autocorell

November 9, 2021

[3]:

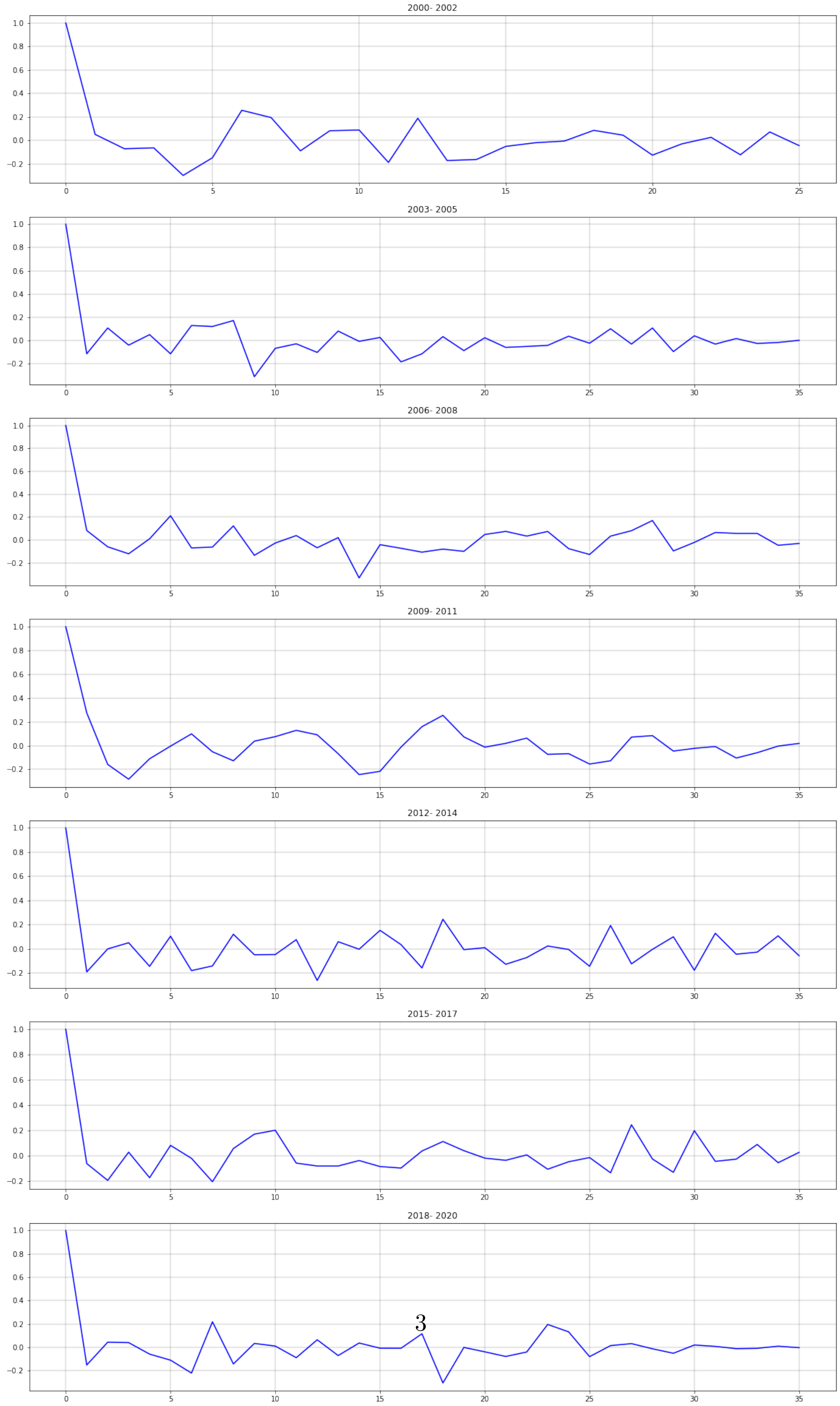
```python
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

import pandas as pd
import csv
import statsmodels.api as sm
import matplotlib.pyplot as plt
```

[4]:

```python
log_returns = []
dates=[]
with open('AMAZONmonthly.csv', mode='r') as csv_file:
    csv_file = csv.DictReader(csv_file)
    line_count = 0
    for row in csv_file:
        if line_count == 0:
#             print(f'Column names are {", ".join(row)}')
            line_count += 1

        dates.append(row["ï»¿Date"])
        log_returns.append(row["Monthly log return"])
        line_count += 1
#     print(f'Processed {line_count} lines.')


#     print(dates[0])
#     print(dates[0].split("/")[2])

    step = 3
    low_bound_year = 2000
    up_bound_year = 2000+step
    autocorrelations = []

    for i in range(7):
        numbers= []
        for i in range(len(dates)):
            year = int(dates[i].split("/")[2])

            if(low_bound_year<= year and year <up_bound_year):
```

```
            numbers.append(log_returns[i])

    correl= sm.tsa.acf(numbers,nlags=35)
    autocorrelations.append(correl)
    low_bound_year = low_bound_year+step
    up_bound_year = up_bound_year+step


plt.rcParams["figure.figsize"]=20,35
fig, axs = plt.subplots(len(autocorrelations))
fig.suptitle('3-years monthly autocorrelations AMAZON')

for i in range(len(autocorrelations)):
    axs[i].grid(color='black', linestyle='-', linewidth=0.2)
    axs[i].plot(autocorrelations[i],"b")
    start=str(i*3+2000)
    end = str((i+1)*3 -1+2000)
    axs[i].title.set_text(start+"- "+end)
```

3-years monthly autocorrelations AMAZON



2000- 2002

2003- 2005

2006- 2008

2009- 2011

2012- 2014

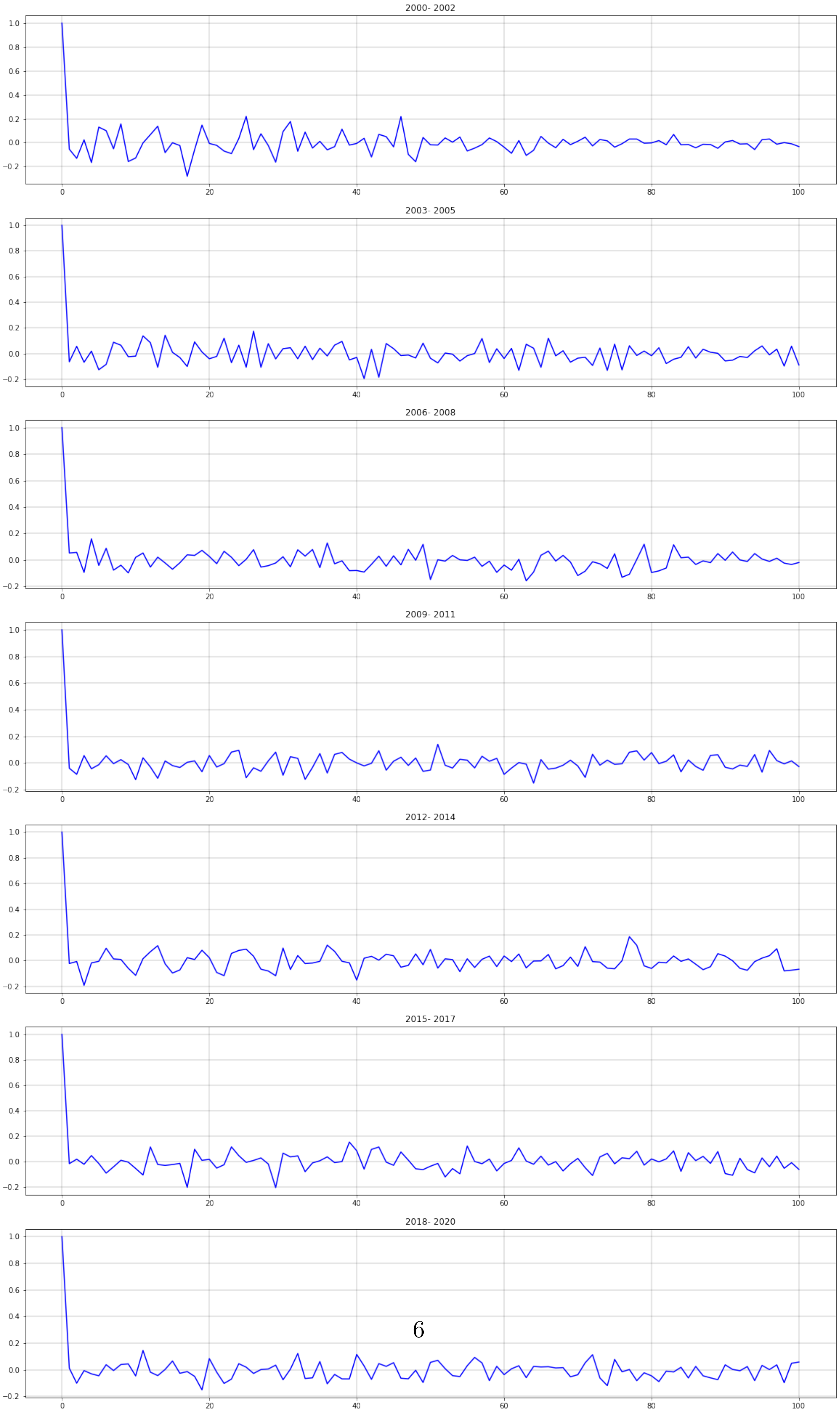
2015- 2017

2018- 2020

[60]:

```
log_returns = []
dates=[]
with open('AMAZON weekly.csv', mode='r') as csv_file:
    csv_file = csv.DictReader(csv_file)
    line_count = 0
    for row in csv_file:
        if line_count == 0:
#             print(f'Column names are {", ".join(row)}')
            line_count += 1

        dates.append(row["ï»¿Date"])
        log_returns.append(row["Weekly log return"])
        line_count += 1
#     print(f'Processed {line_count} lines.')


#     print(dates[0])
#     print(dates[0].split("/")[2])

    step = 3
    low_bound_year = 2000
    up_bound_year = 2000+step
    autocorrelations = []

    for i in range(7):
        numbers= []
        for i in range(len(dates)):
            year = int(dates[i].split("/")[2])

            if(low_bound_year<= year and year <up_bound_year):

                numbers.append(log_returns[i])

        correl= sm.tsa.acf(numbers,nlags=100)
        autocorrelations.append(correl)
        low_bound_year = low_bound_year+step
        up_bound_year = up_bound_year+step


    plt.rcParams["figure.figsize"]=20,35
    fig, axs = plt.subplots(len(autocorrelations))
    fig.suptitle('3-years weekly autocorrelations AMAZON')

    for i in range(len(autocorrelations)):
        axs[i].grid(color='black', linestyle='-', linewidth=0.2)
        axs[i].plot(autocorrelations[i],"b")
        start=str(i*3+2000)
        end = str((i+1)*3 -1+2000)
        axs[i].title.set_text(start+"- "+end)
```

3-years weekly autocorrelations AMAZON



2000- 2002

2003- 2005

2006- 2008

2009- 2011

2012- 2014

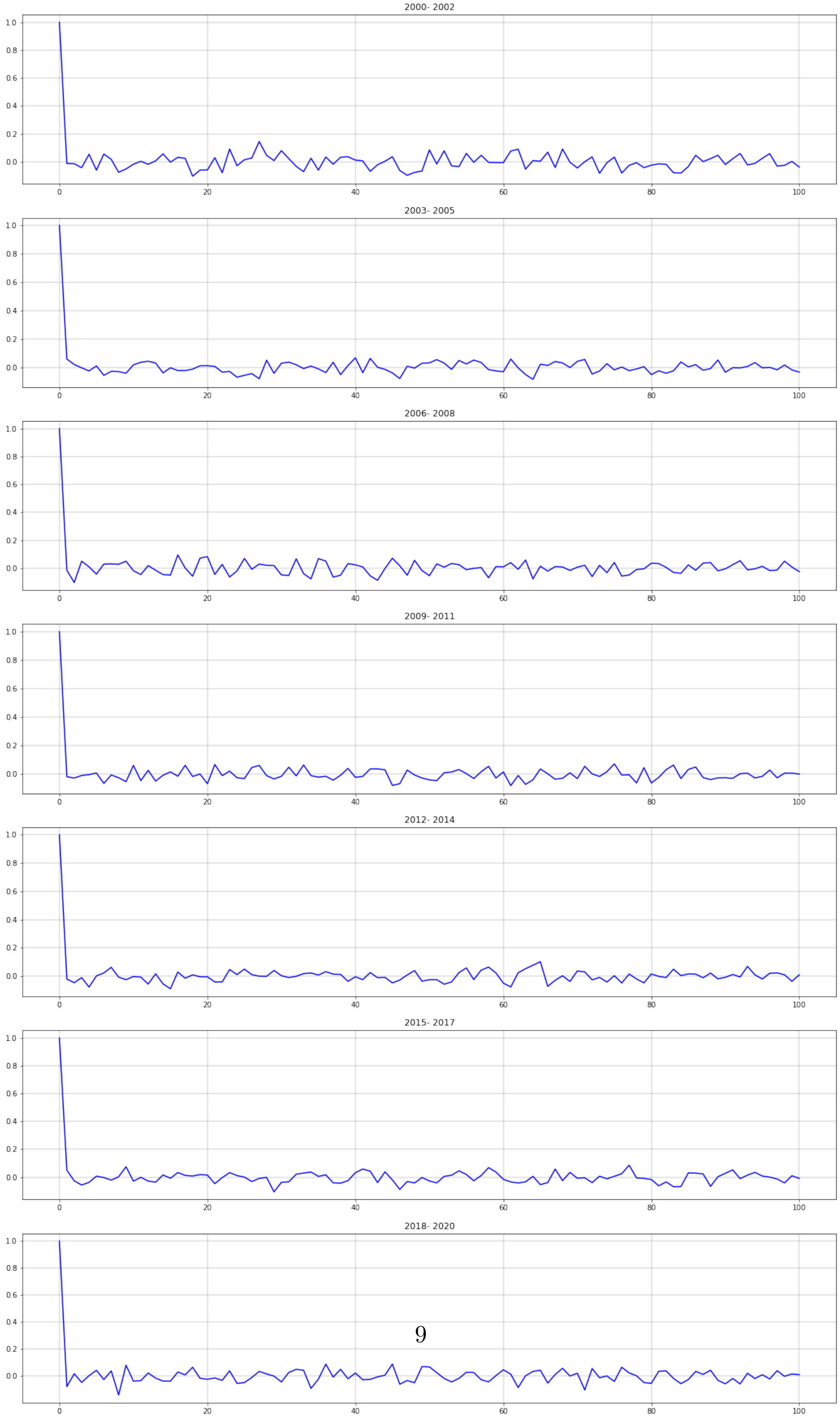
2015- 2017

2018- 2020

[68]:

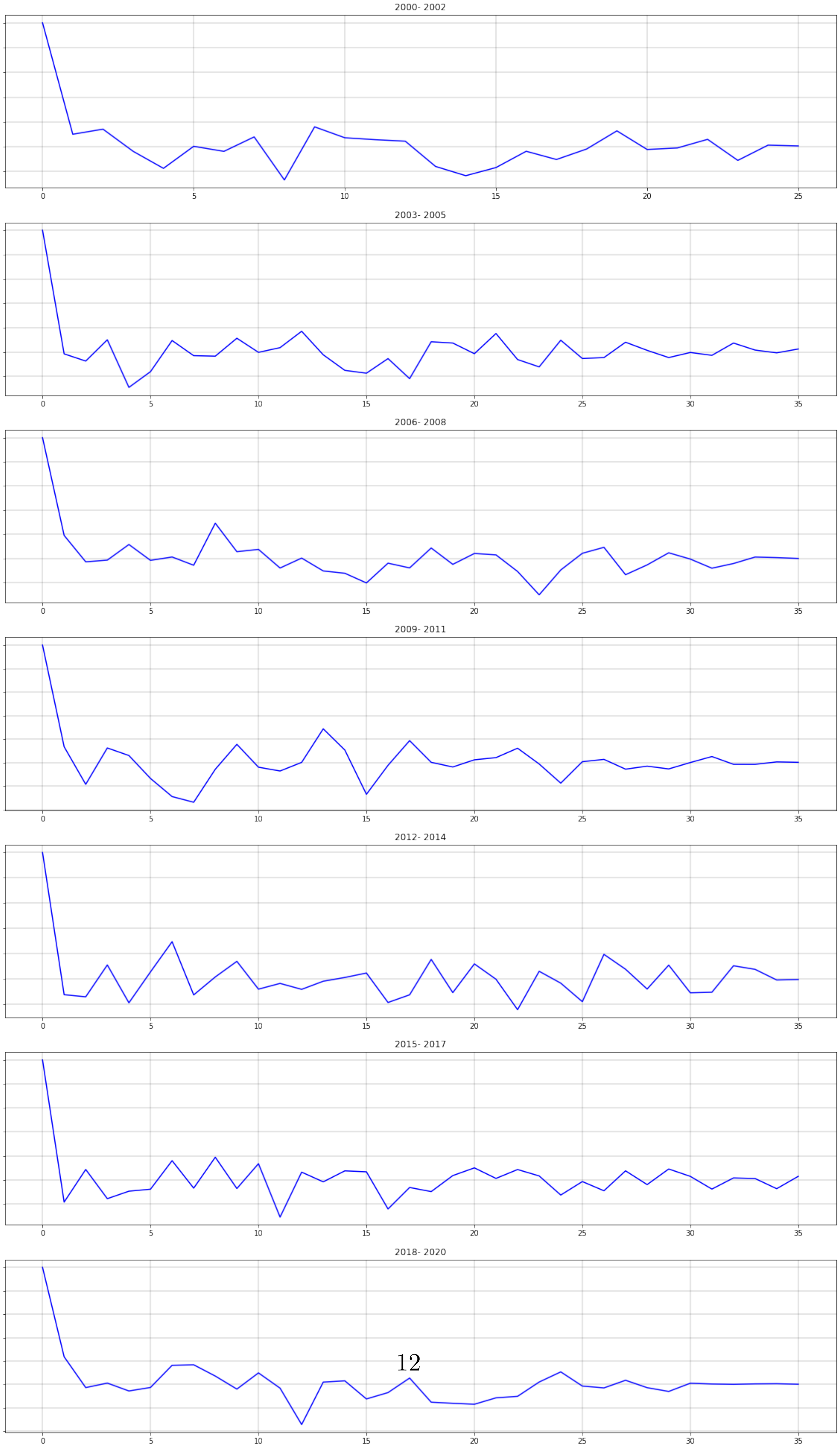
```
log_returns = []
dates=[]
with open('AMAZON daily.csv', mode='r') as csv_file:
    csv_file = csv.DictReader(csv_file)
    line_count = 0
    for row in csv_file:
        if line_count == 0:
#             print(f'Column names are {", ".join(row)}')
            line_count += 1

        dates.append(row["ï»¿Date"])
        log_returns.append(row["Daily log return"])
        line_count += 1
#     print(f'Processed {line_count} lines.')


#     print(dates[0])
#     print(dates[0].split("/")[2])

    step = 3
    low_bound_year = 2000
    up_bound_year = 2000+step
    autocorrelations = []

    for i in range(7):
        numbers= []
        for i in range(len(dates)):
            year = int(dates[i].split("/")[2])

            if(low_bound_year<= year and year <up_bound_year):

                numbers.append(log_returns[i])

        correl= sm.tsa.acf(numbers,nlags=100)
        autocorrelations.append(correl)
        low_bound_year = low_bound_year+step
        up_bound_year = up_bound_year+step


    plt.rcParams["figure.figsize"]=20,35
    fig, axs = plt.subplots(len(autocorrelations))
    fig.suptitle('3-years Daily autocorrelations AMAZON')

    for i in range(len(autocorrelations)):
        axs[i].grid(color='black', linestyle='-', linewidth=0.2)
        axs[i].plot(autocorrelations[i],"b")
        start=str(i*3+2000)
        end = str((i+1)*3 -1+2000)
        axs[i].title.set_text(start+"- "+end)
```

3-years Daily autocorrelations AMAZON



```
[75]: log_returns = []
dates=[]
with open('NVDAmonthly.csv', mode='r') as csv_file:
    csv_file = csv.DictReader(csv_file)
    line_count = 0
    for row in csv_file:
        if line_count == 0:
#             print(f'Column names are {", ".join(row)}')
            line_count += 1

        dates.append(row["ï»¿Date"])
        log_returns.append(row["Monthly log return"])
        line_count += 1
#     print(f'Processed {line_count} lines.')


#     print(dates[0])
#     print(dates[0].split("/")[2])

    step = 3
    low_bound_year = 2000
    up_bound_year = 2000+step
    autocorrelations = []

    for i in range(7):
        numbers= []
        for i in range(len(dates)):
            year = int(dates[i].split("/")[2])

            if(low_bound_year<= year and year <up_bound_year):

                numbers.append(log_returns[i])

        correl= sm.tsa.acf(numbers,nlags=35)
        autocorrelations.append(correl)
        low_bound_year = low_bound_year+step
        up_bound_year = up_bound_year+step


    plt.rcParams["figure.figsize"]=20,35
    fig, axs = plt.subplots(len(autocorrelations))
    fig.suptitle('3-years Monthly autocorrelations NVDA')

    for i in range(len(autocorrelations)):
        axs[i].grid(color='black', linestyle='-', linewidth=0.2)
        axs[i].plot(autocorrelations[i],"b")
        start=str(i*3+2000)
        end = str((i+1)*3 -1+2000)
        axs[i].title.set_text(start+"- "+end)
```

3-years Monthly autocorrelations NVDA



2000- 2002

2003- 2005

2006- 2008

2009- 2011

2012- 2014

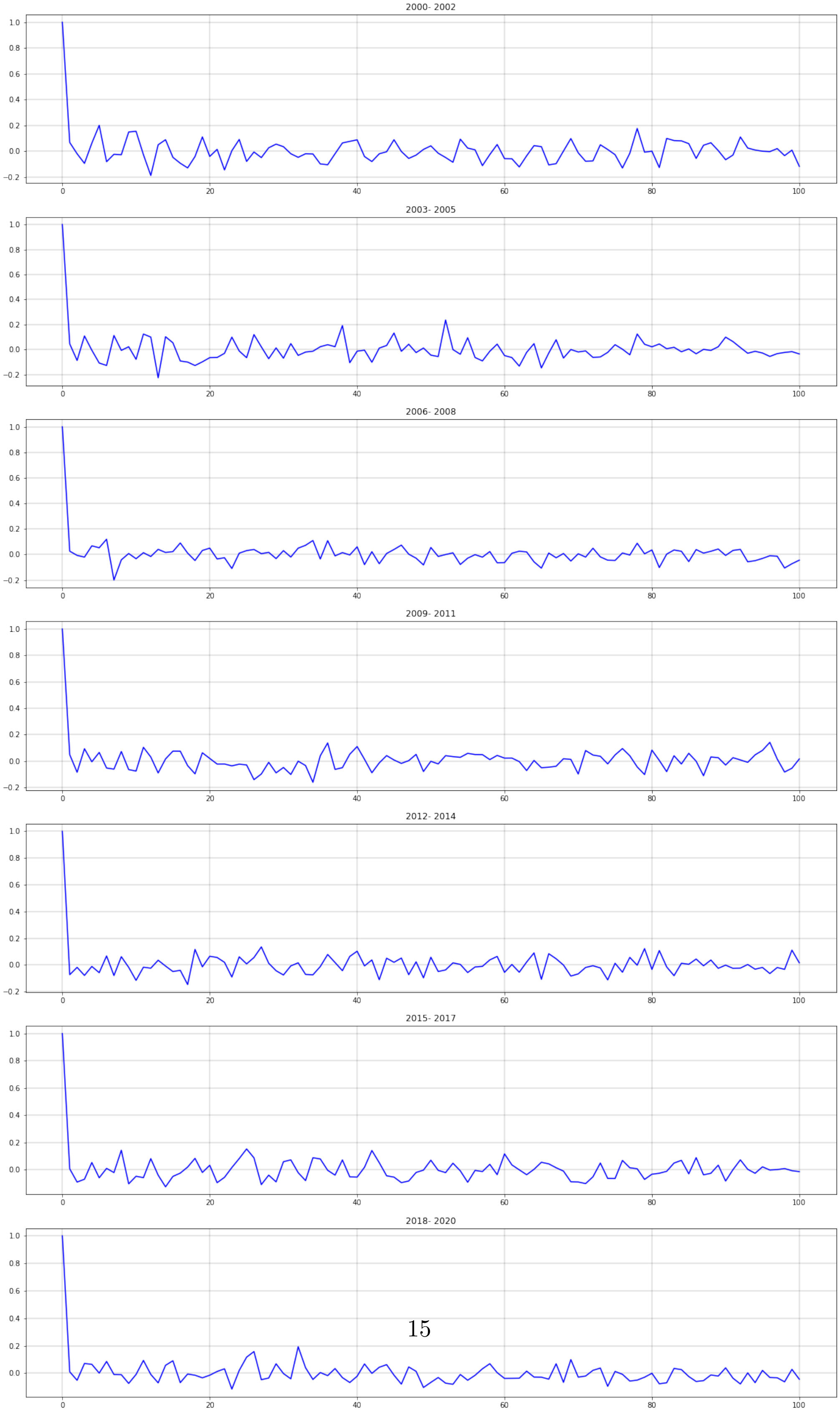
2015- 2017

2018- 2020

[77]:

```python
log_returns = []
dates=[]
with open('NVDAweekly.csv', mode='r') as csv_file:
    csv_file = csv.DictReader(csv_file)
    line_count = 0
    for row in csv_file:
        if line_count == 0:
#             print(f'Column names are {", ".join(row)}')
            line_count += 1

        dates.append(row["ï»¿Date"])
        log_returns.append(row["Weekly log return"])
        line_count += 1
#     print(f'Processed {line_count} lines.')


#     print(dates[0])
#     print(dates[0].split("/")[2])

    step = 3
    low_bound_year = 2000
    up_bound_year = 2000+step
    autocorrelations = []

    for i in range(7):
        numbers= []
        for i in range(len(dates)):
            year = int(dates[i].split("/")[2])

            if(low_bound_year<= year and year <up_bound_year):

                numbers.append(log_returns[i])

        correl= sm.tsa.acf(numbers,nlags=100)
        autocorrelations.append(correl)
        low_bound_year = low_bound_year+step
        up_bound_year = up_bound_year+step


    plt.rcParams["figure.figsize"]=20,35
    fig, axs = plt.subplots(len(autocorrelations))
    fig.suptitle('3-years Weekly autocorrelations NVDA')

    for i in range(len(autocorrelations)):
        axs[i].grid(color='black', linestyle='-', linewidth=0.2)
        axs[i].plot(autocorrelations[i],"b")
        start=str(i*3+2000)
        end = str((i+1)*3 -1+2000)
        axs[i].title.set_text(start+"- "+end)
```

3-years Weekly autocorrelations NVDA


2000- 2002

2003- 2005

2006- 2008

2009- 2011

2012- 2014

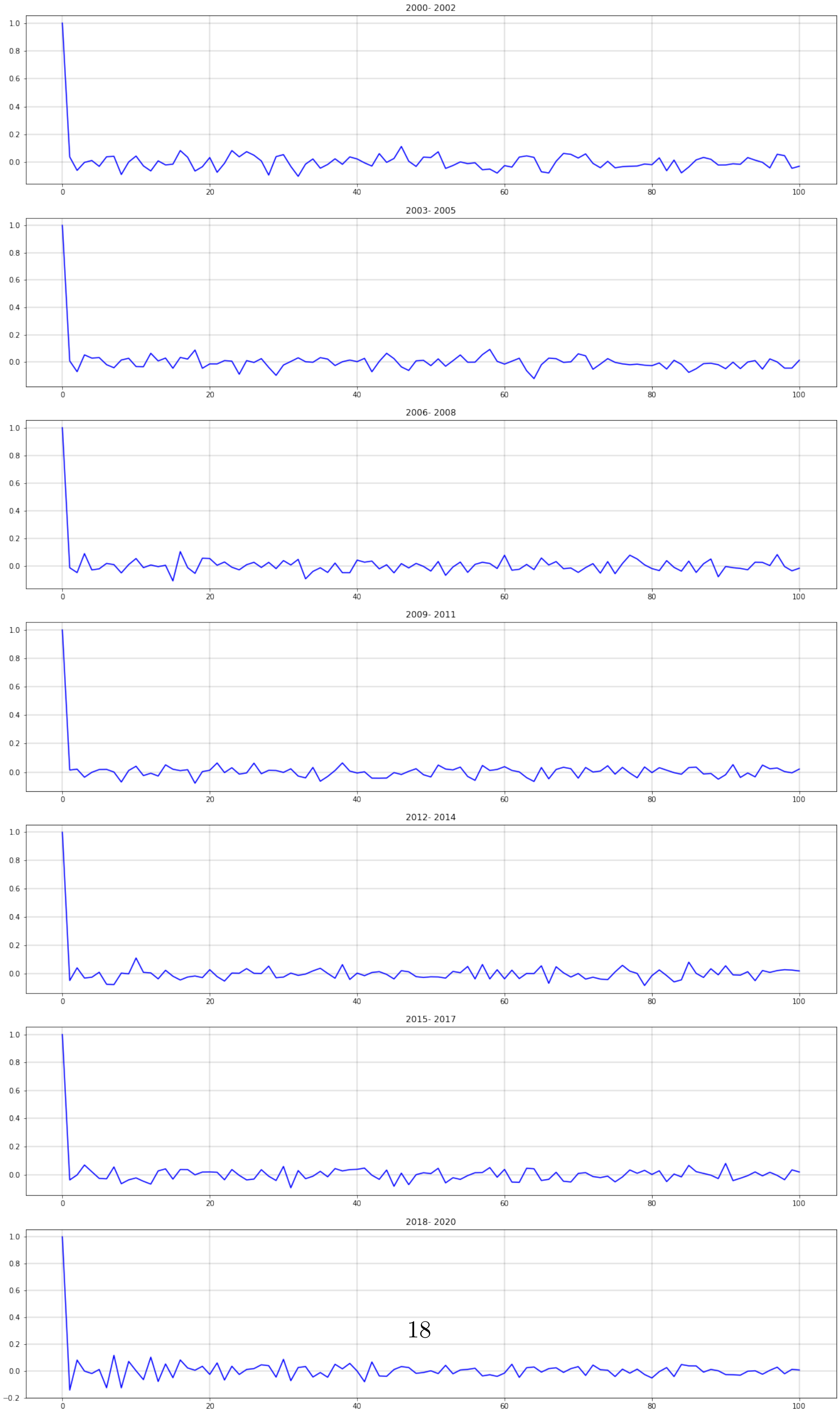
2015- 2017

2018- 2020

[78]:

```
log_returns = []
dates=[]
with open('NVDAdaily.csv', mode='r') as csv_file:
    csv_file = csv.DictReader(csv_file)
    line_count = 0
    for row in csv_file:
        if line_count == 0:
#             print(f'Column names are {", ".join(row)}')
            line_count += 1

        dates.append(row["ï»¿Date"])
        log_returns.append(row["Daily log return"])
        line_count += 1
#     print(f'Processed {line_count} lines.')


#     print(dates[0])
#     print(dates[0].split("/")[2])

    step = 3
    low_bound_year = 2000
    up_bound_year = 2000+step
    autocorrelations = []

    for i in range(7):
        numbers= []
        for i in range(len(dates)):
            year = int(dates[i].split("/")[2])

            if(low_bound_year<= year and year <up_bound_year):

                numbers.append(log_returns[i])

        correl= sm.tsa.acf(numbers,nlags=100)
        autocorrelations.append(correl)
        low_bound_year = low_bound_year+step
        up_bound_year = up_bound_year+step


    plt.rcParams["figure.figsize"]=20,35
    fig, axs = plt.subplots(len(autocorrelations))
    fig.suptitle('3-years Daily autocorrelations NVDA')

    for i in range(len(autocorrelations)):
        axs[i].grid(color='black', linestyle='-', linewidth=0.2)
        axs[i].plot(autocorrelations[i],"b")
        start=str(i*3+2000)
        end = str((i+1)*3 -1+2000)
        axs[i].title.set_text(start+"- "+end)
```

3-years Daily autocorrelations NVDA



2000- 2002

2003- 2005

2006- 2008

2009- 2011

2012- 2014

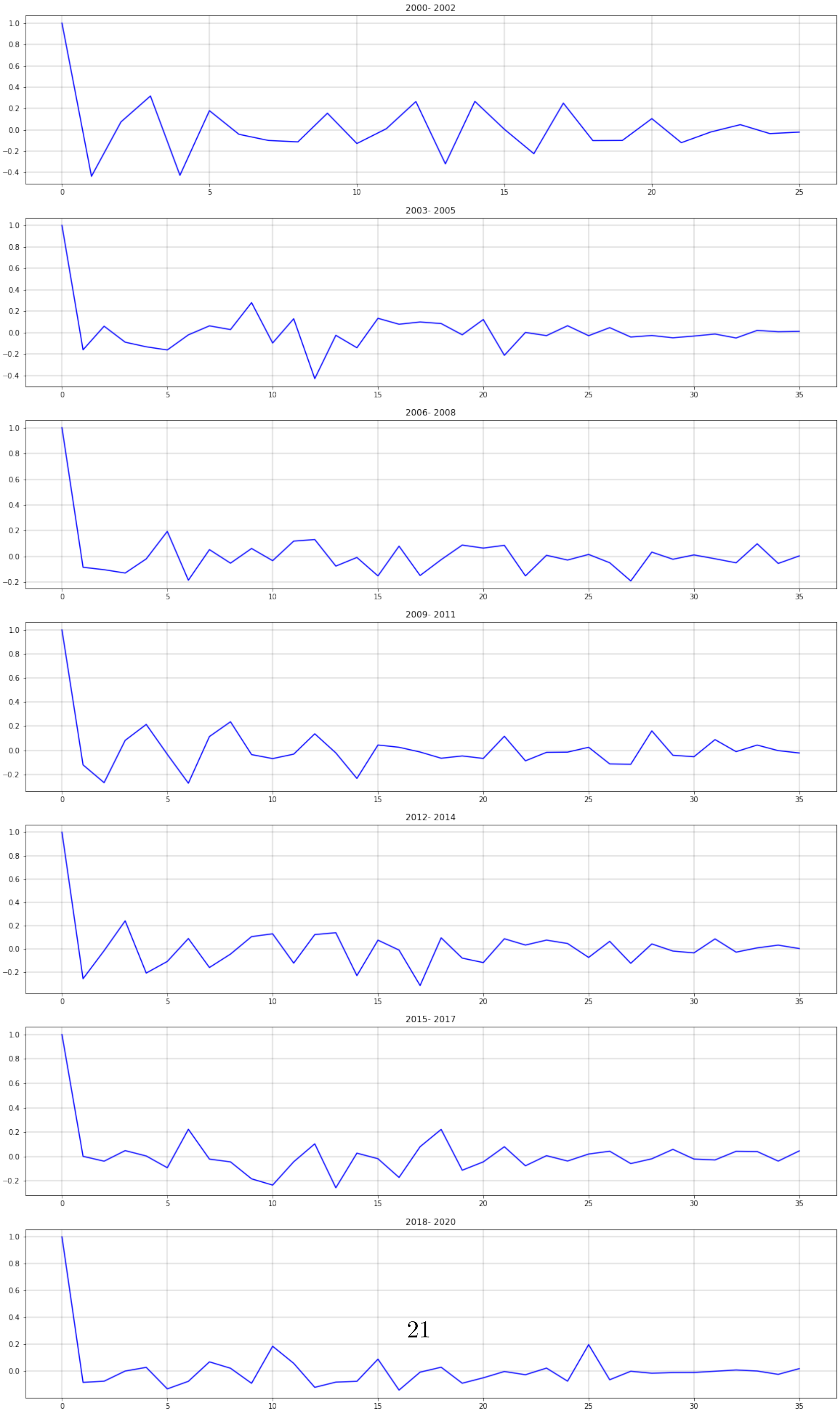
2015- 2017

2018- 2020

[79]:

```
log_returns = []
dates=[]
with open('Realty Income Corporation monthly.csv', mode='r') as csv_file:
    csv_file = csv.DictReader(csv_file)
    line_count = 0
    for row in csv_file:
        if line_count == 0:
#             print(f'Column names are {", ".join(row)}')
            line_count += 1

        dates.append(row["ï»¿Date"])
        log_returns.append(row["Monthly log return"])
        line_count += 1
#     print(f'Processed {line_count} lines.')


#     print(dates[0])
#     print(dates[0].split("/")[2])

    step = 3
    low_bound_year = 2000
    up_bound_year = 2000+step
    autocorrelations = []

    for i in range(7):
        numbers= []
        for i in range(len(dates)):
            year = int(dates[i].split("/")[2])

            if(low_bound_year<= year and year <up_bound_year):

                numbers.append(log_returns[i])

        correl= sm.tsa.acf(numbers,nlags=100)
        autocorrelations.append(correl)
        low_bound_year = low_bound_year+step
        up_bound_year = up_bound_year+step


    plt.rcParams["figure.figsize"]=20,35
    fig, axs = plt.subplots(len(autocorrelations))
    fig.suptitle('3-years Monthly autocorrelations RIC')

    for i in range(len(autocorrelations)):
        axs[i].grid(color='black', linestyle='-', linewidth=0.2)
        axs[i].plot(autocorrelations[i],"b")
        start=str(i*3+2000)
        end = str((i+1)*3 -1+2000)
        axs[i].title.set_text(start+"- "+end)
```

3-years Monthly autocorrelations RIC



2000- 2002

2003- 2005

2006- 2008

2009- 2011

2012- 2014

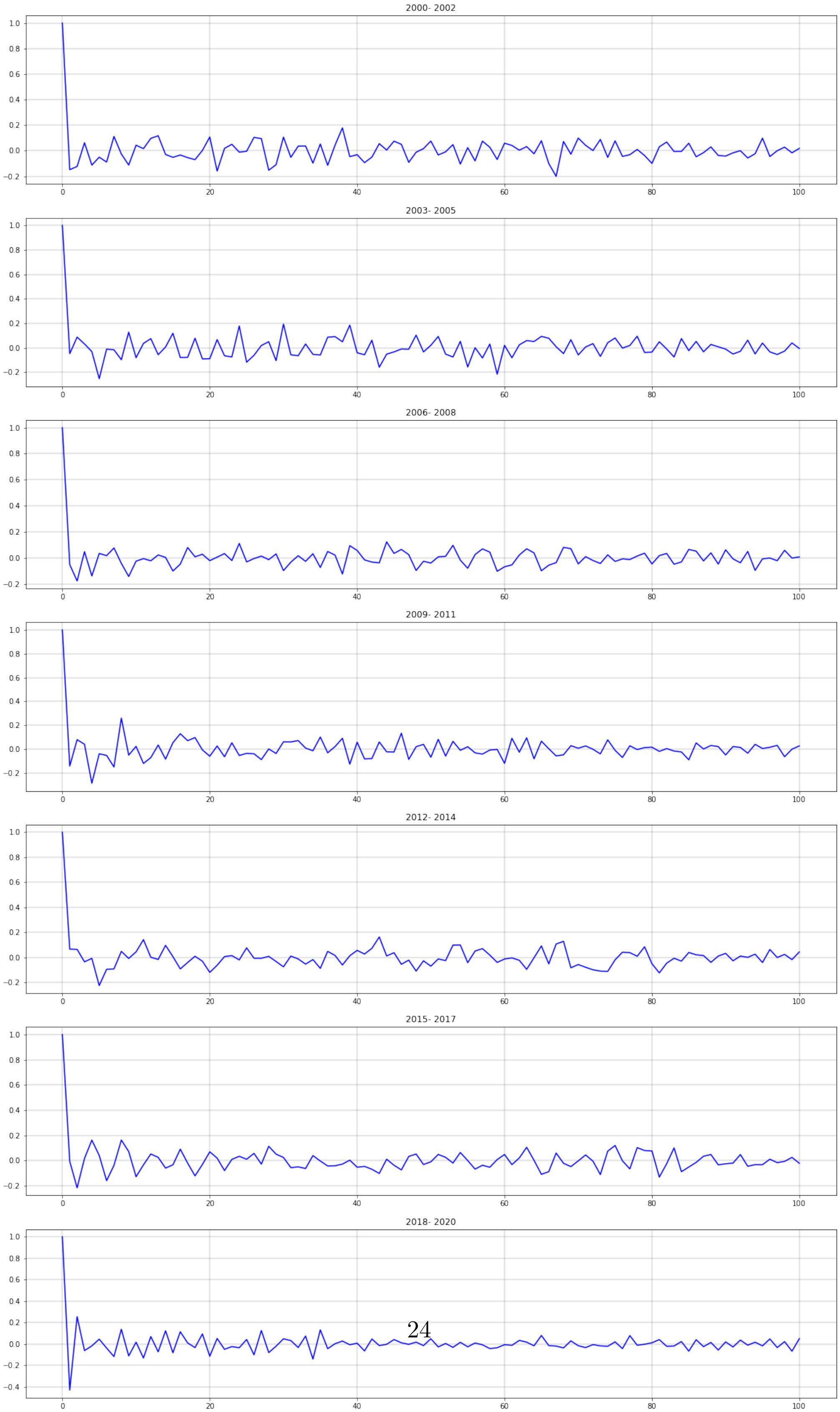
2015- 2017

2018- 2020

[80]:

```
log_returns = []
dates=[]
with open('Realty Income Corporation weekly.csv', mode='r') as csv_file:
    csv_file = csv.DictReader(csv_file)
    line_count = 0
    for row in csv_file:
        if line_count == 0:
#             print(f'Column names are {", ".join(row)}')
            line_count += 1

        dates.append(row["ï»¿Date"])
        log_returns.append(row["Weekly log return"])
        line_count += 1
#     print(f'Processed {line_count} lines.')


#     print(dates[0])
#     print(dates[0].split("/")[2])

    step = 3
    low_bound_year = 2000
    up_bound_year = 2000+step
    autocorrelations = []

    for i in range(7):
        numbers= []
        for i in range(len(dates)):
            year = int(dates[i].split("/")[2])

            if(low_bound_year<= year and year <up_bound_year):

                numbers.append(log_returns[i])

        correl= sm.tsa.acf(numbers,nlags=100)
        autocorrelations.append(correl)
        low_bound_year = low_bound_year+step
        up_bound_year = up_bound_year+step


    plt.rcParams["figure.figsize"]=20,35
    fig, axs = plt.subplots(len(autocorrelations))
    fig.suptitle('3-years Weekly autocorrelations RIC')

    for i in range(len(autocorrelations)):
        axs[i].grid(color='black', linestyle='-', linewidth=0.2)
        axs[i].plot(autocorrelations[i],"b")
        start=str(i*3+2000)
        end = str((i+1)*3 -1+2000)
        axs[i].title.set_text(start+"- "+end)
```

3-years Weekly autocorrelations RIC



2000- 2002

2003- 2005

2006- 2008

2009- 2011

2012- 2014

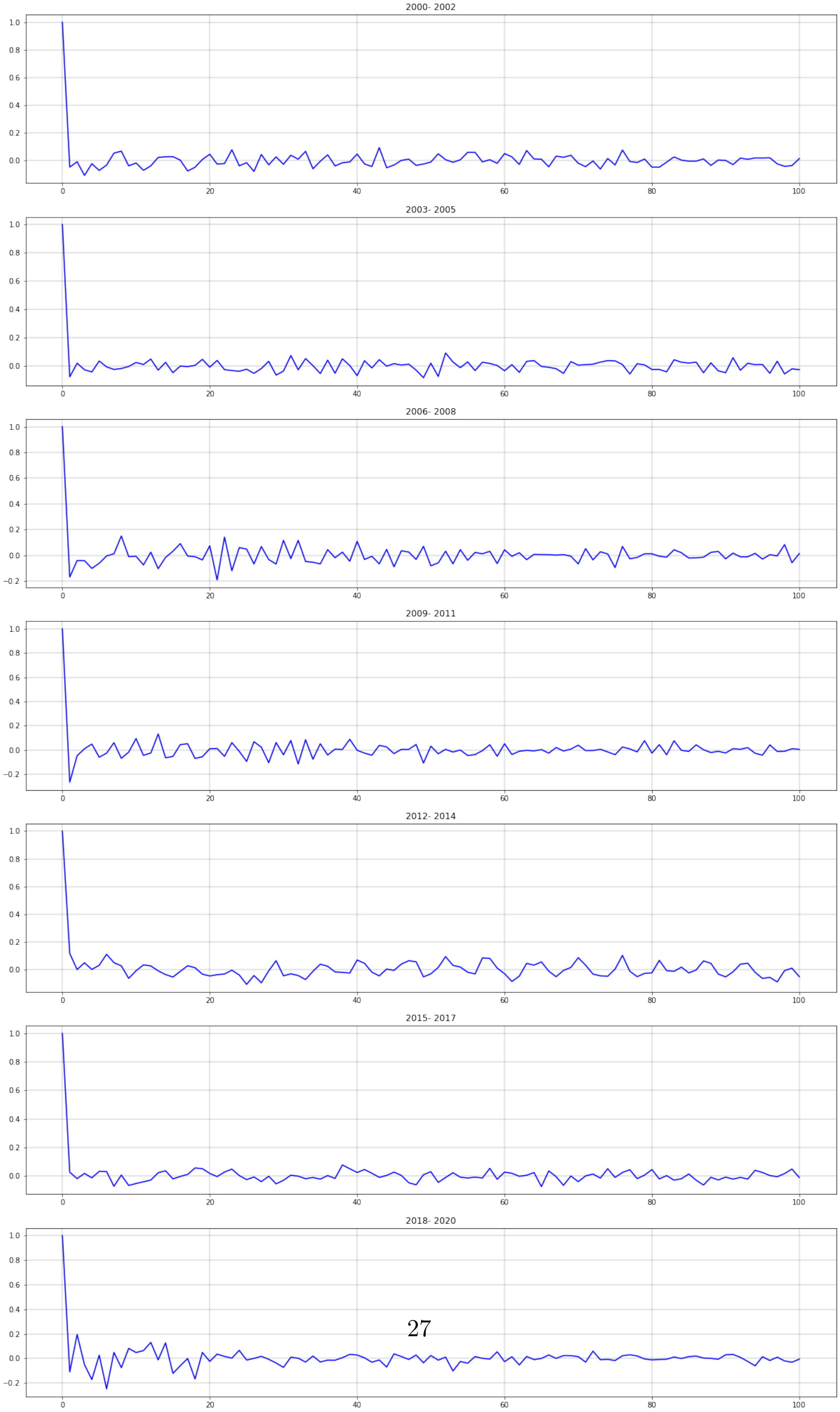
2015- 2017

2018- 2020

[81]:

```
log_returns = []
dates=[]
with open('Realty Income Corporation daily.csv', mode='r') as csv_file:
    csv_file = csv.DictReader(csv_file)
    line_count = 0
    for row in csv_file:
        if line_count == 0:
#             print(f'Column names are {", ".join(row)}')
            line_count += 1

        dates.append(row["ï»¿Date"])
        log_returns.append(row["Daily log return"])
        line_count += 1
#     print(f'Processed {line_count} lines.')


#     print(dates[0])
#     print(dates[0].split("/")[2])

    step = 3
    low_bound_year = 2000
    up_bound_year = 2000+step
    autocorrelations = []

    for i in range(7):
        numbers= []
        for i in range(len(dates)):
            year = int(dates[i].split("/")[2])

            if(low_bound_year<= year and year <up_bound_year):

                numbers.append(log_returns[i])

        correl= sm.tsa.acf(numbers,nlags=100)
        autocorrelations.append(correl)
        low_bound_year = low_bound_year+step
        up_bound_year = up_bound_year+step


    plt.rcParams["figure.figsize"]=20,35
    fig, axs = plt.subplots(len(autocorrelations))
    fig.suptitle('3-years Daily autocorrelations RIC')

    for i in range(len(autocorrelations)):
        axs[i].grid(color='black', linestyle='-', linewidth=0.2)
        axs[i].plot(autocorrelations[i],"b")
        start=str(i*3+2000)
        end = str((i+1)*3 -1+2000)
        axs[i].title.set_text(start+"- "+end)
```

3-years Daily autocorrelations RIC



2000- 2002

2003- 2005

2006- 2008

2009- 2011

2012- 2014

2015- 2017

2018- 2020