Part 1 - Import NVIDIA stock data using quantmod and visualize the closing prices over time.

Load necessary packages

```
library(quantmod)
```

Importing NVIDIA stock data using quantmod

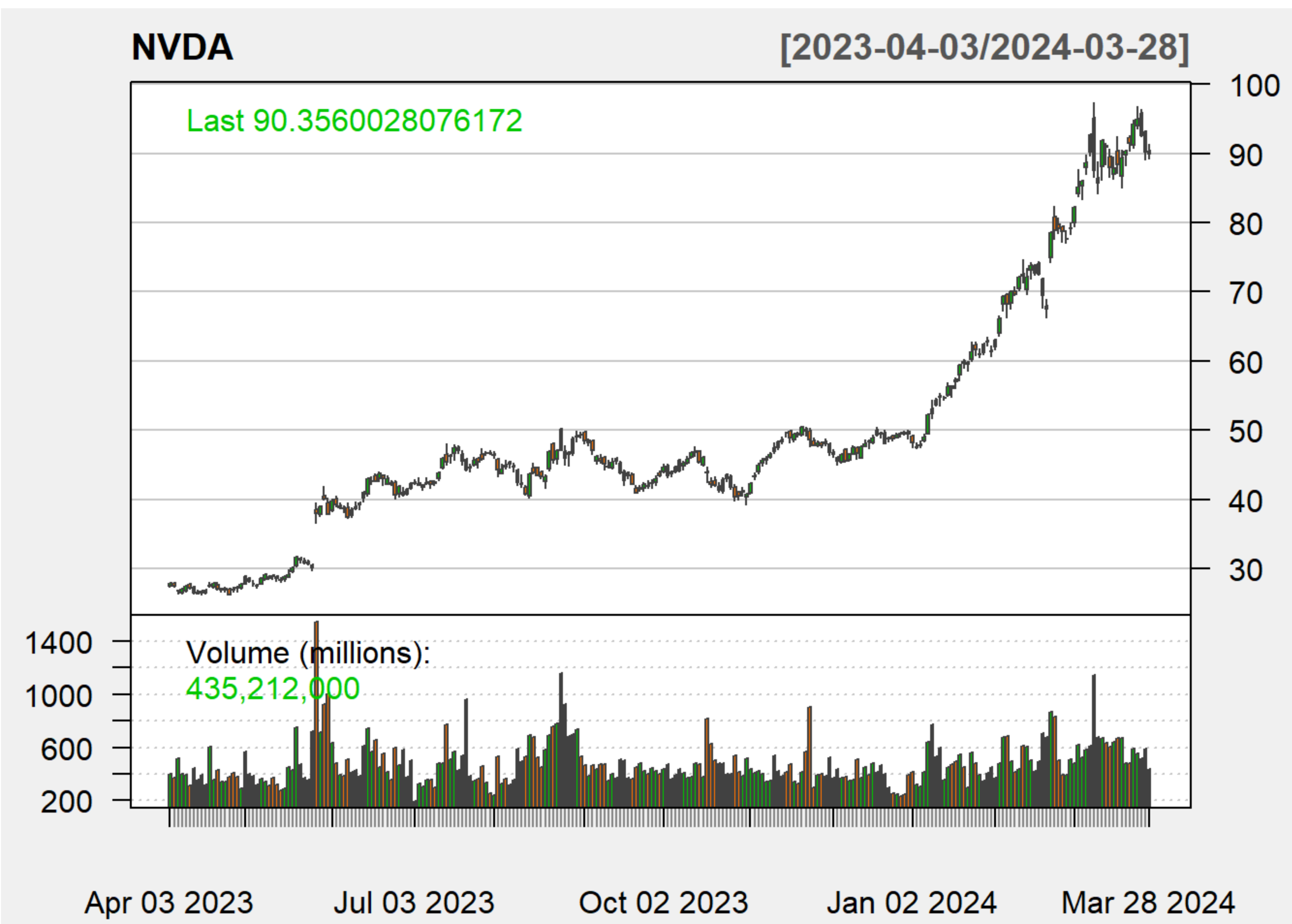
```
getSymbols("NVDA", src = "yahoo", from = "2023-04-01", to = "2024-04-01")
```

```
## [1] "NVDA"
```

Visualization of NVIDIA's Gains & Losses over a period of 1 year

Green = gains; Red = losses

```
chartSeries(NVDA, theme = chartTheme("white"))
```



Part 2 - Calculate daily and monthly returns for NVIDIA and Create plots showing price trends and returns.

Calculate daily returns using the quantmod package with the function dailyReturn().

This function computes the percentage change between the closing prices of two consecutive days (y axis = rate of change).

```
NVDA_daily_returns <- dailyReturn(NVDA)
```

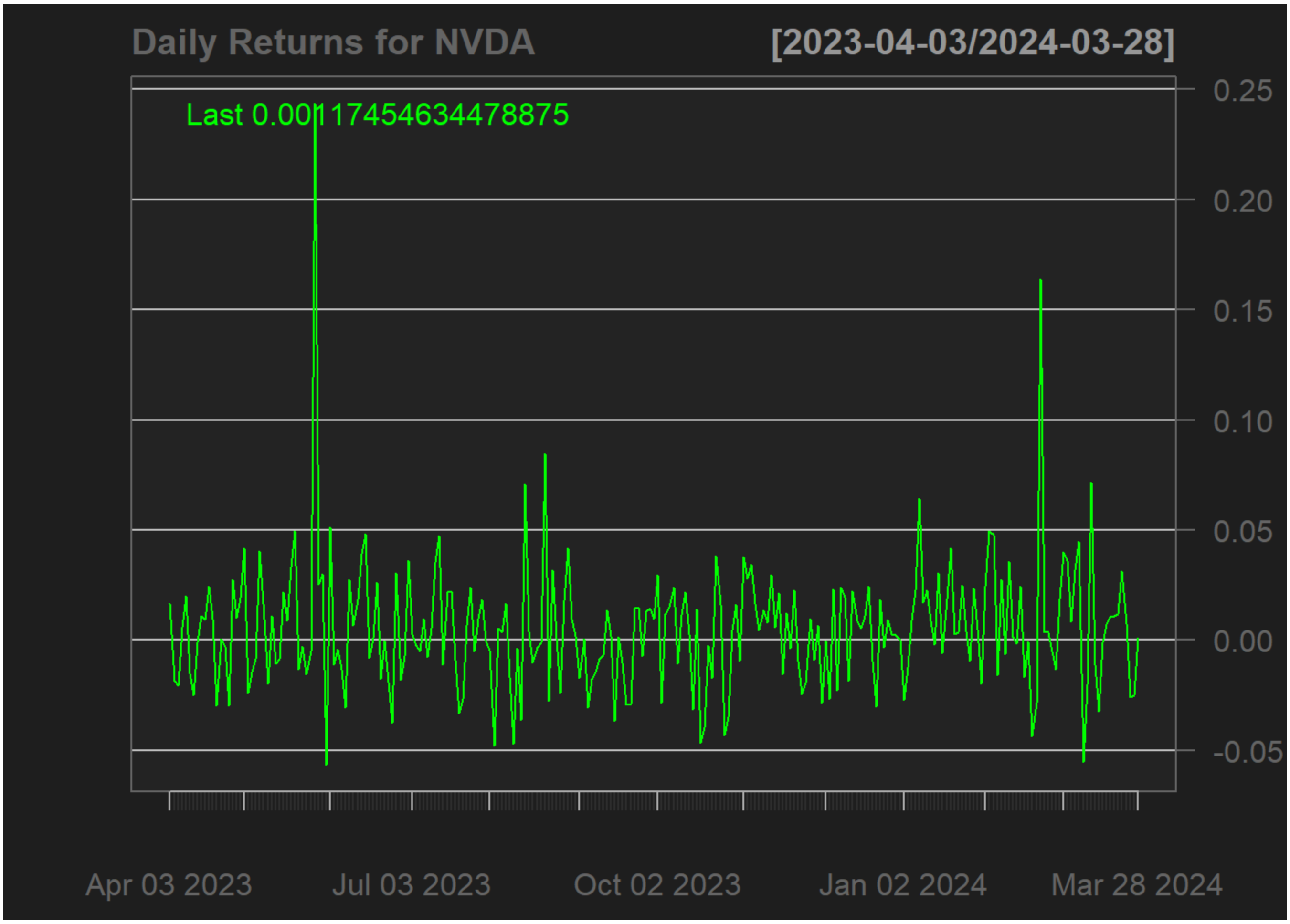
Calculate monthly returns using the quantmad package.

Involves converting daily stock prices to monthly prices and then computing the returns based on these monthly prices.

```
NVDA_monthly_returns <- monthlyReturn(NVDA)
```

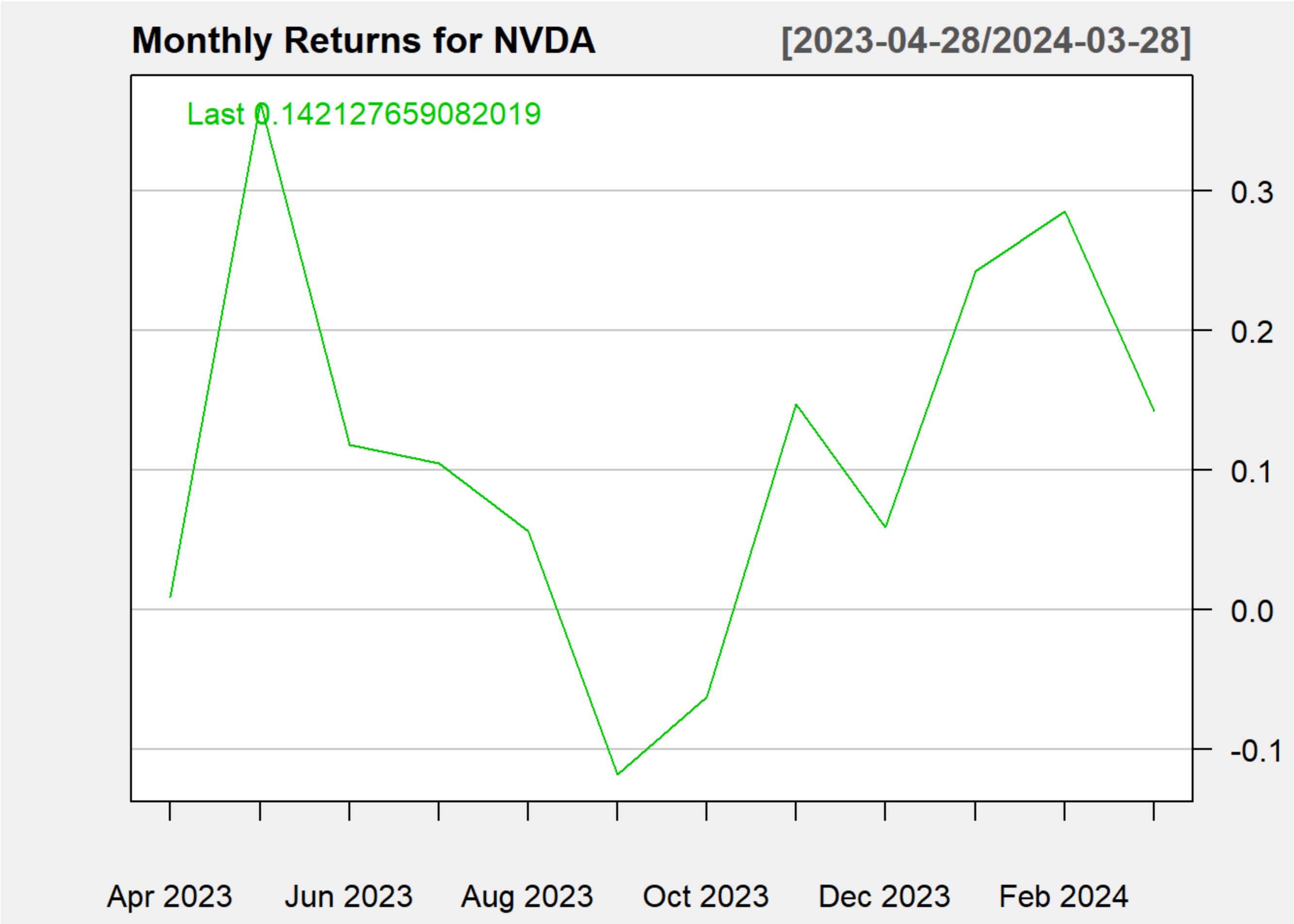
Visualizing daily returns of NVDA

```
barChart(NVDA_daily_returns, theme = chartTheme("black", grid.col="gray"), name =  "Daily Return
s for NVDA")
```



Visualizing monthly returns of NVDA

```
barChart(NVDA_monthly_returns, theme = chartTheme("white"), name = "Monthly Returns for NVDA")
```



Part 3 - Compute and plot moving averages and volatility for NVIDIA and Apply technical indicators to generate trading signals.

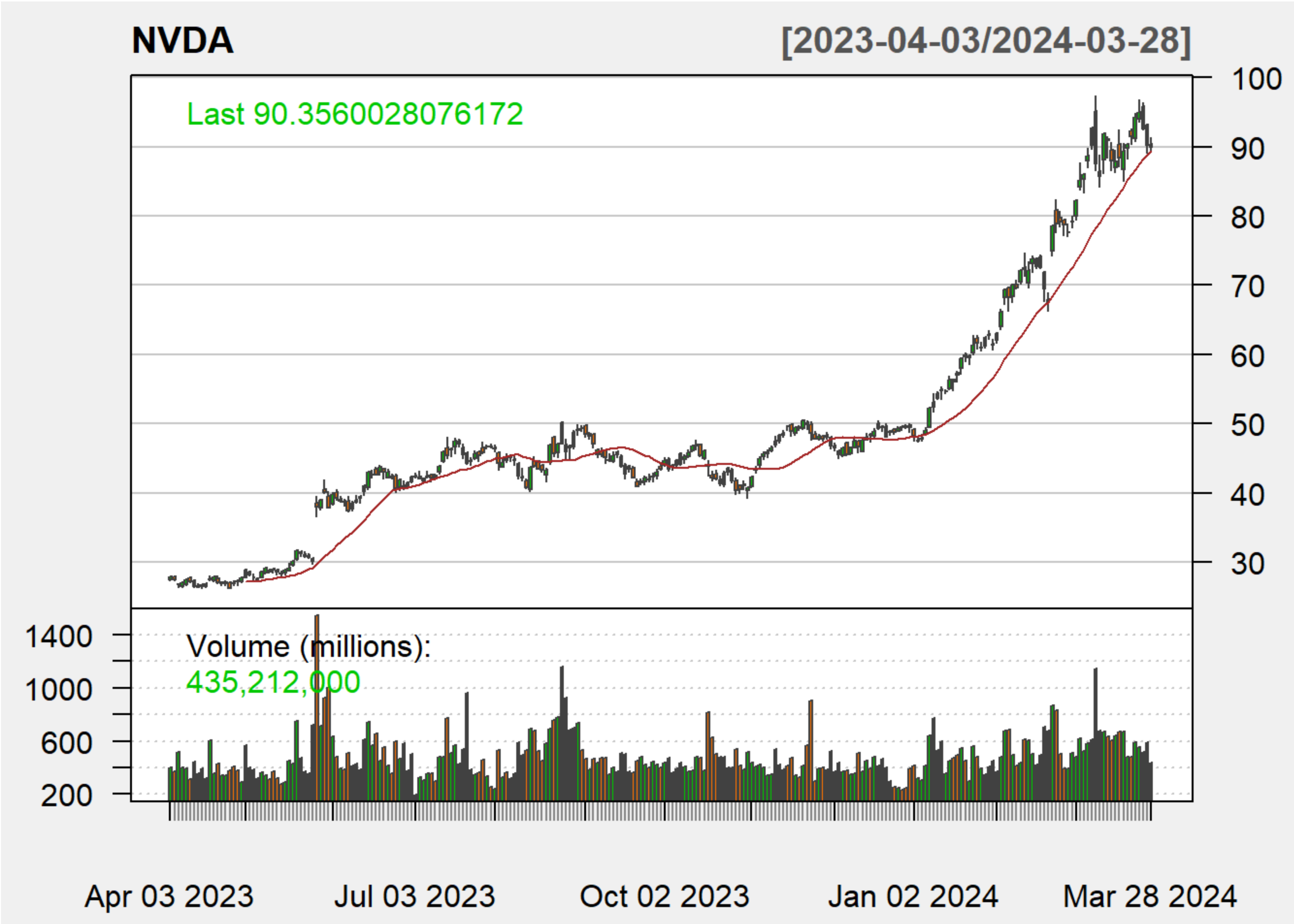
Moving averages

SMA calculates the arithmetic mean of the series over the past n observations.

Enhanced chart with volume and SMA overlays

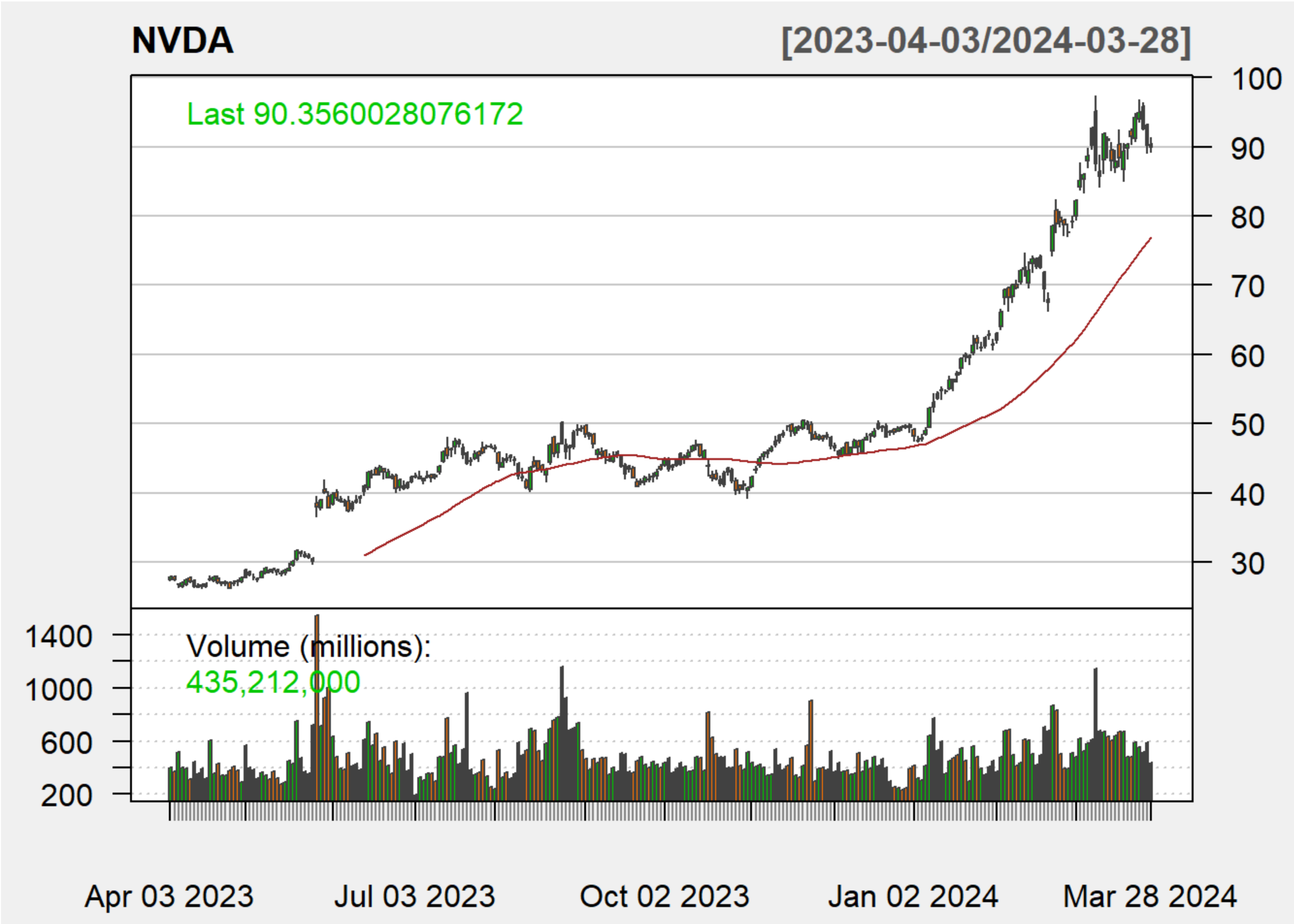
Visualization with SMA = 20

```
chartSeries(NVDA, TA = "addVo();addSMA(20)",theme = chartTheme("white"))
```



Visualization with SMA = 50

```
chartSeries(NVDA, TA = "addVo();addSMA(50)",theme = chartTheme("white"))
```



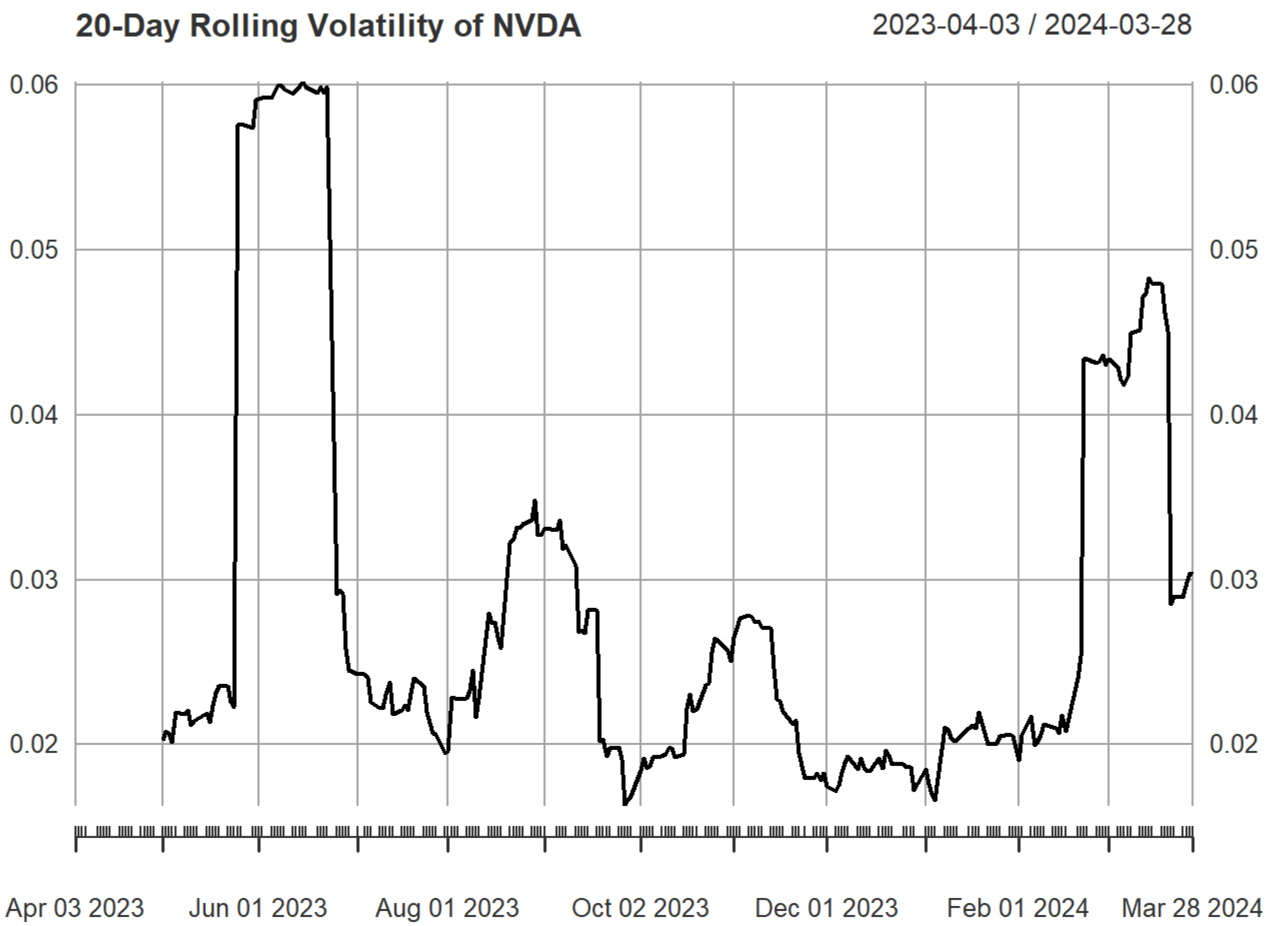
Volatility (standard deviation of daily returns)

Volatility is a statistical measure of the dispersion of returns for a given security or market index, and it's typically used to quantify the risk associated with a particular investment.

```
NVDA_volatility <- runSD(NVDA_daily_returns, n = 20)
```

Plotting volatility

```
plot(NVDA_volatility, main = "20-Day Rolling Volatility of NVDA")
```



Technical indicators

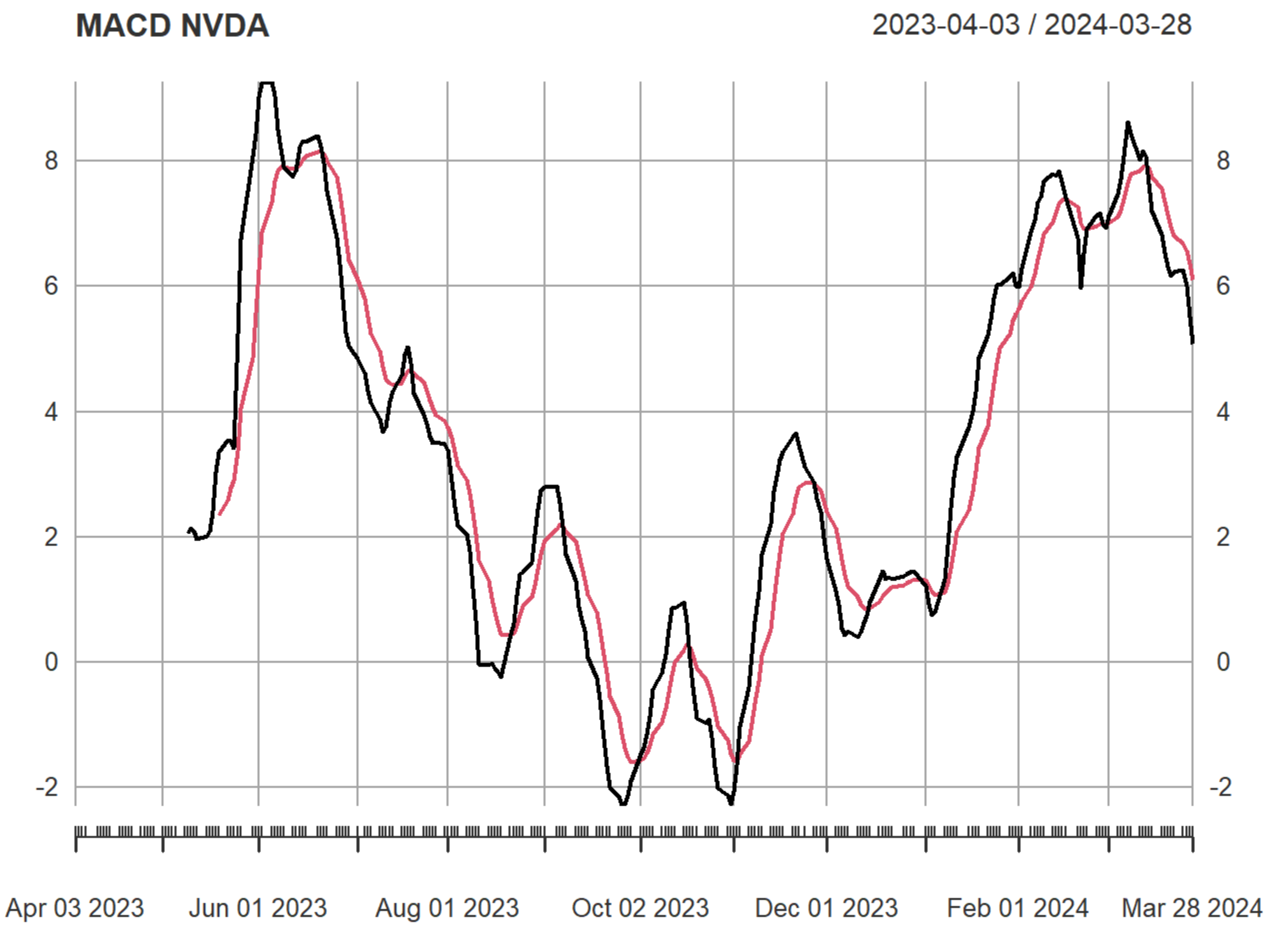
Moving Average Convergence/Divergence (MACD)

Relative Strength Index (RSI) calculates a ratio of the recent upward price movements to the absolute price movement.

```
NVDA_macd_vals <- MACD(Cl(NVDA))
NVDA_rsi_vals <- RSI(Cl(NVDA))
```

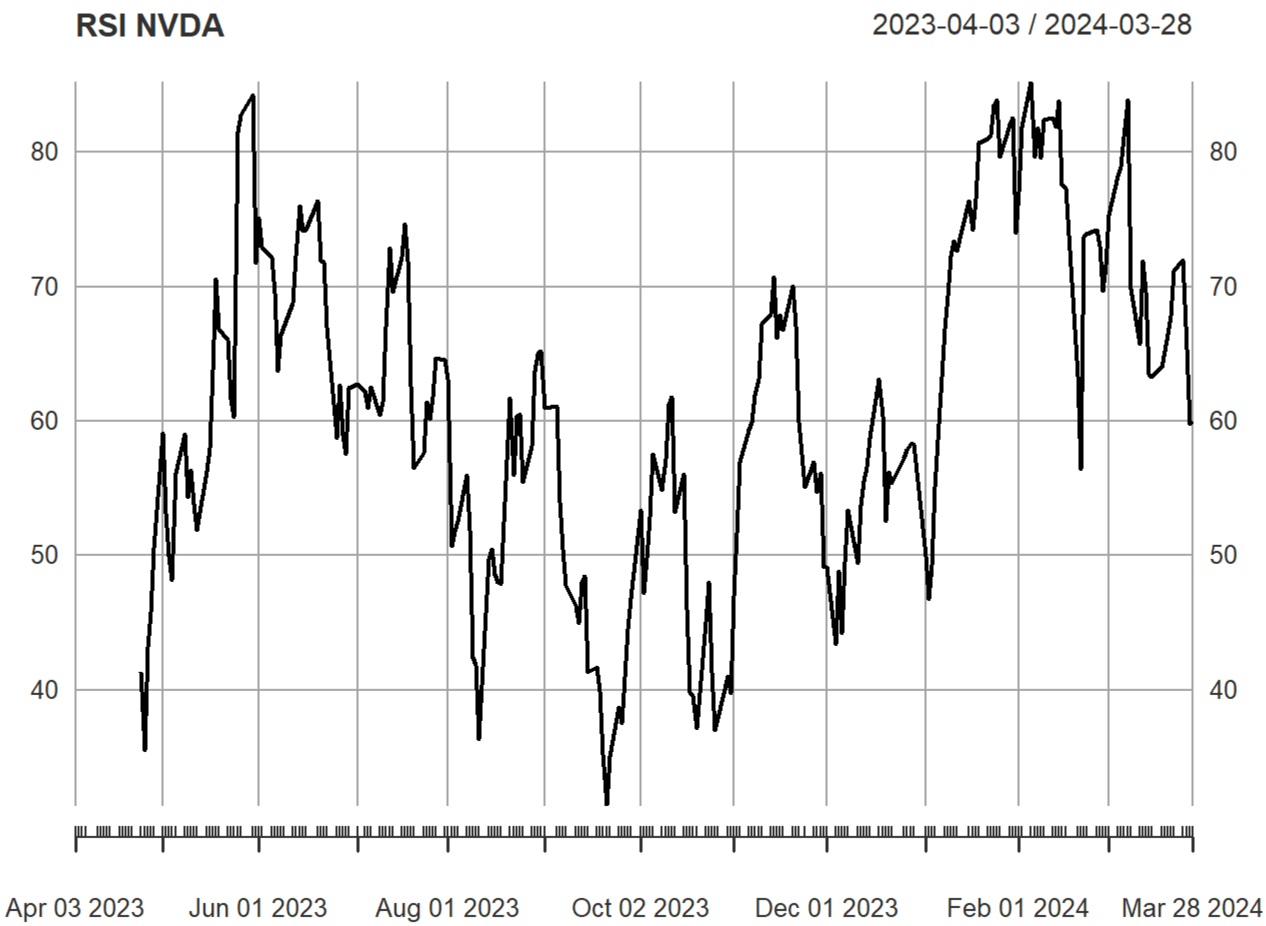
Plotting MACD of NVDA

```
plot(NVDA_macd_vals, main = "MACD NVDA")
```



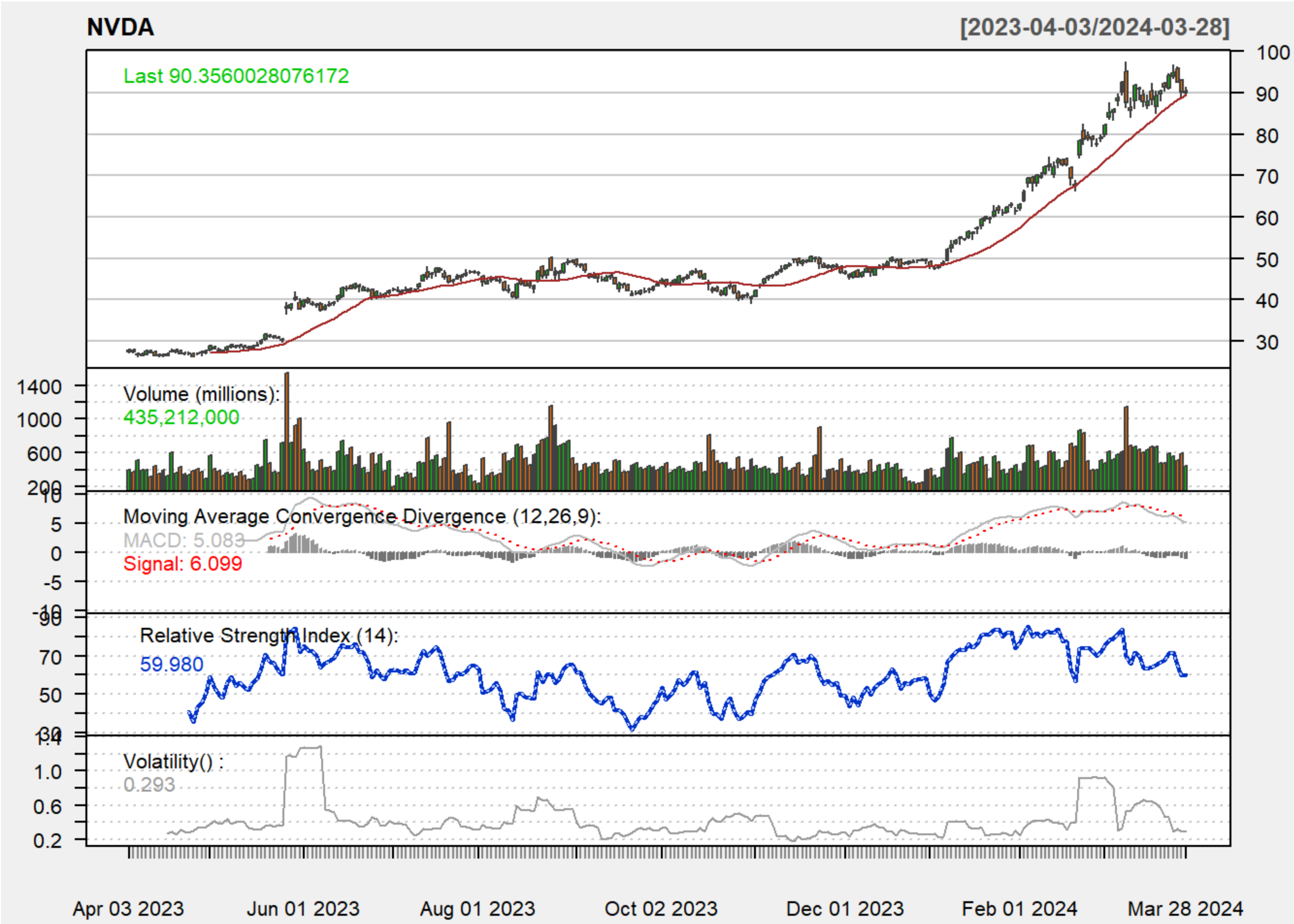
Plotting RSI of NVDA

```
plot(NVDA_rsi_vals, main = "RSI NVDA")
```



Plotting with technical indicators

```
chartSeries(NVDA, TA = "addVo();addMACD();addRSI();addSMA(20);addVolatility()",theme = chartThem
e("white"))
```



Part 4 - Build a simple ARIMA model to forecast next month's prices for NVIDIA and Evaluate the model's accuracy and discuss its implications.

Load necessary libraries

```
library(forecast)
```

Visualization without forecasting

```
plot(NVDA$NVDA.Open, main = "NVDA Open Price without forecasting")
```

Fit an ARIMA model

```
NVDA_OPfit <- auto.arima(NVDA$NVDA.Open)
```

Model parameters

```
summary(NVDA_OPfit)
```

```
## Series: NVDA$NVDA.Open
## ARIMA(0,1,1) with drift
##
## Coefficients:
##           ma1   drift
##       -0.1391  0.2541
## s.e.   0.0685  0.0908
##
## sigma^2 = 2.779:  log likelihood = -477.65
## AIC=961.3   AICc=961.4   BIC=971.84
##
## Training set error measures:
##                        ME     RMSE     MAE        MPE     MAPE      MASE
## Training set 9.373656e-05 1.657027 1.11468 -0.1575239 2.177063 0.9824078
##                     ACF1
## Training set 0.008703701
```

ARIMA(0,1,1)

We have an ARIMA model: ARIMA(0,1,1)

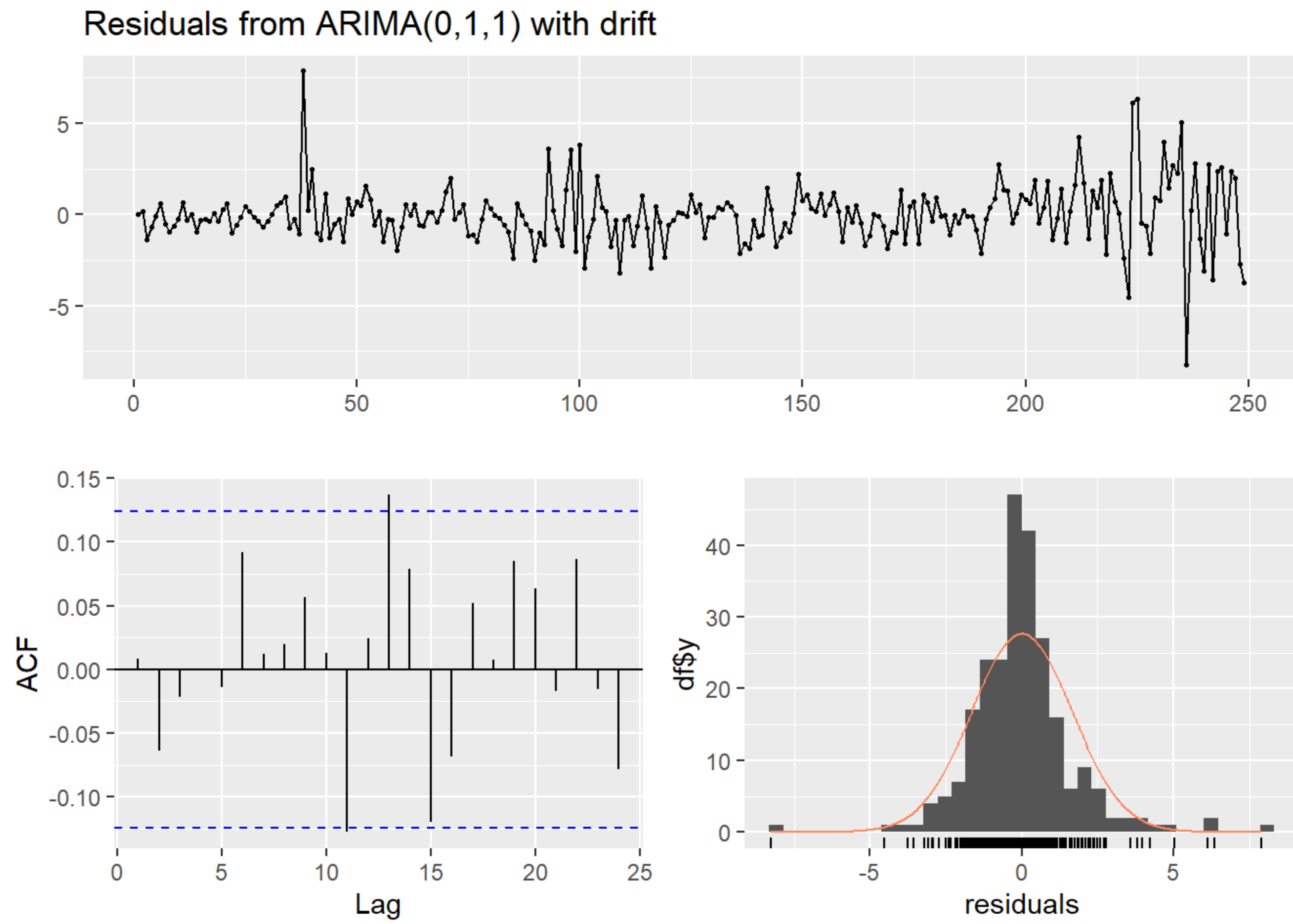
0 for the AR (autoregressive) part, indicating no lagged term of the variable was used.

1 for the I (integrated) part, meaning the data has been differenced once to achieve stationarity.

1 for the MA (moving average) part, indicating that the model uses one lagged forecast error term.

Check model diagnostics

```
checkresiduals(NVDA_OPfit)
```



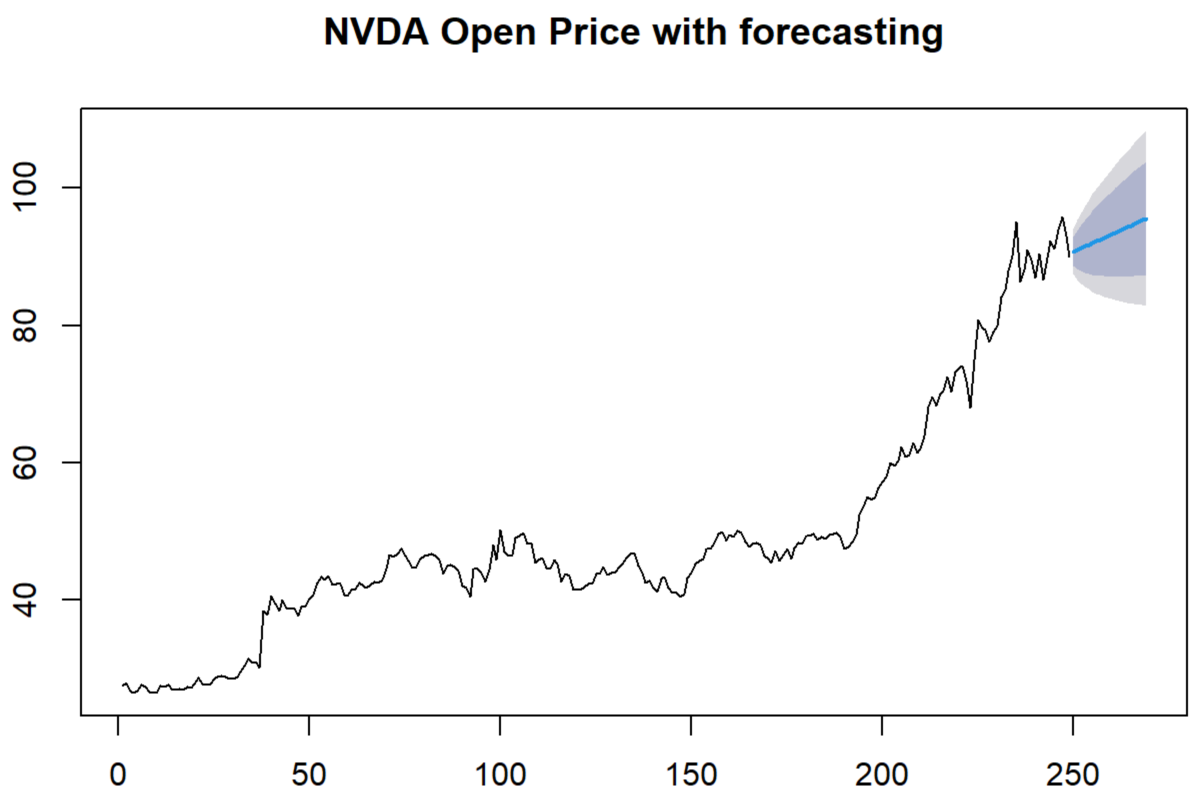

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,1) with drift
## Q* = 4.4063, df = 9, p-value = 0.8827
##
## Model df: 1.   Total lags used: 10
```

Create a forecast

```
NVDA_OP_future_values <- forecast(NVDA_OPfit, 20)  # Forecasting next 20 values
```

Plot the forecast with historical data, showing 80% & 95% prediction intervals obtained using exponential smoothing state space models

```
plot(NVDA_OP_future_values, main = "NVDA Open Price with forecasting")
```



Interpretation:

As we look at the forecast, we see that the trend is upward, indicating that the model predicts continued growth in NVIDIA's stock price. It's not a flat line, which would suggest uncertainty or no clear trend.

In this current forecast, the model is confident enough to predict an upward trend. This is shown within the 80% confidence interval, suggesting that while we are fairly certain about the growth, there is still a 20% chance that the actual prices could fall outside this range. The prices are expected to grow, but they could fluctuate within the values outlined by this interval.

On the other hand, the 95% confidence interval is wider, accounting for more variability. It implies that while the overall trend suggests little change, the stock price could

experience significant volatility within this interval. Essentially, 95% of the time, the price could swing wildly up or down but ultimately lead to a relatively stable trend over the forecast period.