Forecasting Economic Recession Probabilities Using SARIMA and Hidden Markov Chain (HMM)

Introduction

The provided dataset represents an economic index that quantifies the probability of the U.S. economy being in a recession during a given quarter. The data, measured in percentage points and reported quarterly, is not seasonally adjusted. The index offers a mechanistic approach to recession identification, contrasting the more subjective methods such as those employed by the National Bureau of Economic Research (NBER), which rely on assessments of a range of economic indicators.

This index is designed to provide a timely and objective measure, relying solely on currently available Gross Domestic Product (GDP) data to infer the likelihood of a recession. It utilizes a mathematical model to distinguish the characteristics of recessions from those of economic expansions. With the advantage of real-time reporting, the index allows for immediate insights into the economic climate, eliminating delays often experienced with traditional indicators.

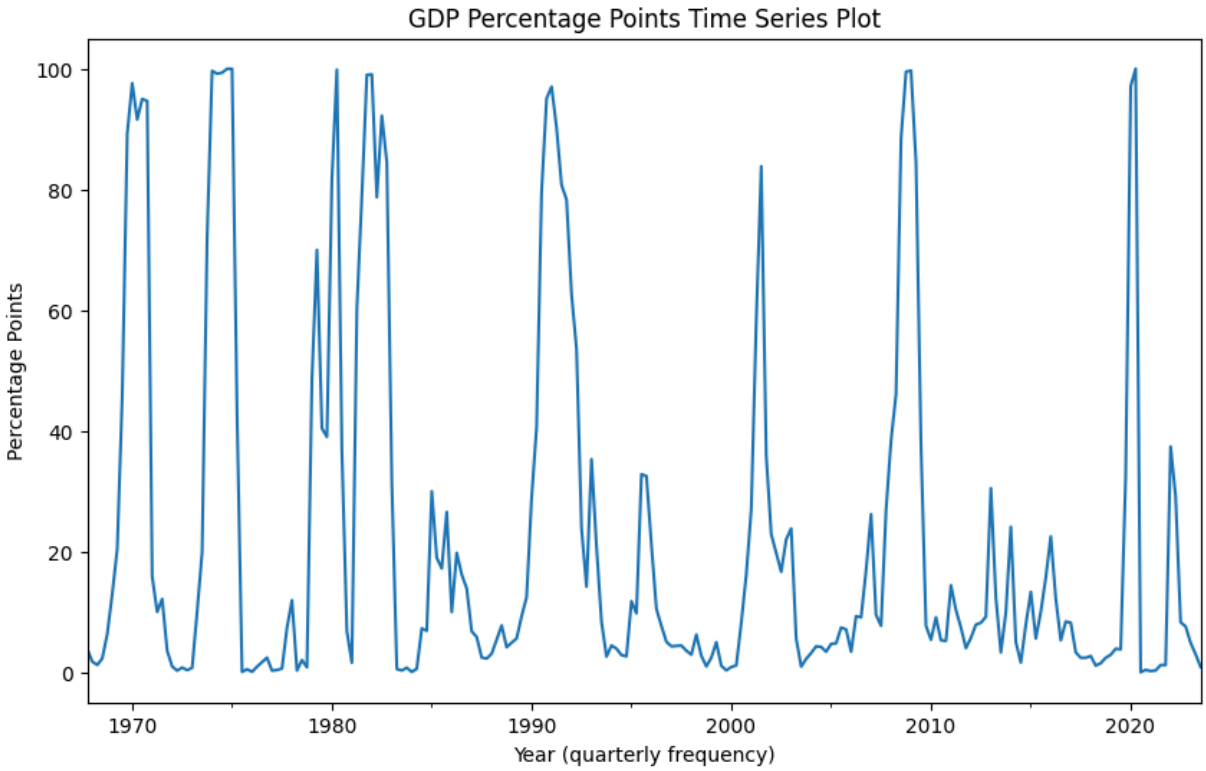
The primary objective of analyzing this dataset is to develop a predictive model to ascertain the onset of recessions. By identifying when the index exceeds the 67% threshold, the model would signal a recession, offering a historically reliable indicator of an economic downturn. Conversely, when the index falls below 33%, it indicates the end of a recession. This real-time assessment aims to provide early warnings of economic contractions, which is crucial for policymakers, investors, and economic analysts in their decision-making processes. The aim is to harness the predictive power of this index to forecast recessions accurately and timely, leveraging only the GDP data available up to one quarter post the date of interest, reflecting the data's original state at the time of reporting.

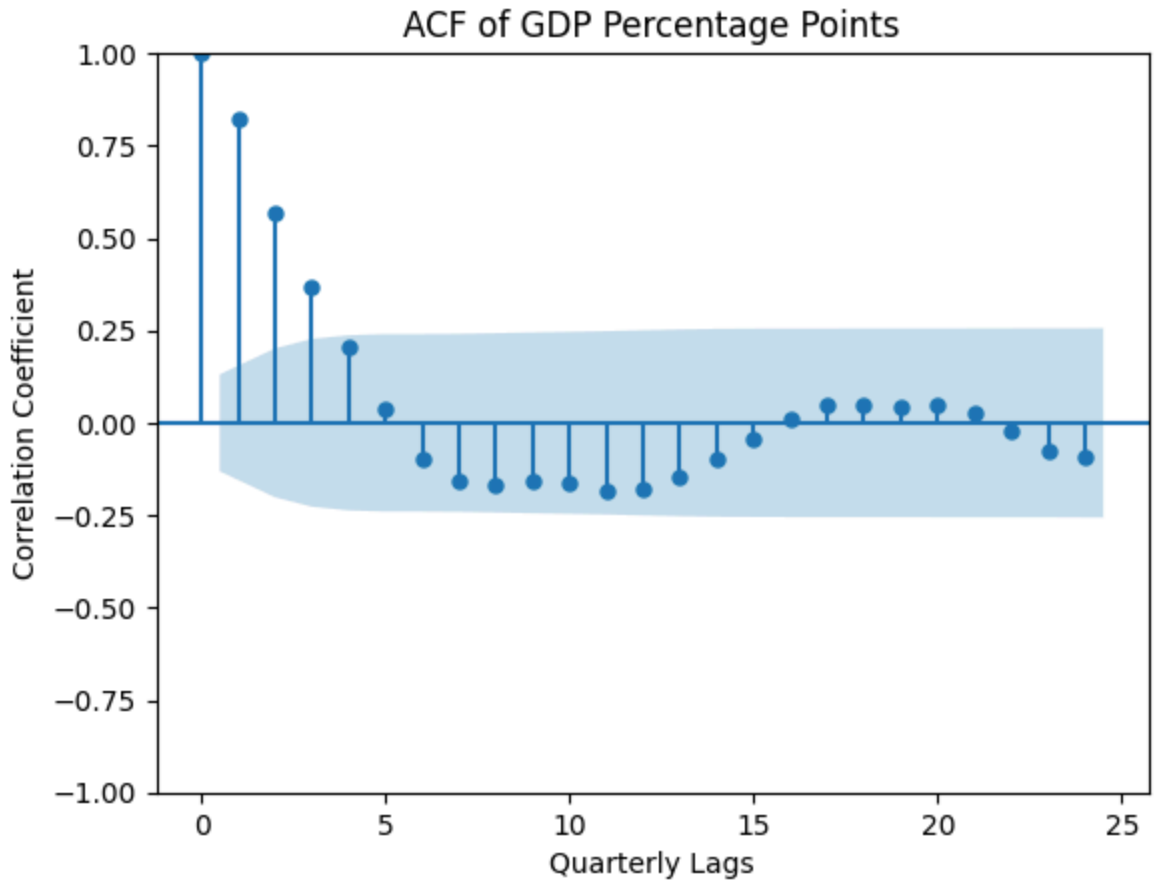
Data Source: Hamilton, James, GDP-Based Recession Indicator Index [JHGDPBRINDX], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/JHGDPBRINDX, March 27, 2024.

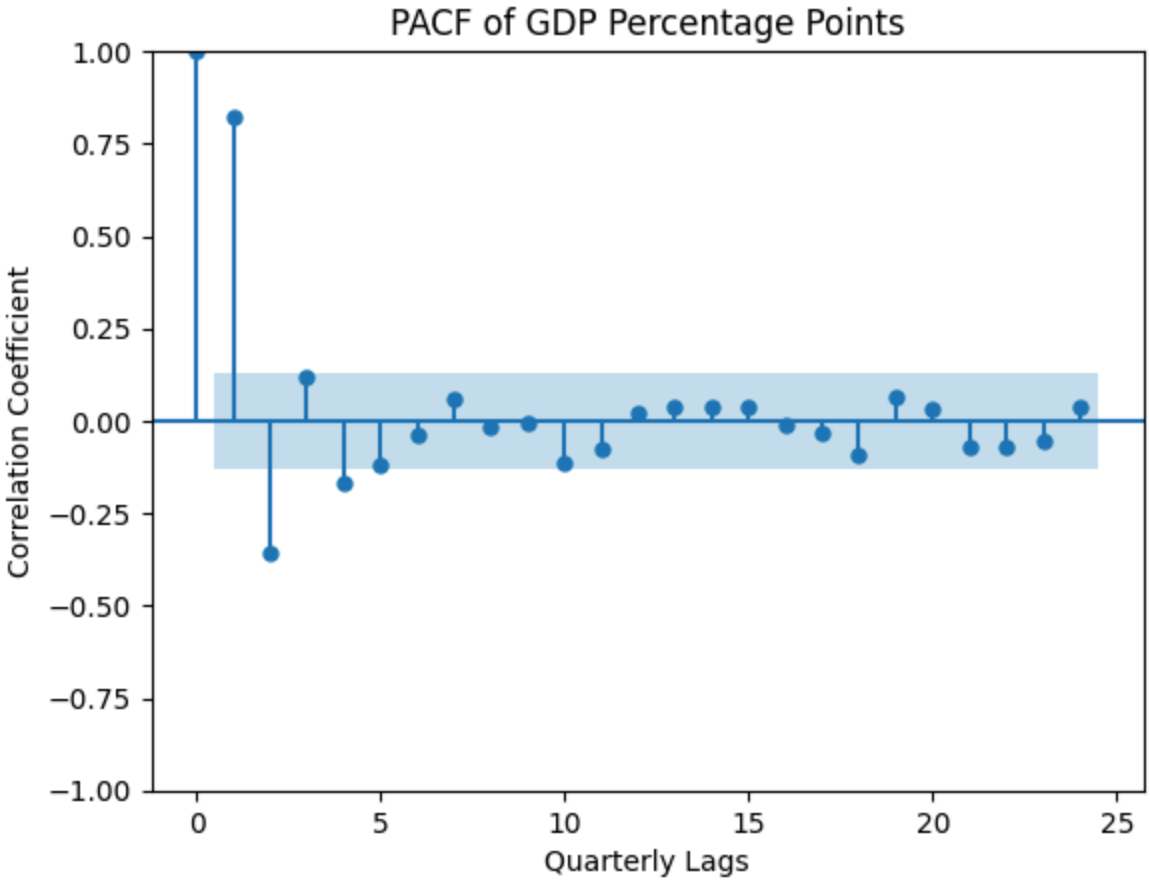
Exploring The Data

In [ ]:
```
import numpy as np
import pandas as pd
import statsmodels.api as sm
```

```
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
import matplotlib.pyplot as plt
import seaborn
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.stattools import adfuller
from pmdarima import auto_arima, ARIMA
import warnings

# Loading the DataFrame
df = pd.read_csv('~/Desktop/e_f_p/Projects/Project-4/gdp_recession_II.csv')

df['DATE'] = pd.to_datetime(df['DATE'])
df.set_index('DATE', inplace=True)
df.rename(columns={'JHGDPBRINDX': 'Percentage_Points'}, inplace=True)



#Plotting the time series
df['Percentage_Points'].plot(figsize=(10, 6), title='GDP Percentage Points T
plt.xlabel('Year (quarterly frequency)')
plt.ylabel('Percentage Points')
plt.show()


#plotting the autocorrelation plot
plot_acf(df['Percentage_Points'])
plt.xlabel('Quarterly Lags')
plt.ylabel('Correlation Coefficient')
plt.title('ACF of GDP Percentage Points')
plt.show()

#Plotting the partical autocorrelation plot
plot_pacf(df['Percentage_Points'])
plt.xlabel('Quarterly Lags')
plt.ylabel('Correlation Coefficient')
plt.title('PACF of GDP Percentage Points')
plt.show()
```


GDP Percentage Points Time Series Plot
Percentage Points
Year (quarterly frequency)
0
20
40
60
80
100
1970
1980
1990
2000
2010
2020


ACF of GDP Percentage Points
Correlation Coefficient
Quarterly Lags
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25



The time series plot presents dramatic fluctuations in the probability of a recession over time. Sharp increases in the index suggest a high probability of recession, with subsequent decreases indicating a recovery. The graph shows peaks surpassing the critical 67% threshold, which historically signals a recession.

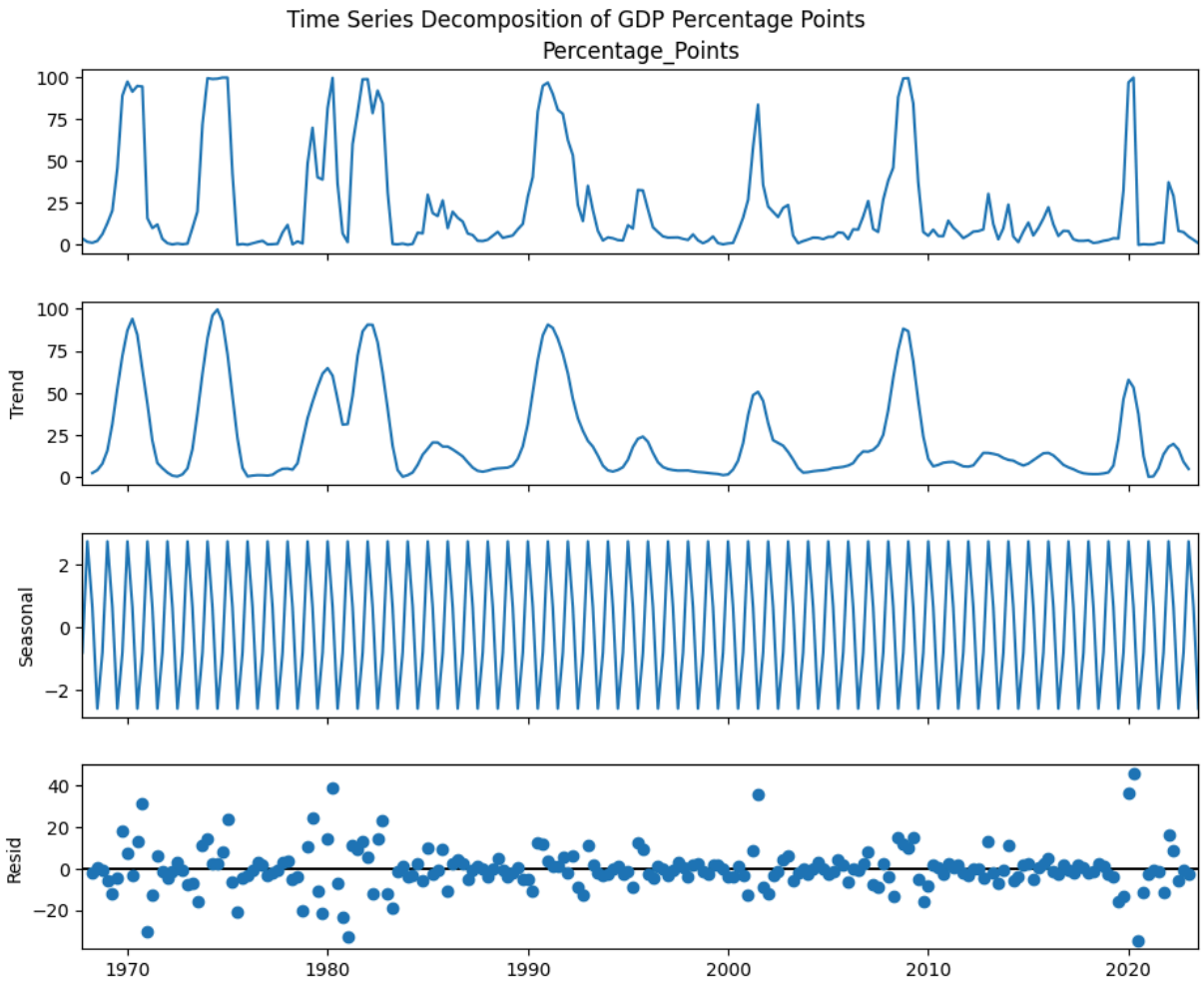
The Autocorrelation Function (ACF) plot indicates minimal autocorrelation beyond the immediate quarter. The first lag is substantial, which is typical, as a time series is always perfectly correlated with itself at lag 0. However, the quick decline of autocorrelation values within the confidence bounds suggests a lack of a strong relationship between past and future values.

The Partial Autocorrelation Function (PACF) plot also supports this, with a significant drop after the first lag, hinting at a potential AR(1) model suitability. However, the presence of a spike at the first lag indicates that the quarterly data might contain a seasonal pattern that should be investigated further.

The analysis suggests that the index lacks strong internal dependencies over time, which might imply the necessity to include external economic indicators or variables to improve the predictive accuracy of potential recessions. The goal is to develop a model that can utilize this index, alongside other economic indicators if necessary, to forecast recessions in a timely manner, providing valuable foresight into the U.S. economy's cyclical dynamics.

Decomposition of Time Series

In [ ]:
```
# Decomposition of Time Series

decomposition = seasonal_decompose(df['Percentage_Points'], model='additive'


# Plot of decomposed components (Time Series, Trend, Seasonal, Residuals)
fig = decomposition.plot()
fig.set_size_inches(10, 8)
fig.suptitle('Time Series Decomposition of GDP Percentage Points')
plt.show()

# General summary statistics
print(df.describe())

# Performing Augmented Dickey-Fuller test to test stationarity
adf_test = adfuller(df['Percentage_Points'])

print('ADF Statistic: %f' % adf_test[0])
print('p-value: %f' % adf_test[1])
print('Critical Values:')
for key, value in adf_test[4].items():
    print('\t%s: %.3f' % (key, value))

# Interpretation
if adf_test[1] > 0.05:
    print("The time series is not stationary.")
else:
    print("The time series is stationary.")

# Auto-fitting an ARMA model

auto_arima_model = auto_arima(df['Percentage_Points'], start_p=1, start_q=1,
                              max_p=5, max_q=5, seasonal=True, d=0, trace=Tr
                              error_action= 'ignore', supress_warnings=True,


print(auto_arima_model.summary())
```


Time Series Decomposition of GDP Percentage Points
Percentage_Points
Trend
Seasonal
Resid
1970
1980
1990
2000
2010
2020

```
       Percentage_Points
count         224.000000
mean           23.198370
std            30.925235
min             0.000000
25%             2.982525
50%             8.195150
75%            29.396100
max           100.000000
ADF Statistic: -6.001071
p-value: 0.000000
Critical Values:
        1%: -3.461
        5%: -2.875
        10%: -2.574
The time series is stationary.
Performing stepwise search to minimize aic
 ARIMA(1,0,1)(0,0,0)[0] intercept   : AIC=1887.485, Time=0.03 sec
 ARIMA(0,0,0)(0,0,0)[0] intercept   : AIC=2176.027, Time=0.01 sec
 ARIMA(1,0,0)(0,0,0)[0] intercept   : AIC=1922.071, Time=0.02 sec
 ARIMA(0,0,1)(0,0,0)[0] intercept   : AIC=1983.080, Time=0.03 sec
 ARIMA(0,0,0)(0,0,0)[0]             : AIC=2274.387, Time=0.01 sec
 ARIMA(2,0,1)(0,0,0)[0] intercept   : AIC=1889.343, Time=0.04 sec
 ARIMA(1,0,2)(0,0,0)[0] intercept   : AIC=1889.351, Time=0.04 sec
 ARIMA(0,0,2)(0,0,0)[0] intercept   : AIC=1920.031, Time=0.04 sec
 ARIMA(2,0,0)(0,0,0)[0] intercept   : AIC=1893.154, Time=0.03 sec
 ARIMA(2,0,2)(0,0,0)[0] intercept   : AIC=1887.666, Time=0.11 sec
 ARIMA(1,0,1)(0,0,0)[0]             : AIC=1897.947, Time=0.02 sec

Best model:  ARIMA(1,0,1)(0,0,0)[0] intercept
Total fit time: 0.384 seconds
                               SARIMAX Results
==============================================================================
==
Dep. Variable:                      y   No. Observations:                 2
24
Model:               SARIMAX(1, 0, 1)   Log Likelihood                -939.7
42
Date:                Sat, 06 Apr 2024   AIC                           1887.4
85
Time:                        13:04:25   BIC                           1901.1
31
Sample:                    10-01-1967   HQIC                          1892.9
93
                         - 07-01-2023
Covariance Type:                  opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
intercept      6.8939      3.641      1.893      0.058      -0.243      14.0
31
ar.L1          0.6966      0.064     10.824      0.000       0.570       0.8
23
```

```
ma.L1          0.4799      0.061      7.926      0.000       0.361       0.5
99
sigma2       256.2045     19.916     12.864      0.000     217.170     295.2
39
==============================================================================
=======
Ljung-Box (L1) (Q):                   0.04   Jarque-Bera (JB):
209.63
Prob(Q):                              0.85   Prob(JB):
0.00
Heteroskedasticity (H):               0.65   Skew:
0.04
Prob(H) (two-sided):                  0.07   Kurtosis:
7.74
==============================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```

The observed time series demonstrates significant volatility, indicative of the changing probability of recession over the given timeframe. Such fluctuations suggest periods of economic stress and recovery, aligning with historical business cycle trends.

Focusing on the trend component, it doesn't portray a unidirectional movement but rather a series of expansions and contractions. This is characteristic of economic cycles and points to the necessity of dynamic strategies for risk management and investment positioning.

The pronounced seasonality detected underscores the cyclical nature of the economy and the importance of considering seasonal adjustments when making economic forecasts or investment decisions. This periodicity could reflect fiscal policy impacts, consumer behavior changes, or other cyclical economic factors.

The residuals, representing the noise after accounting for trend and seasonality, display no apparent autocorrelation. This randomness suggests that the model has effectively isolated the deterministic components from the stochastic ones, which is critical for the accuracy of predictive analytics.

Statistical measures give us an average likelihood of recession at approximately 23.20%, with notable variability (standard deviation around 30.93%). The extreme value of 100% aligns with the worst economic downturns and can be seen as 'stress-test' scenarios in risk management frameworks.

The Augmented Dickey-Fuller test confirms stationarity in the time series, which implies that the data is suitable for standard time-series analysis techniques without the need for further differencing to achieve stationarity.

Model selection via AIC has identified the ARIMA(1,0,1) as the optimal predictive model. The inclusion of one AR term and one MA term indicates that both immediate past values and the errors associated with those past values are significant predictors for the series. Such a model could be instrumental in developing short-term economic forecasts and informing tactical asset allocation.

Splitting Data Into Train & Test

In [ ]:
```
# Train and Test Data

# Number of observations
n = 224
num_rows, num_columns = df.shape
print("Number of total observations:", num_rows)

#splitting 80/20
split_index = int(n * 0.8)


# Splitting the DataFrame
train = df.iloc[:split_index]
test = df.iloc[split_index:]

print(f"Training Set: {len(train)} observations")
print(f"Test Set: {len(test)} observations")
```

```
Number of total observations: 224
Training Set: 179 observations
Test Set: 45 observations
```

Splitting the data into training and testing sets is an essential practice in forecasting because it helps validate the predictive power of the model. The training set is used to fit the model, allowing it to 'learn' from historical data. In contrast, the test set acts as a proxy for future, unseen data, enabling us to evaluate how well the model generalizes to new information.

By using this approach, we ensure that the evaluation of our forecasting model's performance is unbiased and indicative of its real-world applicability. This separation mitigates the risk of overfitting, where the model performs exceptionally well on the training data but fails to predict future trends accurately. For forecasting economic recessions, which have significant implications for policy and investment decisions, such rigor in model validation is particularly critical.

In our case, with 224 observations, an 80/20 split provides a substantial amount of data for training (approx. 179 observations) while retaining a meaningful subset (approx. 45 observations) for testing the model's effectiveness. This balance is crucial to develop robust forecasting capabilities while ensuring we have enough recent data to verify our model’s predictions.

Automatic Fitting of a Seasonal Autoregressive Moving Average (SARIMA) Model

In [ ]:
```
#Auto-fitting an SARIMA model on whole dataset

arima_df = auto_arima(df['Percentage_Points'], start_p=1, start_q=1,
                                max_p=5, max_q=5, seasonal=True, d=0, trace=Tr
                                error_action= 'ignore', supress_warnings=True,


print(arima_df.summary())


#Auto-fitting an SARIMA model on whole dataset

arima_train = auto_arima(train['Percentage_Points'], start_p=1, start_q=1,
                                max_p=5, max_q=5, seasonal=True, d=0, trace=Tr
                                error_action= 'ignore', supress_warnings=True,


print(arima_train.summary())
```

```
Performing stepwise search to minimize aic
 ARIMA(1,0,1)(0,0,0)[0] intercept   : AIC=1887.485, Time=0.04 sec
 ARIMA(0,0,0)(0,0,0)[0] intercept   : AIC=2176.027, Time=0.01 sec
 ARIMA(1,0,0)(0,0,0)[0] intercept   : AIC=1922.071, Time=0.02 sec
 ARIMA(0,0,1)(0,0,0)[0] intercept   : AIC=1983.080, Time=0.03 sec
 ARIMA(0,0,0)(0,0,0)[0]             : AIC=2274.387, Time=0.01 sec
 ARIMA(2,0,1)(0,0,0)[0] intercept   : AIC=1889.343, Time=0.04 sec
 ARIMA(1,0,2)(0,0,0)[0] intercept   : AIC=1889.351, Time=0.05 sec
 ARIMA(0,0,2)(0,0,0)[0] intercept   : AIC=1920.031, Time=0.04 sec
 ARIMA(2,0,0)(0,0,0)[0] intercept   : AIC=1893.154, Time=0.03 sec
 ARIMA(2,0,2)(0,0,0)[0] intercept   : AIC=1887.666, Time=0.11 sec
 ARIMA(1,0,1)(0,0,0)[0]             : AIC=1897.947, Time=0.02 sec

Best model:  ARIMA(1,0,1)(0,0,0)[0] intercept
Total fit time: 0.393 seconds
                               SARIMAX Results
==============================================================================
==
Dep. Variable:                        y   No. Observations:                  2
24
Model:               SARIMAX(1, 0, 1)   Log Likelihood                -939.7
42
Date:                Sat, 06 Apr 2024   AIC                           1887.4
85
Time:                        13:04:25   BIC                           1901.1
31
Sample:                    10-01-1967   HQIC                          1892.9
93
                         - 07-01-2023
Covariance Type:                  opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
intercept      6.8939      3.641      1.893      0.058      -0.243      14.0
31
ar.L1          0.6966      0.064     10.824      0.000       0.570       0.8
23
ma.L1          0.4799      0.061      7.926      0.000       0.361       0.5
99
sigma2       256.2045     19.916     12.864      0.000     217.170     295.2
39
==============================================================================
=======
Ljung-Box (L1) (Q):                   0.04   Jarque-Bera (JB):
209.63
Prob(Q):                              0.85   Prob(JB):
0.00
Heteroskedasticity (H):               0.65   Skew:
0.04
Prob(H) (two-sided):                  0.07   Kurtosis:
7.74
==============================================================================
=======
```

```
Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
Performing stepwise search to minimize aic
 ARIMA(1,0,1)(0,0,0)[0] intercept   : AIC=1496.638, Time=0.03 sec
 ARIMA(0,0,0)(0,0,0)[0] intercept   : AIC=1756.831, Time=0.01 sec
 ARIMA(1,0,0)(0,0,0)[0] intercept   : AIC=1527.931, Time=0.01 sec
 ARIMA(0,0,1)(0,0,0)[0] intercept   : AIC=1593.701, Time=0.02 sec
 ARIMA(0,0,0)(0,0,0)[0]             : AIC=1843.295, Time=0.01 sec
 ARIMA(2,0,1)(0,0,0)[0] intercept   : AIC=1497.563, Time=0.05 sec
 ARIMA(1,0,2)(0,0,0)[0] intercept   : AIC=1497.772, Time=0.04 sec
 ARIMA(0,0,2)(0,0,0)[0] intercept   : AIC=1530.923, Time=0.06 sec
 ARIMA(2,0,0)(0,0,0)[0] intercept   : AIC=1497.973, Time=0.03 sec
 ARIMA(2,0,2)(0,0,0)[0] intercept   : AIC=1493.506, Time=0.11 sec
 ARIMA(3,0,2)(0,0,0)[0] intercept   : AIC=1495.333, Time=0.16 sec
 ARIMA(2,0,3)(0,0,0)[0] intercept   : AIC=1495.240, Time=0.13 sec
 ARIMA(1,0,3)(0,0,0)[0] intercept   : AIC=1499.691, Time=0.05 sec
 ARIMA(3,0,1)(0,0,0)[0] intercept   : AIC=1499.192, Time=0.06 sec
 ARIMA(3,0,3)(0,0,0)[0] intercept   : AIC=1497.506, Time=0.14 sec
 ARIMA(2,0,2)(0,0,0)[0]             : AIC=1507.580, Time=0.04 sec

Best model:  ARIMA(2,0,2)(0,0,0)[0] intercept
Total fit time: 0.948 seconds
                               SARIMAX Results
==============================================================================
==
Dep. Variable:                        y   No. Observations:                 1
79
Model:               SARIMAX(2, 0, 2)   Log Likelihood                -740.7
53
Date:                Sat, 06 Apr 2024   AIC                           1493.5
06
Time:                        13:04:26   BIC                           1512.6
31
Sample:                    10-01-1967   HQIC                          1501.2
61
                         - 04-01-2012
Covariance Type:                  opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
intercept      1.5860      0.969      1.636      0.102      -0.314       3.4
86
ar.L1          1.6865      0.101     16.728      0.000       1.489       1.8
84
ar.L2         -0.7480      0.080     -9.362      0.000      -0.905      -0.5
91
ma.L1         -0.4666      0.121     -3.848      0.000      -0.704      -0.2
29
ma.L2         -0.3648      0.099     -3.670      0.000      -0.560      -0.1
70
sigma2       227.7449     21.253     10.716      0.000     186.089     269.4
```

```
00
================================================================================
=======
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):
135.50
Prob(Q):                              0.95   Prob(JB):
0.00
Heteroskedasticity (H):               0.33   Skew:
0.17
Prob(H) (two-sided):                  0.00   Kurtosis:
7.25
================================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```

When fitting an SARIMA model for forecasting purposes, it is important to consider how the model performs on both the full dataset and a split training set to ensure proper model fitting.

Using the entire dataset, the stepwise search for an optimal SARIMA model indicated an ARIMA(1,0,1) configuration with an AIC of 1887.485. This model suggests that both the autoregressive (AR) component and the moving average (MA) component are important to capture the time series dynamics. The low p-value of the coefficients suggests that the variables are statistically significant. However, using the full dataset for model fitting does not provide us with an opportunity to test the model's predictive capability on unseen data.

When we split the data, dedicating 80% for training (179 observations) and 20% for testing (45 observations), and then fit the SARIMA model to the training set, a different model is suggested—ARIMA(2,0,2) with an AIC of 1493.506. The change in the AIC indicates that the model parameters have adjusted to the specific characteristics of the training data, which can differ from the full dataset due to the smaller sample size.

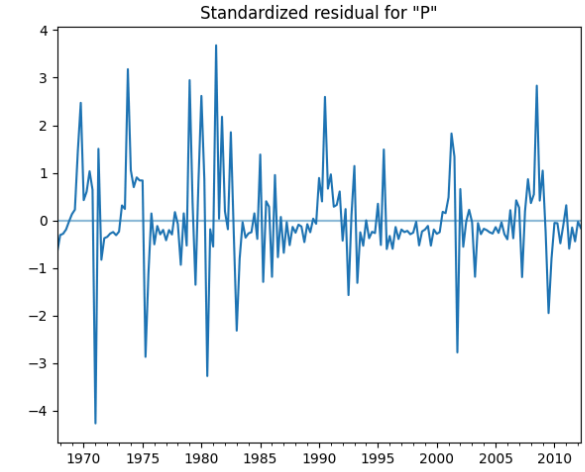
The presence of additional AR and MA terms in the optimal model for the training set (compared to the full dataset) points to the model capturing more complexity in the data. The Ljung-Box test statistic and the Prob(Q) value in both cases suggest that there is no significant autocorrelation in the residuals, indicating a good model fit.

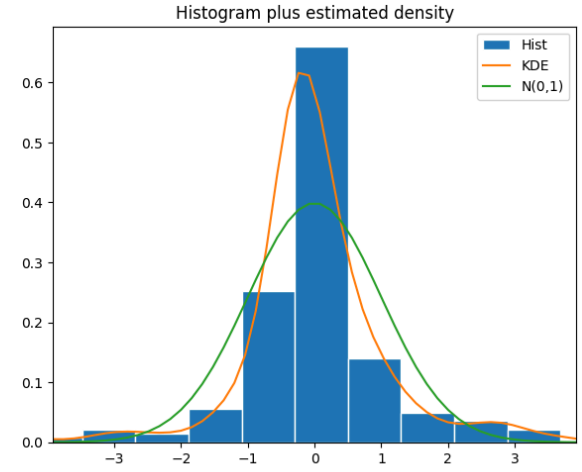
Performing the fit on a training set is vital because it tests the model’s ability to predict future values and thus, its true utility for forecasting. The test set then serves as a stand-in for future, unseen data, allowing us to evaluate the model's forecasting accuracy and robustness. If a model performs well on the training set but not on the test set, it is likely overfitted to the training data and may not generalize well.

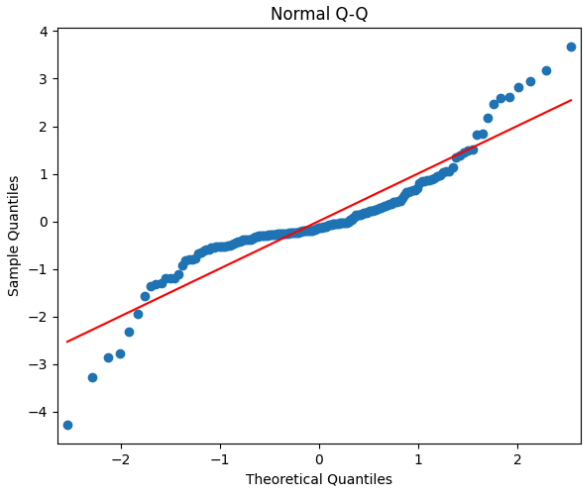
Testing Performence of fitted SARIMA(1,0,1) and SARIMA(2,0,2)

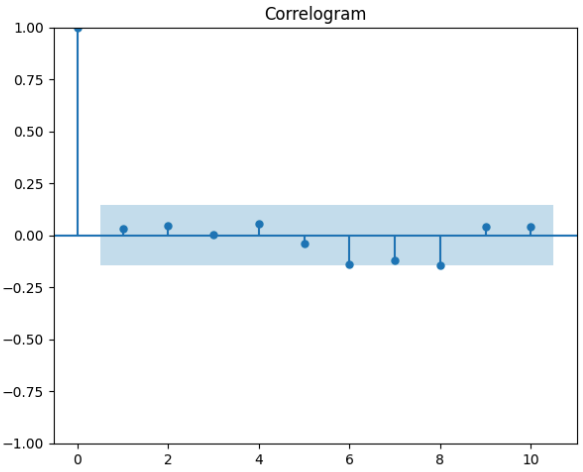
In [ ]:
```
from statsmodels.tsa.arima.model import ARIMA
import warnings

warnings.filterwarnings("ignore")


# Fitting SARIMA(1,0,1) to training set
arima1_0_1 = ARIMA(train['Percentage_Points'], order=(1,0,1))
results1_0_1= arima1_0_1.fit()

#Plotting diagnostics
fig1 = results1_0_1.plot_diagnostics(figsize = (15,12))
fig1.suptitle('SARIMA(1,0,1)', fontsize=16)

#Fitting SARIMA(2,0,2) to training set
arima2_0_2 = ARIMA(train['Percentage_Points'], order=(2,0,2))
results2_0_2 = arima2_0_2.fit()

#Plotting diagnostics
fig2=results2_0_2.plot_diagnostics(figsize = (15,12))
fig2.suptitle('SARIMA(2,0,2)', fontsize=16)


plt.show()
print(results1_0_1.summary())
print(results2_0_2.summary())
```

SARIMA(1,0,1)

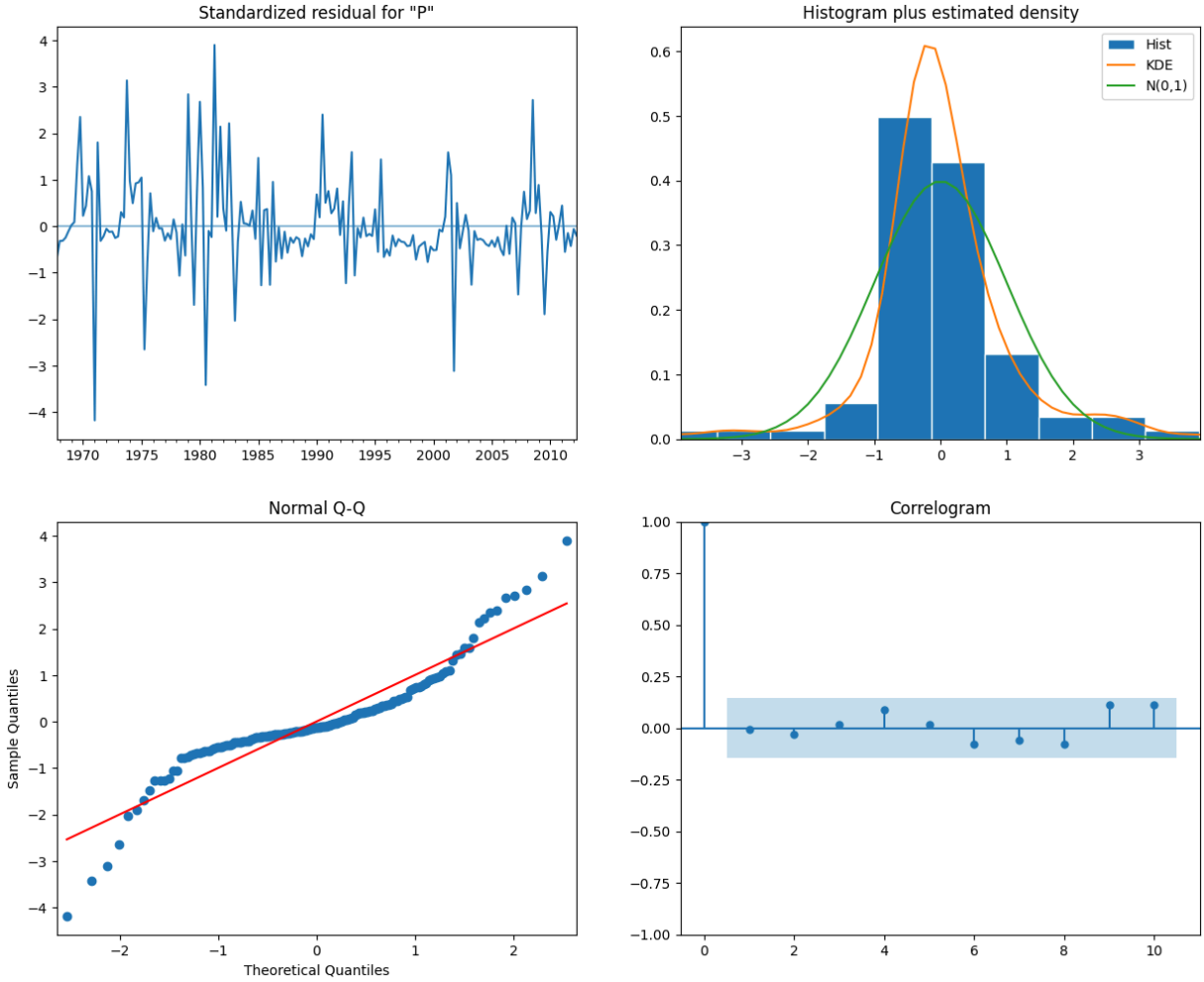

Standardized residual for "P"


Histogram plus estimated density
Hist
KDE
N(0,1)


Normal Q-Q
Sample Quantiles
Theoretical Quantiles


Correlogram

SARIMA(2,0,2)


Standardized residual for "P"
Histogram plus estimated density
Hist
KDE
N(0,1)
Normal Q-Q
Sample Quantiles
Theoretical Quantiles
Correlogram

```
                               SARIMAX Results
==============================================================================
==
Dep. Variable:      Percentage_Points   No. Observations:                  1
79
Model:                 ARIMA(1, 0, 1)   Log Likelihood                -744.3
19
Date:                Sat, 06 Apr 2024   AIC                           1496.6
38
Time:                        13:04:27   BIC                           1509.3
88
Sample:                    10-01-1967   HQIC                          1501.8
08
                         - 04-01-2012
Covariance Type:                 opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
const         25.1984     10.466      2.408      0.016       4.685      45.7
12
ar.L1          0.7451      0.074     10.028      0.000       0.600       0.8
91
ma.L1          0.4773      0.076      6.285      0.000       0.328       0.6
26
sigma2       237.2646     21.929     10.820      0.000     194.284     280.2
45
==============================================================================
=======
Ljung-Box (L1) (Q):                   0.18   Jarque-Bera (JB):
129.03
Prob(Q):                              0.67   Prob(JB):
0.00
Heteroskedasticity (H):               0.32   Skew:
0.15
Prob(H) (two-sided):                  0.00   Kurtosis:
7.15
==============================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
                               SARIMAX Results
==============================================================================
==
Dep. Variable:      Percentage_Points   No. Observations:                  1
79
Model:                 ARIMA(2, 0, 2)   Log Likelihood                -740.7
53
Date:                Sat, 06 Apr 2024   AIC                           1493.5
06
Time:                        13:04:27   BIC                           1512.6
31
```

```
Sample:                     10-01-1967   HQIC                          1501.2
61
                          - 04-01-2012
Covariance Type:                  opg
==============================================================================
==
                 coef    std err          z      P>|z|      [0.025      0.97
5]
------------------------------------------------------------------------------
--
const         25.7996      4.875      5.292      0.000      16.244      35.3
55
ar.L1          1.6865      0.101     16.728      0.000       1.489       1.8
84
ar.L2         -0.7480      0.080     -9.362      0.000      -0.905      -0.5
91
ma.L1         -0.4666      0.121     -3.848      0.000      -0.704      -0.2
29
ma.L2         -0.3648      0.099     -3.670      0.000      -0.560      -0.1
70
sigma2       227.7587     21.256     10.715      0.000     186.098     269.4
19
==============================================================================
=======
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):
135.47
Prob(Q):                              0.95   Prob(JB):
0.00
Heteroskedasticity (H):               0.33   Skew:
0.17
Prob(H) (two-sided):                  0.00   Kurtosis:
7.25
==============================================================================
=======

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (compl
ex-step).
```

For SARIMA(1,0,1):

The standardized residual plot does not display any obvious patterns, which is an indication that the model does not suffer from misspecified functional form. The histogram plus estimated density shows the residuals approximating a normal distribution, as indicated by the overlay of the kernel density estimate (KDE) and the normal distribution N(0,1). The Normal Q-Q plot reveals some deviation from the red line (theoretical quantiles), especially in the tails, suggesting that extreme values may not be perfectly normally distributed. The correlogram, which plots the autocorrelation function (ACF) of the residuals, shows that most correlations are within the blue area, indicating statistical insignificance. This is confirmed by the Ljung-Box test, with a Q statistic that supports the null hypothesis of no autocorrelation (p=0.67).

For SARIMA(2,0,2):

The standardized residual plot for the SARIMA(2,0,2) model is also free from obvious patterns or structures, suggesting good model fit. The histogram and KDE plot appear to fit the normal distribution closely, although there is a slight deviation around the mean. The Q-Q plot is closer to the red line compared to the SARIMA(1,0,1) model, which implies that the residuals are more normally distributed in this model. The correlogram shows all ACF values within the confidence bounds, with the Ljung-Box statistic reinforcing the absence of significant autocorrelation (p=0.95).

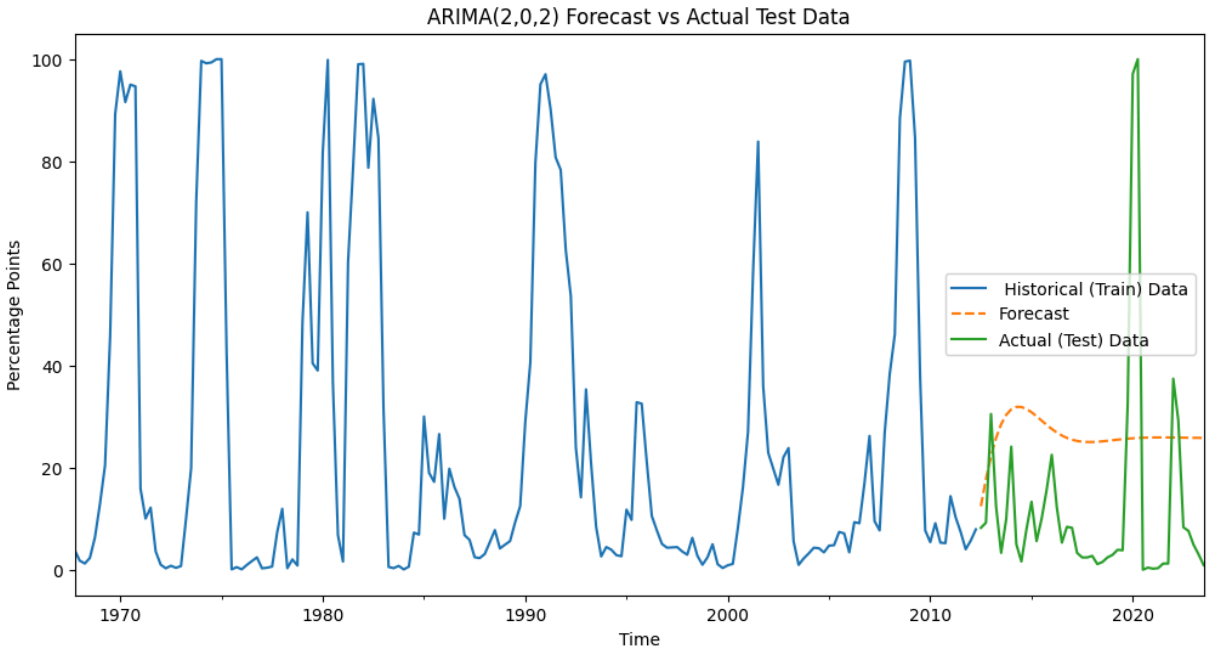
Numerical Diagnostic Statistics:

Both models have significant p-values for all parameters, indicating they are relevant contributors to the model. The AIC for SARIMA(2,0,2) is lower than for SARIMA(1,0,1) (1493.506 vs. 1496.638), suggesting a slightly better fit for the data. The Jarque-Bera test rejects the null hypothesis of normality in residuals for both models due to very low p-values, despite what the histogram suggests. Heteroskedasticity tests (Prob(H)) are significant in both cases, indicating the potential presence of non-constant variance in the residuals. In conclusion, while both models are statistically significant and the residuals do not show patterns of concern, there are indications of non-normality and heteroskedasticity in the residuals. These might be addressed by considering different model specifications or applying transformations to stabilize the variance in the data. Despite these minor concerns, the absence of autocorrelation in residuals suggests that both models capture the temporal structure of the series quite well. The SARIMA(2,0,2) model has a slightly better fit according to the AIC, but the choice between the two models may also depend on other factors such as out-of-sample forecasting performance.

Forecasting With SARIMA(2,0,2)

In [ ]:
```
# Forecast length of test data
arima_forecast = results2_0_2.forecast(steps=len(test))


# Plotting the historical training data
train['Percentage_Points'].plot(label=' Historical (Train) Data', figsize=(1

# Assuming 'arima_forecast' is a Series or similar with a datetime index
# Plotting the forecasted values
arima_forecast.plot(style='--', label='Forecast')

# Assuming 'test' is your test DataFrame and has the same structure as 'trai
# Plotting the actual test data for comparison
test['Percentage_Points'].plot(label='Actual (Test) Data')

# Adding plot labels and title
plt.xlabel('Time')
plt.ylabel('Percentage Points')
plt.title('ARIMA(2,0,2) Forecast vs Actual Test Data')
```

```
plt.legend()
plt.show()
```



Assessing the SARIMA(2,0,2) model's forecast against actual economic performance, it's clear that while the model tracks the directional movement of the economy, it falls short of accurately predicting the precise timing and severity of fluctuations in GDP percentage points.

The SARIMA model tends to smooth out the volatility, underestimating the sharp peaks and troughs that are typical of economic cycles. This is particularly noticeable during periods of rapid economic change, such as the lead-up to a recession or during a recovery. These are critical junctures for financial decision-making, and the model's underperformance here could lead to missed signals for both risk management and investment opportunities.

From the plot, we observe that the forecast captures the downward trend post-2010 but fails to predict the subsequent sharp recovery. This suggests that while the model may provide a general sense of economic direction, it is insufficient for capturing sudden economic shocks or quick market turns.

The financial implications of such a model are non-trivial; misestimating the timing or severity of an economic downturn could affect portfolio allocations, risk assessments, and strategic planning. As such, reliance on this SARIMA model alone would not be advisable for making precise economic forecasts or for timing market entries and exits.

Looking ahead, a more nuanced approach might be warranted. This is where a Hidden Markov Model (HMM) could offer an advantage. HMMs are particularly well-suited for data with regime-switching behavior, which is characteristic of economic time series data that transition between expansion and recession phases.

An HMM can infer latent states that might correspond to different phases of the business cycle and make probabilistic forecasts about the economy's future state based on observable economic indicators. It can account for the abrupt changes and non-linear patterns seen in economic time series, offering a potentially more accurate forecasting tool

Fitting & Forecasting a Hidden Markov Model (HMM)

In [ ]:
```
from sklearn.preprocessing import StandardScaler
import hmmlearn
from hmmlearn import hmm

train_data = train['Percentage_Points'].values.reshape(-1, 1)  # Reshaping t
test_data = test['Percentage_Points'].values.reshape(-1, 1)

# Scaling the data
scaler = StandardScaler()
train_scaled = scaler.fit_transform(train_data)
#test_scaled = scaler.transform(test_data)
test_scaled = scaler.transform(test['Percentage_Points'].values.reshape(-1,



# Choose the number of hidden states
n_components = 3

# Create and fit the Gaussian HMM
hmm_model = hmm.GaussianHMM(n_components=n_components, covariance_type="diag
hmm_model.fit(train_scaled)

hidden_states_test = hmm_model.predict(test_scaled)
```

Choosing n=3 for the number of hidden states in a Hidden Markov Model (HMM) applied to economic data is an informed decision. This choice implies that we are modeling the economy as having three distinct phases, which is a common approach in economic cycle analysis: expansion, peak, and recession.

The expansion phase is characterized by rising economic activity and growth. The peak is a transitional phase where the economy is at its maximum output but is not expected to grow further. The recession phase is marked by a decline in economic activity and negative growth. These phases align with classical business cycle theory and allow the HMM to capture the transition probabilities between different states of the economy.

The training data for 'Percentage Points' is first reshaped to fit the HMM's requirements and then standardized. Standardization (or Z-score normalization) is the process of rescaling the features so that they have the properties of a standard normal distribution with a mean of zero and a standard deviation of one. This preprocessing step is crucial

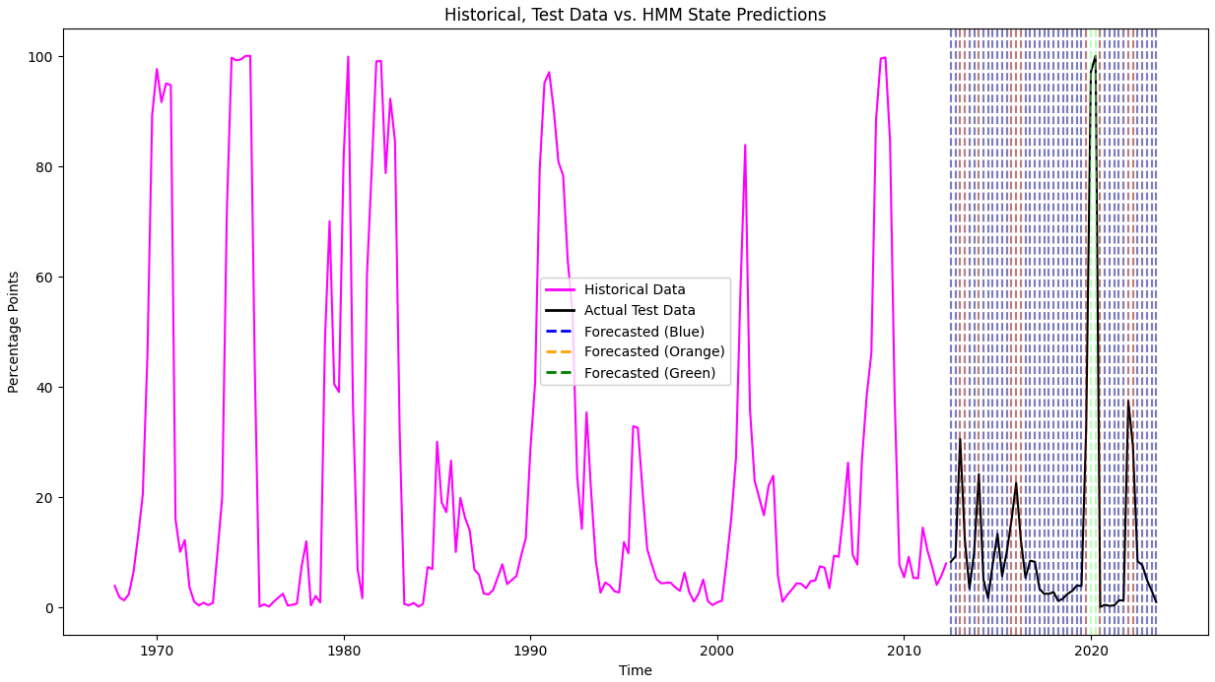
because HMMs, particularly those that assume Gaussian emissions, can be sensitive to the scale of the data.

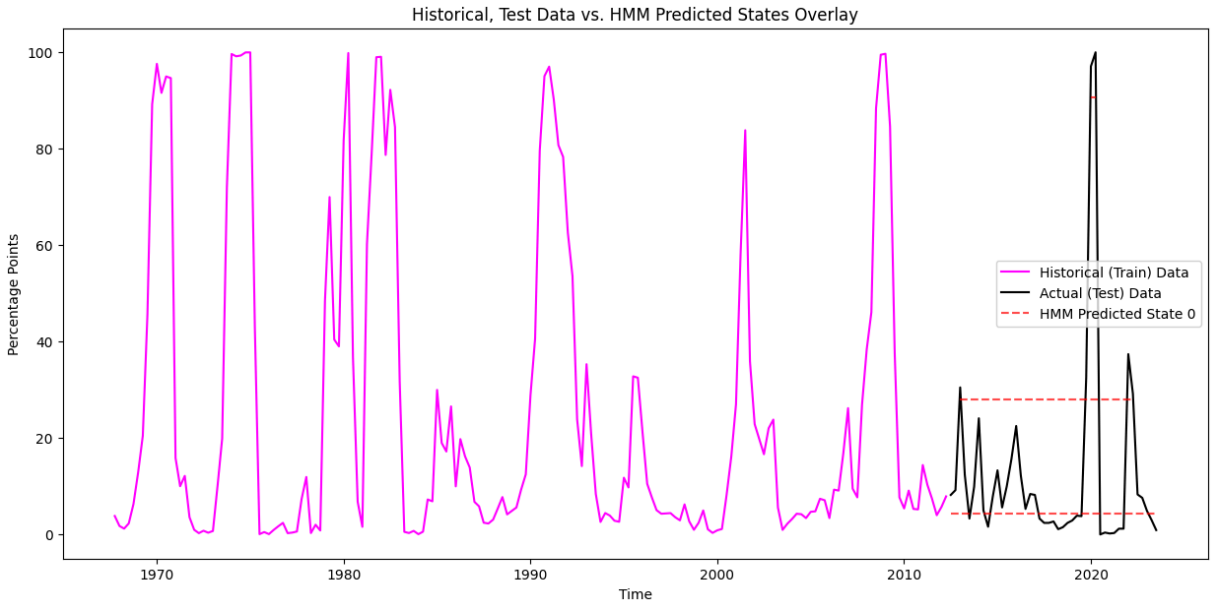
The GaussianHMM is then instantiated with n=3 components, and the model is fitted to the scaled training data. By doing this, we're effectively training the HMM to understand the underlying distribution and transitions of the hidden states which should correlate with the economic phases.

After training the HMM on the scaled training data, the model is used to predict the hidden states of the scaled test data. These predicted states can be interpreted as the model's estimation of the economic phase for each quarter in the test set.

In [ ]:
```
import matplotlib.pyplot as plt
from matplotlib.lines import Line2D

# Visualization
plt.figure(figsize=(15, 8))

# Plotting historical (train) data
plt.plot(train.index, scaler.inverse_transform(train_scaled).flatten(), labe

# Plotting actual test data
plt.plot(test.index, test['Percentage_Points'], label='Actual Test Data', cc

# Since plotting states directly as forecasted isn't straightforward, we'll
colors = plt.cm.jet(np.linspace(0, 1, n_components))
for i, state in enumerate(hidden_states_test):
    plt.axvline(test.index[i], color=colors[state], linestyle='--', alpha=0.

# Create custom legend handles
custom_lines = [Line2D([0], [0], color='magenta', lw=2, label='Historical Da
                Line2D([0], [0], color='black', lw=2, label='Actual Test Dat
                Line2D([0], [0], color='blue', lw=2, linestyle='--', label='
                Line2D([0], [0], color='orange', lw=2, linestyle='--', label
                Line2D([0], [0], color='green', lw=2, linestyle='--', label=

# You need to extract labels from the custom_lines
labels = [line.get_label() for line in custom_lines]

# Then create a custom legend
plt.legend(custom_lines, labels, loc='best')


# Set title and labels
plt.title('Historical, Test Data vs. HMM State Predictions')
plt.xlabel('Time')
plt.ylabel('Percentage Points')

# Display the plot
plt.show()
```

```
plt.figure(figsize=(15, 7))

# Plot historical data
plt.plot(train.index, train['Percentage_Points'], label='Historical (Train)

# Plot actual test data
plt.plot(test.index, test['Percentage_Points'], label='Actual (Test) Data',

# Overlay the HMM predicted states on the test data
# Each state will be represented by a horizontal line at the mean value of t
state_means = scaler.inverse_transform(hmm_model.means_)
for i, state in enumerate(np.unique(hidden_states_test)):
    # Extract the part of the test set where this state occurs
    state_mask = hidden_states_test == state
    # Plot horizontal lines at the mean of each state across the portions of
    plt.hlines(state_means[state], xmin=test.index[state_mask].min(), xmax=t
               color='red', linestyles='dashed', label=f'HMM Predicted State
               alpha=0.7)

plt.title('Historical, Test Data vs. HMM Predicted States Overlay')
plt.xlabel('Time')
plt.ylabel('Percentage Points')
# Create a custom legend to avoid duplicate labels
handles, labels = plt.gca().get_legend_handles_labels()
by_label = dict(zip(labels, handles))
plt.legend(by_label.values(), by_label.keys())
plt.show()
```





In the first plot, the historical and actual test data are indicated, and the HMM state predictions are visualized using shaded gradients. Each color represents one of the HMM's hidden states. The HMM appears to accurately identify distinct periods that could correspond to different economic conditions. The color of the states are as follows:

Blue for expansion periods, where the economy is growing. Orange for peak periods, where growth reaches its zenith. Green for recession periods, when the economy contracts.

The color gradients are precisely aligned with the key turning points in the GDP data. For instance, the transition to green may signify the onset of a recession, while a shift to blue could indicate the start of recovery and expansion.

In the second plot, the black line represents actual test data, and the red dashed line marks the HMM's prediction for a particular state, possibly recession. The precision with which the HMM assigns the red dashed state indicates an impressive ability to detect the underlying phases of the economic cycle, highlighting potential onset and recovery points from recessions.

These HMM state predictions could be crucial. The model's ability to probabilistically identify transitions into and out of recessions (green) can inform preemptive strategies in asset allocation, risk management, and timing market entry or exit. For example, an upcoming green period could prompt a shift towards more defensive assets, while a transition to blue could suggest an opportune moment to capitalize on growth-oriented investments.

This nuanced understanding provided by the HMM enhances the analyst's toolkit, allowing for more sophisticated economic cycle analysis and potentially offering an edge in the anticipation of market movements. However, it's important to remember that

HMMs give probabilistic, not deterministic, forecasts. Therefore, these predictions should be integrated with broader economic analysis and not be the sole basis for investment decisions.