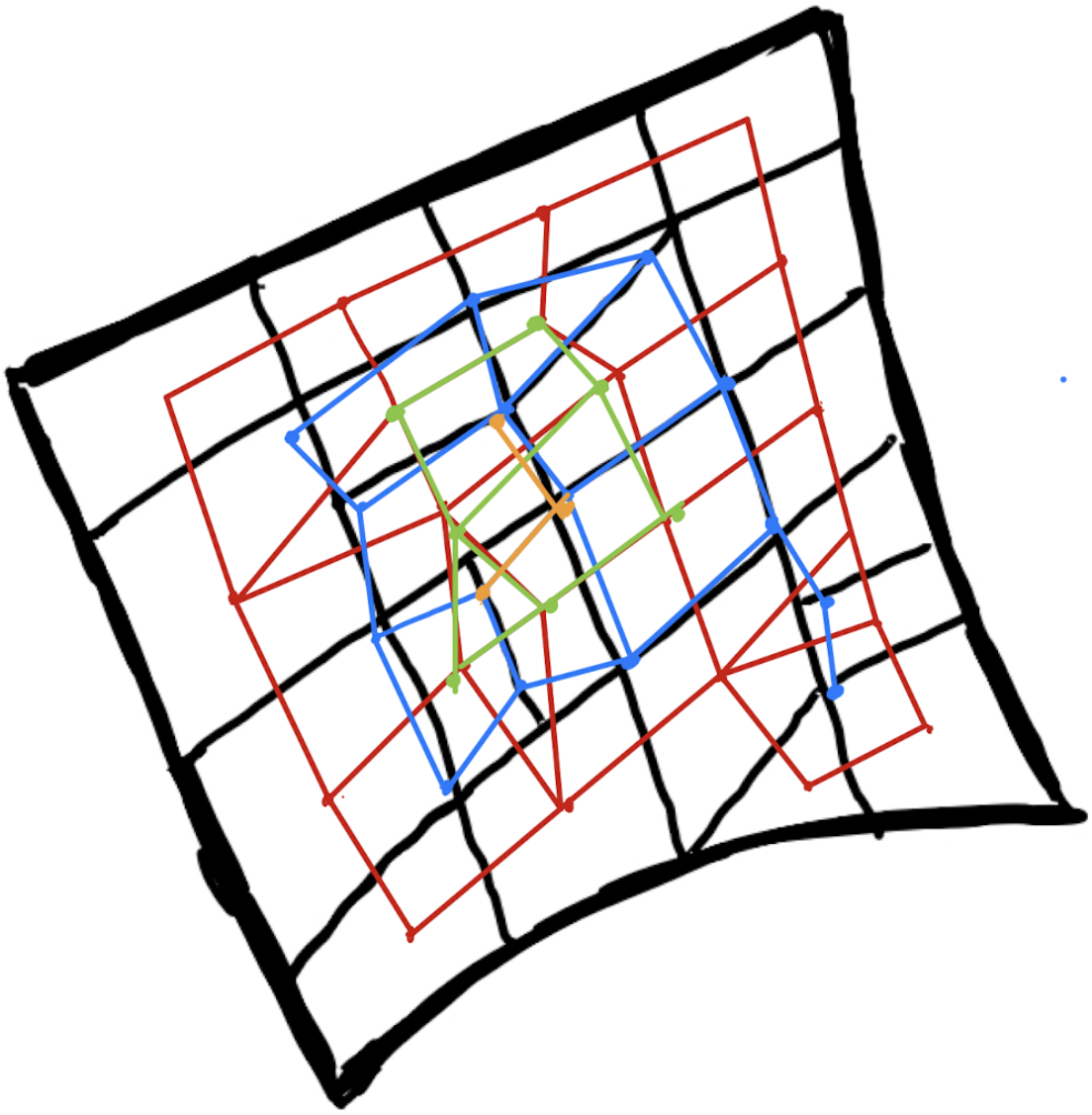
The following images show each iteration of connecting midpoints of adjacent blocks until a dot or line is obtained.











As we went through 4 iterations, the block complexity k_max is 4.