Importing Necessary Libraries

In [78]:
```
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import skew, kurtosis
from statsmodels.tsa.regime_switching.markov_regression import MarkovRegression
```

Setting Working Directory

In [79]:
```
os.chdir(r"C:\Users\jeswi\OneDrive\Desktop\Quantitative Research\Momentum Strategy with Regime Switching")
print("Working directory:", os.getcwd())
```

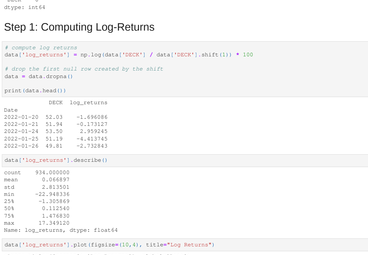
```
Working directory: C:\Users\jeswi\OneDrive\Desktop\Quantitative Research\Momentum Strategy with Regime Switching
```

Loading the Data

In [80]:
```
data = pd.read_csv("Data.csv")
data['Date'] = pd.to_datetime(data['Date'], format="%d-%m-%Y")
data = data.set_index('Date').sort_index()

print(data.head())
```

```
            DECK
Date
2022-01-19  52.92
2022-01-20  52.03
2022-01-21  51.94
2022-01-24  53.50
2022-01-25  51.19
```

In [81]:
```
plt.figure(figsize=(10,4))
plt.plot(data.index, data['DECK'], color='blue', linewidth=1.5)
plt.title("DECK Closing Price Over Time")
plt.xlabel("Time")
plt.ylabel("DECK")
plt.show()
```



In [82]:
```
missing_counts = data.isnull().sum()
print("Missing values per column:\n", missing_counts)
```

```
Missing values per column:
 DECK    0
dtype: int64
```

Step 1: Computing Log-Returns

In [83]:
```
# compute log returns
data['log_returns'] = np.log(data['DECK'] / data['DECK'].shift(1)) * 100

# drop the first null row created by the shift
data = data.dropna()

print(data.head())
```

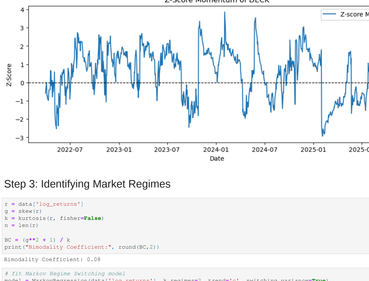
```
             DECK  log_returns
Date
2022-01-20  52.03    -1.696086
2022-01-21  51.94    -0.173127
2022-01-24  53.50     2.959245
2022-01-25  51.19    -4.413745
2022-01-26  49.81    -2.732843
```

In [84]:
```
data['log_returns'].describe()
```

Out[84]:
```
count    934.000000
mean       0.066897
std        2.815501
min      -22.948336
25%       -1.305869
50%        0.112540
75%        1.476830
max       17.349120
Name: log_returns, dtype: float64
```

In [85]:
```
data['log_returns'].plot(figsize=(10,4), title="Log Returns")
```

Out[85]:
```
<Axes: title={'center': 'Log Returns'}, xlabel='Date'>
```



Step 2: Momentum Signals - Z score

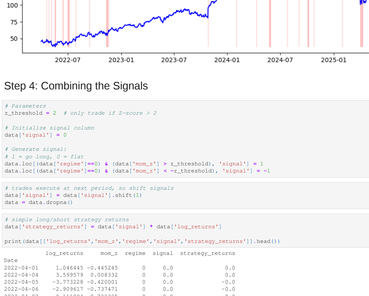
In [86]:
```
# set rolling window (you can adjust later)
window = 50

# compute rolling mean and rolling std
rolling_mean = data['DECK'].rolling(window=window).mean()
rolling_std = data['DECK'].rolling(window=window).std()

# compute z-score momentum
data['mom_z'] = (data['DECK'] - rolling_mean) / rolling_std

# drop initial rows with NaN (first 'window' rows)
data = data.dropna()

print(data.head())
```

```
             DECK  log_returns     mom_z
Date
2022-03-31  45.63    -2.745228 -0.592902
2022-04-01  46.11     1.046445 -0.445245
2022-04-04  47.80     3.599579  0.008332
2022-04-05  46.03    -3.773228 -0.420001
2022-04-06  44.71    -2.909417 -0.737471
```

In [87]:
```
print(data['mom_z'].describe())
```

```
count    885.000000
mean       0.526499
std        1.326205
min       -2.932925
25%       -0.593985
50%        0.792685
75%        1.567039
max        3.861983
Name: mom_z, dtype: float64
```

In [88]:
```
plt.figure(figsize=(11,4))
plt.plot(data.index, data['mom_z'], label="Z-score Momentum")
plt.axhline(0, color='black', linestyle='--', linewidth=1)
plt.title("Z-Score Momentum of DECK")
plt.xlabel("Date")
plt.ylabel("Z-Score")
plt.legend()
plt.show()
```



Step 3: Identifying Market Regimes

In [89]:
```
r = data['log_returns']
g = skew(r)
k = kurtosis(r, fisher=False)
n = len(r)

BC = (g**2 + 1) / k
print("Bimodality Coefficient:", round(BC,2))
```

```
Bimodality Coefficient: 0.08
```

In [90]:
```
# fit Markov Regime Switching model
model = MarkovRegression(data['log_returns'], k_regimes=2, trend='c', switching_variance=True)
result = model.fit()

# print summary
print(result.summary())
```

```
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has been provided, but it has no asso
ciated frequency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
                        Markov Switching Model Results
==============================================================================
Dep. Variable:          log_returns   No. Observations:                  885
Model:         MarkovRegression   Log Likelihood               -2020.152
Date:            Thu, 20 Nov 2025   AIC                           4052.304
Time:                    15:15:59   BIC                           4081.018
Sample:                         0   HQIC                          4063.282
                              - 885
Covariance Type:            approx
                        Regime 0 parameters
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.0770      0.075      1.033      0.301      -0.069       0.223
sigma2         3.8871      0.313     12.438      0.000       3.275       4.500
                        Regime 1 parameters
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.1735      0.962      0.180      0.857      -1.713       2.060
sigma2        53.6407     15.774      3.401      0.001      22.725      84.557
                      Regime transition parameters
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
p[0->0]        0.9542      0.016     60.970      0.000       0.924       0.985
p[1->0]        0.5597      0.188      2.973      0.003       0.191       0.929
==============================================================================

Warnings:
[1] Covariance matrix calculated using numerical (complex-step) differentiation.
```

In [91]:
```
smoothed_probs = result.smoothed_marginal_probabilities[1]
data['regime_prob'] = smoothed_probs

# assign regime based on probability > 0.5
data['regime'] = (data['regime_prob'] > 0.5).astype(int)

print(data[['log_returns','mom_z','regime','regime_prob']].head())
```

```
            log_returns     mom_z  regime  regime_prob
Date
2022-03-31    -2.745228 -0.592902       0     0.039760
2022-04-01     1.046445 -0.445245       0     0.023693
2022-04-04     3.599579  0.008332       0     0.069968
2022-04-05    -3.773228 -0.420001       0     0.087937
2022-04-06    -2.909417 -0.737471       0     0.047570
```

In [92]:
```
plt.figure(figsize=(12,4))
plt.plot(data['DECK'], label='DECK Price', color='blue')
plt.fill_between(data.index, data['DECK'].min(), data['DECK'].max(),
                 where=data['regime']==1, color='red', alpha=0.2,
                 label="High-Volatility Regime")
plt.title("DECK Price with Detected Regimes (Markov)")
plt.legend()
plt.show()
```


Step 4: Combining the Signals

In [93]:
```
# Parameters
z_threshold = 2  # only trade if Z-score > 2

# Initialize signal column
data['signal'] = 0

# Generate signal:
# 1 -> go long, 0 -> flat
data.loc[(data['regime']==0) & (data['mom_z'] > z_threshold), 'signal'] = 1
data.loc[(data['regime']==0) & (data['mom_z'] <= -z_threshold), 'signal'] = -1
```

In [94]:
```
# trades execute at next period, so shift signals
data['signal'] = data['signal'].shift(1)
data = data.dropna()
```

In [95]:
```
# simple long/short strategy returns
data['strategy_returns'] = data['signal'] * data['log_returns']

print(data[['log_returns','mom_z','regime','signal','strategy_returns']].head())
```

```
            log_returns     mom_z  regime  signal  strategy_returns
Date
2022-04-01     1.046445 -0.445245       0     0.0               0.0
2022-04-04     3.599579  0.008332       0     0.0               0.0
2022-04-05    -3.773228 -0.420001       0     0.0              -0.0
2022-04-06    -2.909417 -0.737471       0     0.0              -0.0
2022-04-07    -0.111894 -0.722325       0     0.0              -0.0
```

In [96]:
```
# get path to Downloads folder
downloads_path = os.path.join(os.path.expanduser("~"), "Downloads")

# filename
file_path = os.path.join(downloads_path, "DECK_strategy_data.csv")

# save dataframe
data.to_csv(file_path)

print(f"Data saved to: {file_path}")
```

```
Data saved to: C:\Users\jeswi\Downloads\DECK_strategy_data.csv
```