STOCHASTIC PROCESSES

HW03

Hanie Hatami(99100614)
Stochastic processes course

Contents

1 Abstract

In this exercise, we want to check the second-order stationarity of a data. We have used two data, one is normal data generated by ourselves and the other is Autoregressive data. To check the second-order stationary, for first method we must first separate the data into windows of length L, where L is from 3(because variance can be obtained from 3 numbers of data) to 1/5 of number of data and obtain the variance of each window, then take the average of the variance of windows. Then draw the plot of average variances according to L. If this graph converges towards a number, then we can say that this data is second-order stationary(because we have used variance) and the length of the second-order stationary is the point where the plot starts to converge.
The second method is to check the second-order stationarity of the data at different levels of confidence through p-value.

2 Results

First, let's go to the normal data with a different number of data to check the stationery second order for them and get its length.

2.1 Normal distribution

2.1.1 Normal distribution n=10^2

2.1.1.1 First method

For 10^2 data, the length of the second-order stationary is 26. (This value will be output to you if you run the code for the number of data you want.)
In order to show a better plot when drawing the error bars, in addition to drawing the original data, we have also drawn a smoother data along with the error bars, which we obtained from summarizing the original data. With less numbers of data, these two plots are identical.

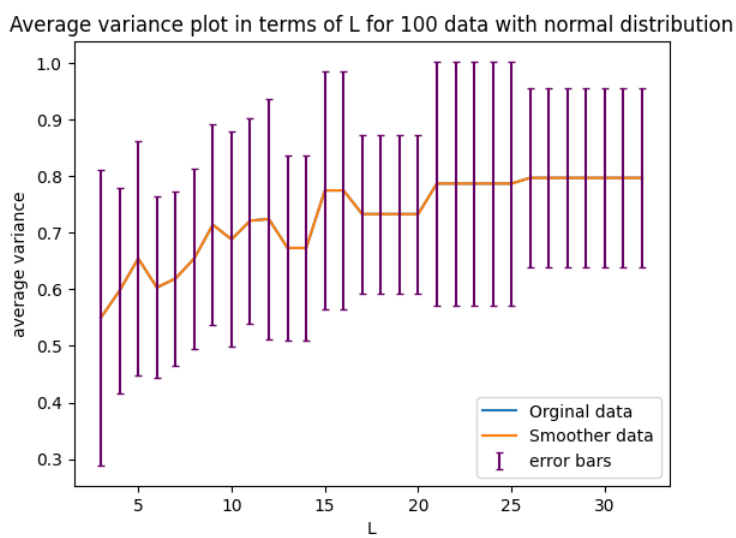

Figure 1: Average variance plot in terms of L for 10^2 data with normal distribution

2.1.1.2 Second method

Augmneted Dickey-fuller Statistic: -9.669236

p-value: 0.000000

critical values at different levels:

    1%: -3.498

    5%: -2.891

    10%: -2.583

2.1.2 Normal distribution n=10^3

2.1.2.1 First method

For 10^3 data, the length of the second-order stationary is 251.

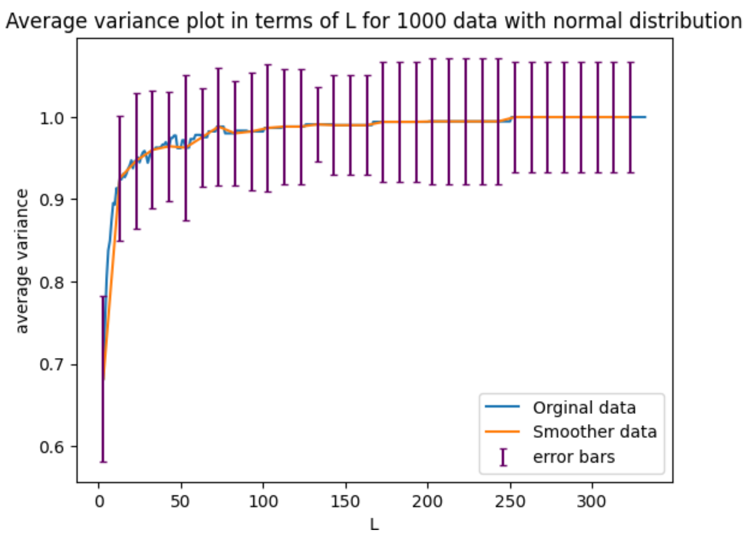

Figure 2: Average variance plot in terms of L for 10^3 data with normal distribution

2.1.2.2 Second method

Augmneted Dickey-fuller Statistic: -13.113123

p-value: 0.000000

critical values at different levels:

    1%: -3.437

    5%: -2.864

    10%: -2.568

2.1.3 Normal distribution n=10^4

2.1.3.1 First method

For 10^4 data, the length of the second-order stationary is 25001.

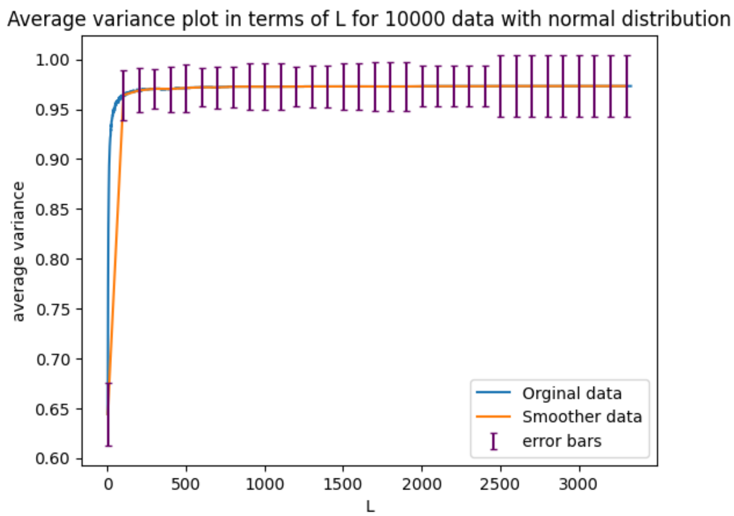

Figure 3: Average variance plot in terms of L for 10^4 data with normal distribution

2.1.3.2 Second method

Augmneted Dickey-fuller Statistic: -99.162574

p-value: 0.000000

critical values at different levels:

  1%: -3.431

  5%: -2.862

  10%: -2.567

2.1.4 Normal distribution n=10^5

2.1.4.1 First method

For 10^5 data, the length of the second-order stationary is 2501.

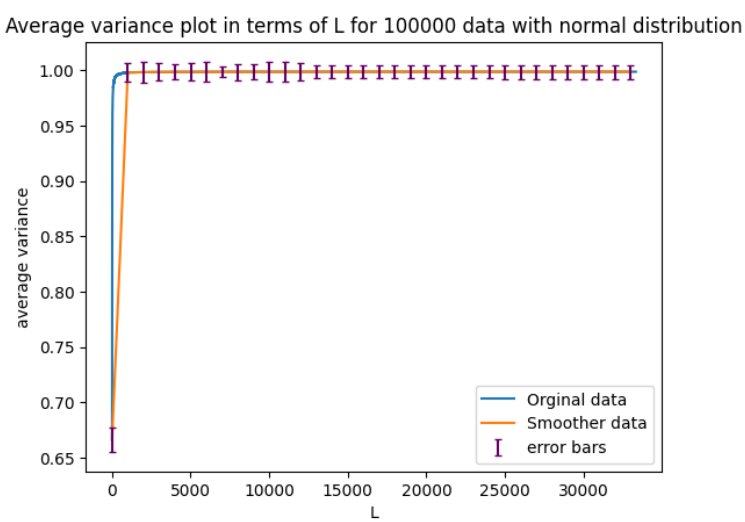

Figure 4: Average variance plot in terms of L for 10^5 data with normal distribution

2.1.4.2 Second method

Augmneted Dickey-fuller Statistic: -100.721542

p-value: 0.000000

critical values at different levels:

1%: -3.430

5%: -2.862

10%: -2.567

2.2 Autoregressive Process

2.2.0.1 First method

For Autoregressive data, we cannot define the length of the second-order stationary well, because in every interval, the data becomes second-order stationary and grows again, and this process repeats.

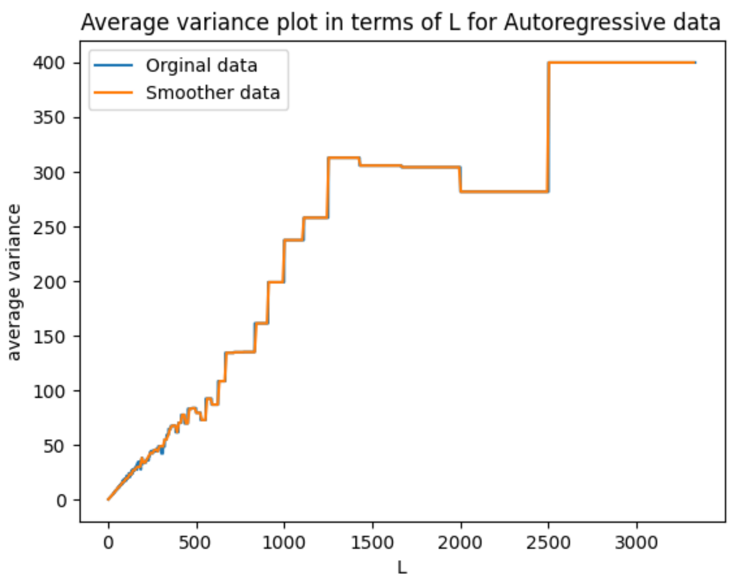

Figure 5: Average variance plot in terms of L for Autoregressive data

2.2.0.2 Second method

Augmneted Dickey-fuller Statistic: -2.249658

p-value: 0.188718

critical values at different levels:

1%: -3.431

5%: -2.862

10%: -2.567

2.3 Exercise 4 data

2.3.0.1 First method

For exercise 4 data, the length of the second-order stationary is 2501.

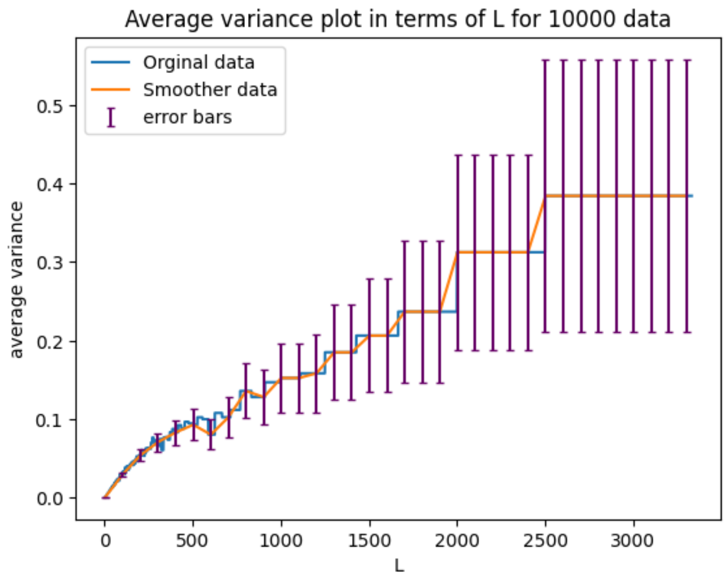

Figure 6: Average variance plot in terms of L for exercise 4 data

2.3.0.2 Second method

Augmneted Dickey-fuller Statistic: -2.133389

p-value: 0.231302

critical values at different levels:

1%: -3.431

5%: -2.862

10%: -2.567

3 Code

```python
# Import needed libraries
import numpy as np
from numpy import inf
import matplotlib.pyplot as plt
from scipy import stats

# Generate n random numbers with normal distribution
datanumbers = int(input('Enter numbers of random data you want to gerenate: '
    ))
random_number = np.random.normal(0, 1, 1000000)
data = np.random.choice(random_number, datanumbers , replace = False)

#read Autoregressive data
data = np.loadtxt('mydataset.txt')

def split_array(array, l):
    # Calculate the number of parts
    num_parts = len(array) // l

    # Split the array into parts of length l
    parts = np.array_split(array, num_parts)

    return parts

# Specify the length of each part
length_of_each_part = 3
errors = []
average_variance = []
l = []
while length_of_each_part <= (int(len(data)/3)-1):
  l.append(length_of_each_part)
  result = split_array(data, length_of_each_part)
  variance = []
  for i in range(len(result)):
    vari = np.var(result[i])
    variance.append(vari)
  av_vari = sum(variance) / len(variance)
  average_variance.append(av_vari)
  variance = np.array(variance)
  error = np.sqrt((np.sum(((variance**(2)) - av_vari)**2)) / (len(variance)*(
    len(variance)-1)))
  errors.append(error)
  length_of_each_part = length_of_each_part + 1

errors = np.array(errors)
from numpy import inf
errors[errors == inf] = 0

for ele in average_variance:
  if ele == average_variance[-1]:
    n = average_variance.index(ele)
    print('Second Order Stationary Lenght:',l[n])
    break

```

```
l = np.array(l)
average_variance = np.array(average_variance)
errors = np.array(errors)

plt.plot(l,average_variance , label='Orginal data')

slicenumber = 1000
lnew = l[::slicenumber]
errorsnew = errors[::slicenumber]
average_variancenew =average_variance[::slicenumber]

plt.plot(lnew,average_variancenew , label='Smoother data')
plt.errorbar(lnew, average_variancenew, yerr=errorsnew, fmt='none', color='
    #660066', capsize=2, label='error bars')
plt.legend()
plt.xlabel("L")
plt.ylabel("average variance")
plt.title(f'Average variance plot in terms of L for {datanumbers} data with
    normal distribution')
plt.show()

import pandas as pd
from statsmodels.tsa.stattools import adfuller


res = adfuller(data)

# Printing the statistical result of the adfuller test
print('Augmneted Dickey_fuller Statistic: %f' % res[0])
print('p-value: %f' % res[1])

# printing the critical values at different alpha levels.
print('critical values at different levels:')
for k, v in res[4].items():
    print('\t%s: %.3f' % (k, v))
```