2.4

Additional Project
Linear Regression — Autocorrelation

Question 1

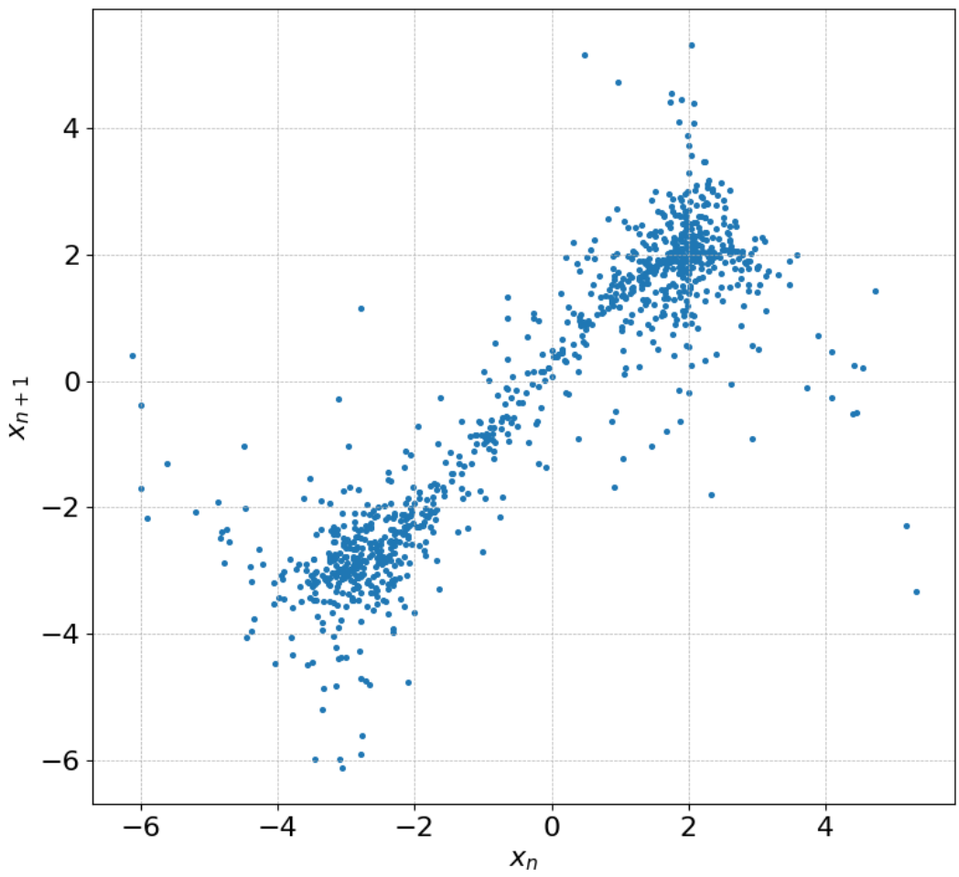


Figure 1: Scatter plot of x_{n+1} against x_n

A linear relationship between x_n seems to be feasible since the data is mostly focussed around a line of best fit of approximately $x_{n+1} = x_n$. However, there would be many anomalous points far from this line whose existence could not be explained by H_1.

Question 2

Use the model $\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$ where, taking $N = 999$,

$$\boldsymbol{Y} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix}, \quad \boldsymbol{X} = \begin{pmatrix} 1 & x_0 \\ 1 & x_1 \\ \vdots & \vdots \\ 1 & x_{N-1} \end{pmatrix}, \quad \boldsymbol{\beta} = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix}$$

Then,

$$\bar{\sigma}^2 = \frac{1}{N}\sum_{n=0}^{N-1}(x_{n+1} - \bar{a}_0 - \bar{a}_1 x_n)^2 \qquad \text{This is easy to derive.} \tag{1}$$

$$\bar{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y} \qquad \text{This is a standard result from lectures.}$$

$$\det(\boldsymbol{X}^T\boldsymbol{X}) = |\boldsymbol{X}^T\boldsymbol{X}| = N\sum_{n=0}^{N-1} x_n^2 - \left(\sum_{n=0}^{N-1} x_n\right)^2$$

$$(\boldsymbol{X}^T\boldsymbol{X})^{-1} = |\boldsymbol{X}^T\boldsymbol{X}|^{-1}\begin{bmatrix} \sum_{n=0}^{N-1} x_n^2 & -\sum_{n=0}^{N-1} x_n \\ -\sum_{n=0}^{N-1} x_n & N \end{bmatrix}$$

$$\implies \quad \bar{a}_0 = \frac{1}{|\boldsymbol{X}^T\boldsymbol{X}|}\left(\sum_{n=0}^{N-1} x_{n+1}\sum_{n=0}^{N-1} x_n^2 - \sum_{n=0}^{N-1} x_n \sum_{n=0}^{N-1} x_n x_{n+1}\right) \tag{2}$$

$$\bar{a}_1 = \frac{1}{|\boldsymbol{X}^T\boldsymbol{X}|}\left(N\sum_{n=0}^{N-1} x_n x_{n+1} - \sum_{n=0}^{N-1} x_n \sum_{n=0}^{N-1} x_{n+1}\right) \tag{3}$$

$$\begin{aligned}
\bar{\sigma}^2 &\sim \frac{\sigma^2}{N}\chi^2_{N-2} \qquad \text{From lectures} \\
\bar{\boldsymbol{\beta}} &\sim N_2(\boldsymbol{\beta}, \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}) \qquad \text{From lectures} \\
\implies \bar{a}_0 &\sim N\left(a_0, \sigma^2|\boldsymbol{X}^T\boldsymbol{X}|^{-1}\sum_{n=0}^{N-1} x_n\right) \\
\bar{a}_1 &\sim N\left(a_1, N\sigma^2|\boldsymbol{X}^T\boldsymbol{X}|^{-1}\right)
\end{aligned}$$

Using equations (1), (2) and (3), we calculate the following values (to 3s.f.),

$$\begin{aligned}
\bar{a}_0 &= -0.0193 \\
\bar{a}_1 &= 0.893 \\
\bar{\sigma}^2 &= 1.08
\end{aligned}$$

Question 3

- For $a_1 < 0$ (and a_0 not too large in the case $a_1 > -1$) x_n will oscillate between positive and negative values. This is illustrated in Figure 2.

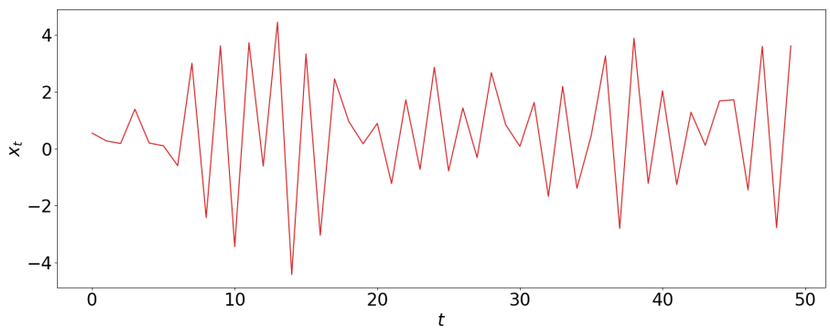


Figure 2: Behaviour of x_n with $a_0 = 1$, $a_1 = -0.8$, $\sigma^2 = 1$

- For $|a_1| > 1$, x_n diverges almost surely as $n \to \infty$. We know this is true because it is true of the random walk when $a_1 = 1$ and similarly for $a_1 = -1$. Then $|a_1| > 1$ will only act to increase the growth of x_n so the divergent behaviour will only be augmented.
- For $|a_1| < 1$, x_n will almost surely not diverge and so the data will be focussed in a finite region as $n \to \infty$. This is the case because for $|x_n|$ sufficiently large it is almost certain that $|x_{n+1}| < |x_n|$. Despite this, the larger the magnitude of a_1, the more opportunity there is for x_n to grow and so there will be a wider variance in the values of the sequence (x_n).

Question 4

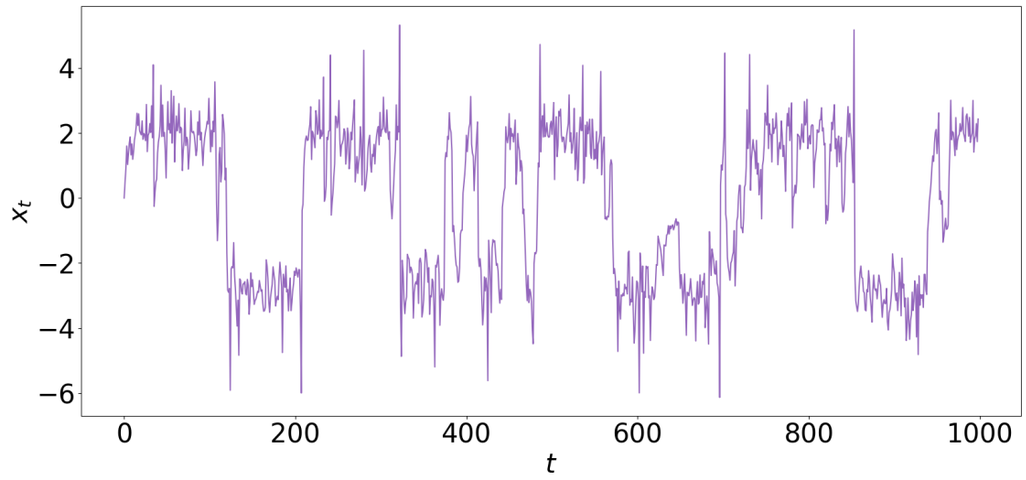


Figure 3: Behaviour of x_n in autocrrelation.dat

From Figure 3, we can see that x_n is not diverging so we predict $|a_1| < 1$. Then, using Figure 1, we can predict that $a_1 > 0.5$ since a_1 represents the gradient of the line of best fit and 0.5 is the gradient of the line between the points $(-4, -2)$ and $(4, 2)$. So take $R = (0.5, 1)$. From Question 2,

$$\bar{a}_1 \sim N\left(a_1, \sigma^2 |\boldsymbol{X}^T\boldsymbol{X}|^{-1} N\right)$$

$$\text{Let, } \mathrm{SE}(\bar{a}_1) = \sqrt{\tilde{\sigma}^2 |\boldsymbol{X}^T\boldsymbol{X}|^{-1} N} \text{ where } \tilde{\sigma}^2 = \text{ RSS } /(N-2) = N\bar{\sigma}^2/(N-2).$$

$$\text{Then, } \frac{\bar{a}_1 - a_1}{\mathrm{SE}(\bar{a}_1)} = \frac{\bar{a}_1 - a_1}{\sqrt{\tilde{\sigma}^2 N |\boldsymbol{X}^T\boldsymbol{X}|^{-1}}} = \frac{(\bar{a}_1 - a_1)/\sqrt{\sigma^2 N |\boldsymbol{X}^T\boldsymbol{X}|^{-1}}}{\sqrt{N\bar{\sigma}^2/((N-2)\sigma^2)}}$$

Observe that the numerator is a standard normal N(0,1) and the denominator is an independent $\sqrt{\chi^2_{N-2}/(N-2)}$. Thus,

$$\frac{\bar{a}_1 - a_1}{\mathrm{SE}(\bar{a}_1)} \sim t_{N-2}$$

$$\implies a_1 \sim \bar{a}_1 - \mathrm{SE}(\bar{a}_1) t_{N-2}$$

$$\mathrm{SE}(\bar{a}_1) = 0.01427, \bar{a}_1 = 0.893$$

$$\text{Then, } t_{N-2} = 27.5, \ t_{N-2} = -7.47 \text{ for } \bar{a}_1 - \mathrm{SE}(\bar{a}_1) t_{N-2} = 0.5 \text{ and } 1 \text{ respectively.}$$

$$P(-7.47 \leq t_{997} \leq 27.5) = 1$$

And so this is the probability a_1 lies in R.

Question 5

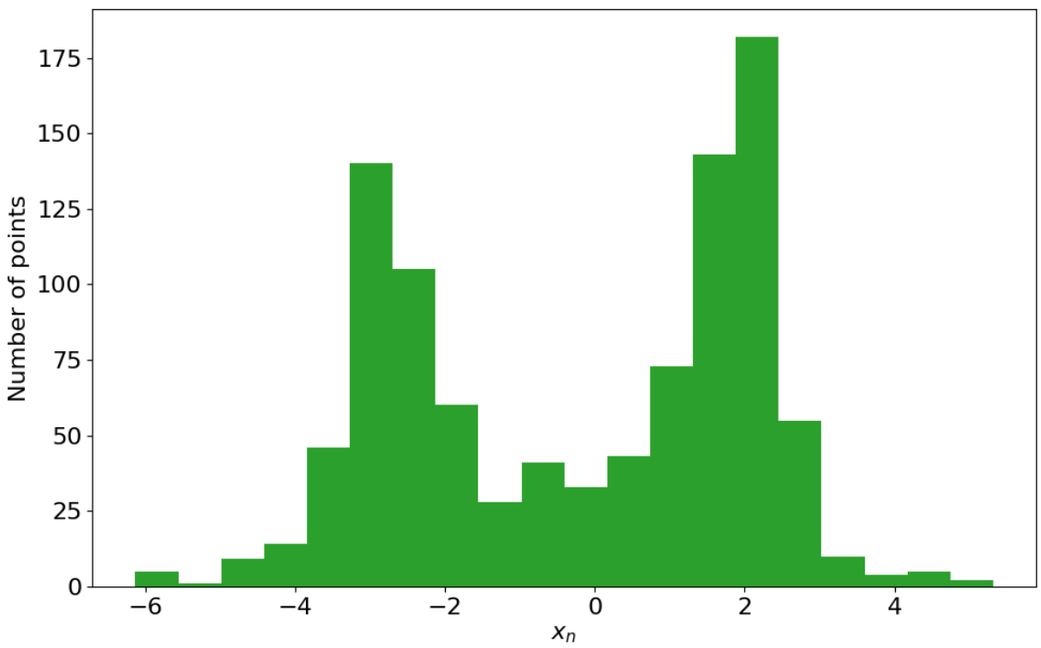


Figure 4: Histogram of x_n

Using the sum of normal distributions and the fact that if $X \sim N(\mu, \sigma^2)$ then $aX \sim N(a\mu, (a\sigma)^2)$ we obtain,

$$\begin{aligned} X_n \sim & N\left(a_0\left(1 + a_1 + ... + a_1^{n-1}\right), \sigma^2\left(1 + a_1^2 + ... + a_1^{2(n-1)}\right)\right) \qquad \text{For } n \geq 1 \\ \implies & X_n \sim N\left(a_0\frac{1-a_1^n}{1-a_1}, \sigma^2\frac{1-a_1^{2n}}{1-a_1^2}\right) \\ \implies & X_n \sim N\left(\frac{a_0}{1-a_1}, \frac{\sigma^2}{1-a_1^2}\right) \qquad \text{For } n \text{ large and } |a_1| < 1 \end{aligned}$$

Thus for our data, since there are 1000 data points, n is large for most x_n so we should expect the histogram to resemble a normal distribution. However, Figure 4 has two separate large peaks so is very unlikely to have been caused by a normal distribution of data. Thus the hypothesis H_1 no longer seems reasonable.

Question 6

Using the model $\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, we have,

$$\begin{aligned} \tilde{\sigma}^2 &= \frac{1}{N}\sum_{n=0}^{N-1}(x_{n+1} - \tilde{a}_0 - \tilde{a}_1 x_n - \tilde{a}_2 x_n^2)^2 \\ \tilde{\boldsymbol{\beta}} &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y} \end{aligned}$$

Then the values to 3s.f. are,

$$\begin{aligned} \tilde{a}_0 &= 0.146 \\ \tilde{a}_1 &= 0.865 \\ \tilde{a}_2 &= -0.0318 \\ \tilde{\sigma}^2 &= 1.062 \end{aligned}$$

Question 7

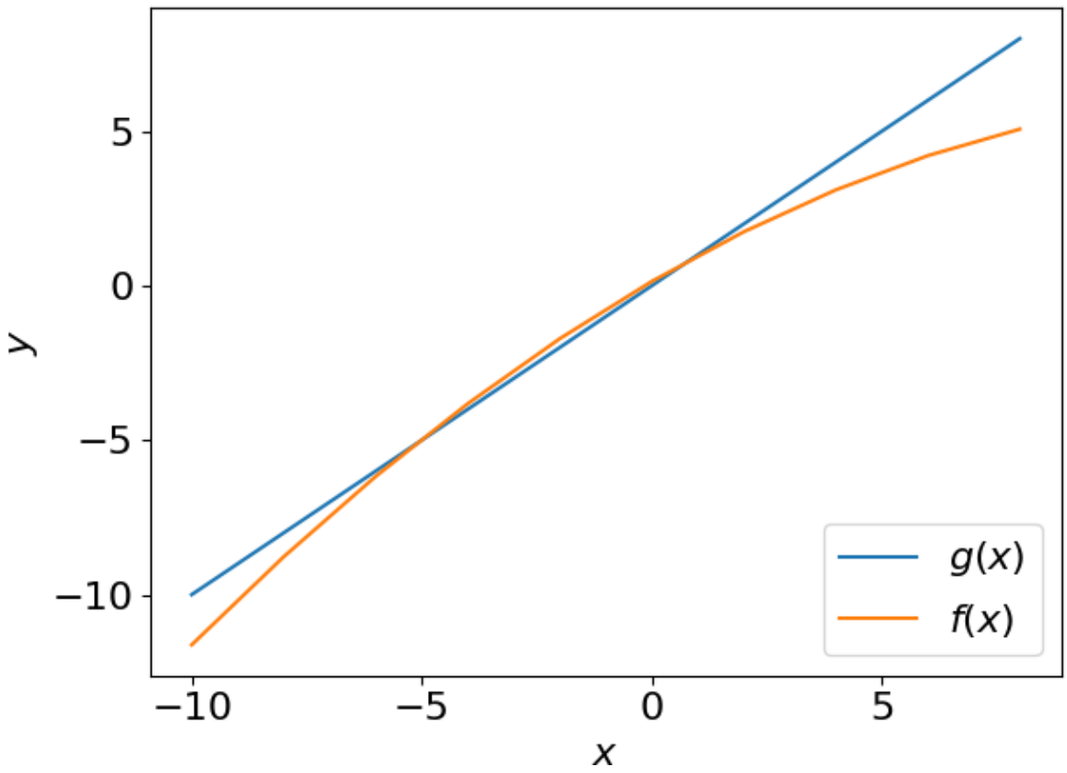


Figure 5: Plots of $y = f(x)$ and $y = g(x)$

Solving $\bar{a}_2x^2 + \bar{a}_1x + a_0 = x$ with the quadratic formula, we obtain the roots,

$$x = -5.14, \quad x = 0.890$$

$x = -5.14$ comes before the first large peak on the histogram in Figure 4 while $x = 0.890$ is at the start of the second peak. $x = 0.890$ is an attractor of the system $x_{n+1} = f(x_n)$. Then, as the data is drawn to 0.890, the $N(0, \sigma^2)$ term will mean you should expect to see the data split in half above and below this fixed point which is exactly the case in the histogram.

Question 8

$$f'(x) = 0.865 - 0.0637x$$
$$\text{In particular, } f'(-5.14) = 1.19, \quad f'(0.890) = 0.808$$

A fixed point t of $x_{n+1} = f(x_n)$ is stable if $|f'(t)| < 1$ and unstable if $|f'(t)| > 1$. Therefore, the chain x_n is expected to approach 0.865 and diverge away from -5.17. Then, with this model we would most likely see a single peak at $x = 0.865$. This is further supported by the fact that the curve $y = f(x)$ in the range $-6 < x < 5$ can be approximated by a straight line. We find that the line $y = h(x) = 0.897x - 0.168$ (found by minimising the MSE between itself and a sample generated with $f(x)$) is a good approximation as shown in Figure 6.

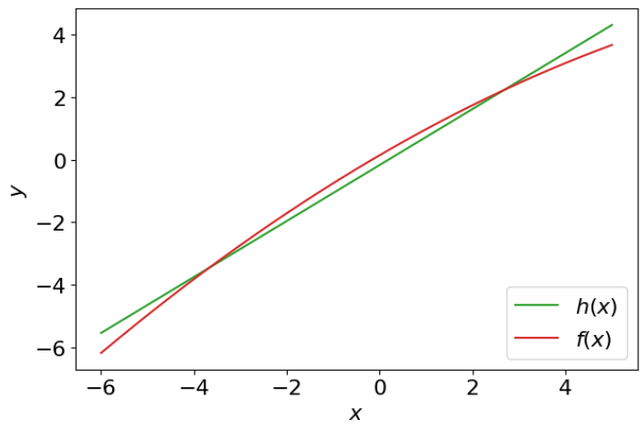


Figure 6: Plots of $y = f(x)$ and $y = h(x)$ in range $(-6, 5)$

Thus H_2 approximates to a linear model. Then as with the hypothesis H_1, you would expect a normally distributed histogram with the peak at the fixed point $x = 0.865$ but this is not the case. Thus H_2 is also implausible.

Question 9

Again using the model $\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, we have,

$$\begin{aligned}
\hat{a}_0 &= 0.315 \\
\hat{a}_1 &= 1.25 \\
\hat{a}_2 &= -0.101 \\
\hat{a}_3 &= -0.0474 \\
\hat{\sigma}^2 &= \frac{1}{N}\sum_{n=0}^{N-1}(x_{n+1} - \bar{a}_0 - \bar{a}_1 x_n - \bar{a}_2 x_n^2 - \hat{a}_3 x_n^3)^2 \\
&= 0.560
\end{aligned}$$

$$\bar{a}_i \sim N\left(a_1, \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}_{ii}\right)$$

Then $\sigma_0^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}_{ii}$ is an unbiased variance estimator if $\mathbb{E}(\sigma_0^2) = \sigma^2$.

This is true for $\sigma_0^2 = \text{RSS}/(N-4) = 0.562$

$$(\boldsymbol{X}^T\boldsymbol{X})^{-1}_{11} = 2.51 \times 10^{-3},\ (\boldsymbol{X}^T\boldsymbol{X})^{-1}_{22} = 5.31 \times 10^{-4},$$
$$(\boldsymbol{X}^T\boldsymbol{X})^{-1}_{33} = 6.34 \times 10^{-5},\ (\boldsymbol{X}^T\boldsymbol{X})^{-1}_{44} = 4.48 \times 10^{-6}$$

Thus we have the following unbiased estimators,

$\text{var}(\hat{a}_1) = 1.41 \times 10^{-3}$, $\text{var}(\hat{a}_2) = 2.99 \times 10^{-4}$, $\text{var}(\hat{a}_3) = 3.56 \times 10^{-5}$, $\text{var}(\hat{a}_4) = 2.52 \times 10^{-6}$

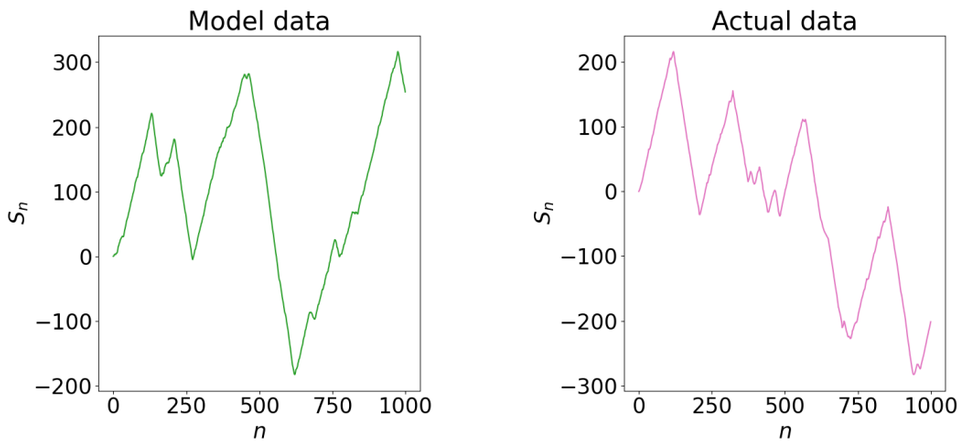
Question 10

The distribution function of an exponential random variable with mean 2 is $F(x) = 1 - e^{-x/2}$. Using the inverse of this function, we can generate a sample of 1000 exponential random variables. An example output to the program is shown below:

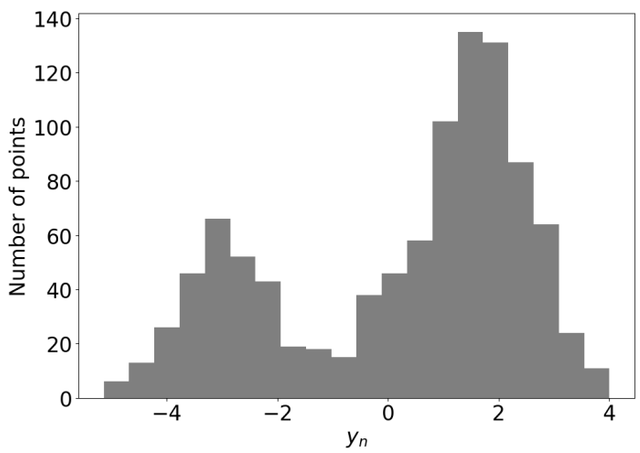
```
Uniform mean =  0.4980221236530167
Uniform variance =  0.0832700785786884
Exponential mean =  1.9994326225147219
Exponential variance =  4.0810734120780605
```

Question 11

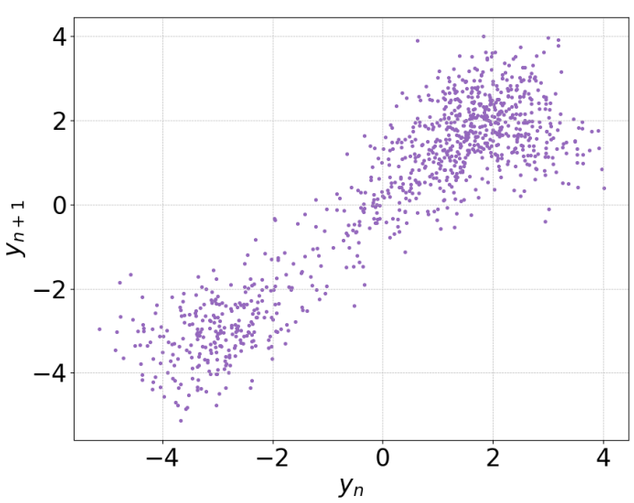
(i)



(ii)



(iii)



The histogram in Figure 4 shows resemblance to the histogram in (ii) as they both have peaks at -3 and 2. Likewise, the plot in Figure 1 is similar to diagram (iii) as both have a concentration of points at (-3, -3) and (2,2). Due to the random nature of the model, we do not expect the plots in (i) to match. Nevertheless, they are similar in that they have sharp up and downward spikes. This all provides evidence to support the model as any good model should have these similarities.

On the other hand, for (i) the value of S_{1000} is about 250 for the model data and about -200 for the actual data which is a large discrepancy. Running the model multiple times, I find that we almost always have a positive value of S_{1000} suggesting that there are too many large y_i; this is supported by the fact that the second peak in (ii) is much larger than the first which is not the case in Figure 4. In addition, the points in (iii) are more spread out than in Figure 1. Therefore, overall I find the model to be unsatisfactory as one would expect these properties to be more likely to be preserved.

question_1.py for question 1

```
import matplotlib.pyplot as plt

def get_data():
    file = open('autocorrelation.dat.txt', 'r')

    data_point = ''
    data_set = []

    for character in file.read():
        if character == ',':
            data_point = float(data_point)
            data_set.append(data_point)
            data_point = ''
        else:
            data_point += character
    data_point = float(data_point)
    data_set.append(data_point)  # Adding the last data point in

    return data_set

if __name__ == '__main__':
    data_set = get_data()

    data_set_x = data_set.copy()
    data_set_x.pop(-1) # the x_n values
    data_set_y = data_set.copy()
    data_set_y.pop(0) # the y_n values

    plt.rc('font', size = 16)
    plt.grid(linestyle = '--', linewidth = 0.5)
    plt.scatter(data_set_x, data_set_y, s = 7)
    plt.xlabel('$x_{n}$')
    plt.ylabel('$x_{n+1}$')
    plt.show()

    print(len(data_set))
```

question_2.py for question 2

```
from question_1 import get_data
import numpy as np

data_set = np.array(get_data())

N = 999
sum1 = sum(data_set[:-1])  # sum of x_n from 0 to 998
sum2 = sum(data_set[1:])  # sum of x_n from 1 to 999
sum3 = np.dot(data_set[:-1], data_set[:-1])
# sum of x_n^2 from 0 to 998
sum4 = np.dot(data_set[:-1], data_set[1:])  # sum of x_nx_n+1
determinant = N*sum3 - sum1**2
print(determinant)

a_0 = (1 / determinant)*(sum2*sum3 - sum1*sum4)
a_1 = (1 / determinant)*(N*sum4 - sum1*sum2)
sigma_sq = (1 / N)*np.dot(data_set[1:] - a_0 - a_1 *
    data_set[:-1], data_set[1:] - a_0 - a_1 * data_set[:-1])

print('a_0 = ', a_0)
print('a_1 = ', a_1)
print('sigma^2 = ', sigma_sq)
```

question_3.py for question 3

```
import math
import numpy as np
import matplotlib.pyplot as plt
import math
from question_10 import generate_uniform, generate_exponential

# simulating H_1
uniform_sample = generate_uniform()
exp_sample = generate_exponential()
vec_cos = np.vectorize(math.cos)
normal_sample = np.sqrt(exp_sample) * \
        vec_cos(2*math.pi*uniform_sample)

x_n = 0
x_list = []
a_0 = 1
a_1 = -0.8
for i in range(50):
    x_n = a_0 + a_1*x_n + normal_sample[i]
    x_list.append(x_n)

plt.rc('font', size = 24)
plt.plot(x_list, color = 'C3')
plt.xlabel('$t$')
plt.ylabel('$x_{t}$')
plt.show()

plt.plot(normal_sample)
plt.show()
```

question_4.py for question 4

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

from question_1 import get_data

N = 999
data_set = np.array(get_data())
sum1 = sum(data_set[:-1])  # sum of x_n from 0 to 999
sum2 = sum(data_set[1:])  # sum of x_n from 1 to 1000
sum3 = np.dot(data_set[:-1], data_set[:-1])
# sum of x_n^2 from 0 to 999
sum4 = np.dot(data_set[:-1], data_set[1:])  # sum of x_nx_n+1

plt.rc('font', size = 30)
plt.plot(data_set, color = 'C4')
plt.xlabel('$t$')
plt.ylabel('$x_{t}$')
plt.show()

determinant = N*sum3 - sum1**2
a_0 = (1 / determinant)*(sum2*sum3 - sum1*sum4)
a_1 = (1 / determinant)*(N*sum4 - sum1*sum2)

RSS = np.dot(data_set[1:] - a_0 - a_1 * data_set[:-1],
                data_set[1:] - a_0 - a_1 * data_set[:-1])
# ^ (notice this is N(sigma_bar)^2)
sigma_tilde_sq = RSS / (N-2)
SE = np.sqrt(sigma_tilde_sq*N/determinant)

print(sigma_tilde_sq)
print(SE)
```

question_5.py for question 5

```
from question_1 import get_data

import numpy as np
import matplotlib.pyplot as plt

data_set = np.array(get_data())

plt.rc('font', size = 16)
plt.hist(data_set, bins = 50, color = 'C2')
plt.xlabel('$x_{n}$')
plt.ylabel('Number of points')
plt.show()
```

question_6.py for question 6

```
import numpy as np
from numpy.linalg import inv

from question_1 import get_data

data_set = np.array(get_data())
N = 999
sum1 = sum(data_set[:-1])
sum2 = sum(data_set[:-1]**2)
sum3 = sum(data_set[:-1]**3)
sum4 = sum(data_set[:-1]**4)

XTX = np.array([N, sum1, sum2, sum1, sum2, sum3,
                sum2, sum3, sum4]).reshape((3, 3))
XTX_inv = inv(XTX)
XTY = np.array([sum(data_set[1:]), np.dot(data_set[:-1],
    data_set[1:]), np.dot(data_set[:-1]**2, data_set[1:])])

beta = np.dot(XTX_inv, XTY)
a_0, a_1, a_2 = beta[0], beta[1], beta[2]
sigma_sq = sum((data_set[1:] - a_0 - a_1*data_set[:-1]
                - a_2*data_set[:-1]**2)**2) / N

print('a_0 = ', a_0)
print('a_1 = ', a_1)
print('a_2 = ', a_2)
print('sigma^2 = ', sigma_sq)
```

question_7.py for question 7

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

a_0 = 0.1456657383510946
a_1 = 0.8647726496931831
a_2 = -0.03182985839655377

f_x = lambda x : a_0 + a_1*x + a_2*x**2
vf = np.vectorize(f_x)

x = np.linspace(-10, 10, 1000)

print(a, b)

if __name__ == '__main__':
    plt.rc('font', size = 16)
    plt.plot(x, color = 'C0', label = '$h(x)$')
    plt.plot(x, vf(x), color = 'C1', label = '$f(x)$')
    plt.legend(loc = 'lower right')
    plt.xlabel(r'$x$')
    plt.ylabel(r'$y$')
    plt.show()
```

question_8.py for question 8

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

a_0 = 0.1456657383510946
a_1 = 0.8647726496931831
a_2 = -0.03182985839655377

f_x = lambda x : a_0 + a_1*x + a_2*x**2
vf = np.vectorize(f_x)

x = np.linspace(-6, 5, 1000)
curve_data = vf(x)
a, b = np.polyfit(x, curve_data, 1)

print(a,b)

h_x = lambda x: a*x + b
vh = np.vectorize(h_x)

if __name__ == '__main__':
    plt.rc('font', size = 16)
    plt.plot(x, vh(x), color = 'C2', label = '$h(x)$')
    plt.plot(x, vf(x), color = 'C3', label = '$f(x)$')
    plt.legend(loc = 'lower right')
    plt.xlabel(r'$x$')
    plt.ylabel(r'$y$')
    plt.show()
```

question_9.py for question 9

```
import numpy as np
from numpy.linalg import inv

from question_1 import get_data

data_set = np.array(get_data())
N = 999
sum1 = sum(data_set[:-1])  # sum of x_n from 0 to 998
sum2 = sum(data_set[:-1]**2)
sum3 = sum(data_set[:-1]**3)
sum4 = sum(data_set[:-1]**4)
sum5 = sum(data_set[:-1]**5)
sum6 = sum(data_set[:-1]**6)

XTX = np.array([N, sum1, sum2, sum3, sum1, sum2, sum3, sum4,
                sum2, sum3, sum4, sum5, sum3, sum4, sum5,
                sum6]).reshape((4, 4))
XTX_inv = inv(XTX)
print(XTX_inv)
XTY = np.array([sum(data_set[1:]), np.dot(data_set[:-1],
        data_set[1:]), np.dot(data_set[:-1]**2, data_set[1:]),
        np.dot(data_set[:-1]**3, data_set[1:])])

beta = np.dot(XTX_inv, XTY)
a_0, a_1, a_2, a_3 = beta[0], beta[1], beta[2], beta[3]
sigma_sq = sum((data_set[1:] - a_0 - a_1*data_set[:-1]
        - a_2*data_set[:-1]**2 - a_3*data_set[:-1]**3)**2) / N

print('a_0 = ', a_0)
print('a_1 = ', a_1)
print('a_2 = ', a_2)
print('a_3 = ', a_3)
print('sigma^2 = ', sigma_sq)
```

question_10.py for question 10

```
import numpy as np
import random
import math


def F_inverse(x):
    return -2 * math.log(1-x)
vF_inv = np.vectorize(F_inverse)

N = 1000

def generate_uniform():
    uniform_random_sample = []
    for i in range(1000):
        point = random.random()
        uniform_random_sample.append(point)
    return np.array(uniform_random_sample)

def generate_exponential():
    return vF_inv(generate_uniform())

if __name__ == '__main__':
    uniform_sample = generate_uniform()
    exp_sample = generate_exponential()

    uniform_mean = sum(uniform_sample) / N
    exp_mean = sum(exp_sample) / N

    uniform_variance = sum(uniform_sample**2) \
                        / N - uniform_mean**2
    exp_variance = sum(exp_sample**2) / N - exp_mean**2

    print('Uniform mean = ', uniform_mean,
        '\nUniform variance = ', uniform_variance)
    print('Exponential mean = ', exp_mean,
        '\nExponential variance = ', exp_variance)
```

question_11.py for question 11

```
import numpy as np
import math
from matplotlib import pyplot as plt
from question_1 import get_data
from question_10 import generate_uniform, generate_exponential

uniform_sample = generate_uniform()
exp_sample = generate_exponential()

vec_cos = np.vectorize(math.cos)

normal_sample = np.sqrt(exp_sample) * \
    vec_cos(2*math.pi*uniform_sample)

a_0 = 0.3152583364332162
a_1 =  1.2525659122769783
a_2 =  -0.10077141891440472
a_3 =  -0.047440285420029184
sigma_sq = 0.6

model_normal_sample = math.sqrt(0.6)*normal_sample

y_n = 0
y_list = [0]
for i in range(999):
    y_n = a_0 + a_1*y_n + a_2*y_n**2 + a_3*y_n**3 + \
                    model_normal_sample[i]
#     if y_n < -5.5:
#         y_n = 0
    y_list.append(y_n)

S_n = 0
Sn_list = []
for n in range(1000):
    S_n += y_list[n]
    Sn_list.append(S_n)

xS_n = 0
xSn_list = []
data_set = get_data()
for n in range(1000):
    xS_n += data_set[n]
    xSn_list.append(xS_n)

y_list_x = y_list.copy() # for diagram (iii)
y_list_x.pop(-1)
y_list_y = y_list.copy()
y_list_y.pop(0)

plt.rc('font', size = 24)

plt.figure(1) # plot of S_n
plt.grid(linestyle = '--', linewidth = 0.5)
plt.subplot2grid(shape = (1, 5), loc = (0, 0), colspan = 2)
plt.plot(Sn_list, color = 'C2')
plt.xlabel('$n$')
```

```
plt.ylabel('$S_{n}$')
plt.title('Model data')
plt.subplot2grid(shape = (1, 5), loc = (0, 3), colspan = 2)
plt.plot(xSn_list, color = 'C6')
plt.xlabel('$n$')
plt.ylabel('$S_{n}$')
plt.title('Actual data')

plt.figure(2) # histogram
plt.hist(y_list, bins = 20, color = 'C7')
plt.xlabel('$y_{n}$')
plt.ylabel('Number of points')

plt.figure(3) # y_{n+1} against y_{n}
plt.grid(linestyle = '--', linewidth = 0.5)
plt.scatter(y_list_x, y_list_y, s = 7, color = 'C4')
plt.xlabel('$y_{n}$')
plt.ylabel('$y_{n+1}$')

plt.show()
```