AMD Structural Break and Volatility Analysis

Jinyan Yong

2025-04-22

## Step 1–3: Download Data and Compute Log Returns

getSymbols("AMD", src = "yahoo", from = "2023-01-31", to = "2025-01-31")

## [1] "AMD"

prices <- Cl(AMD)  
log\_returns <- diff(log(prices))  
log\_returns <- na.omit(log\_returns)

## Step 4–7: Create Dataset and Run Structural Break Regression

break\_date <- as.Date("2024-12-18")  
dates <- index(log\_returns)  
dummy <- ifelse(dates >= break\_date, 1, 0)  
lag\_r <- stats::lag(log\_returns, k = 1)  
interaction <- dummy \* lag\_r  
  
mydf <- data.frame(  
 Date = dates,  
 r = as.numeric(log\_returns),  
 lag\_r = as.numeric(lag\_r),  
 D = dummy,  
 interaction = as.numeric(interaction)  
)  
mydf <- na.omit(mydf)  
  
model <- lm(r ~ lag\_r + D + interaction, data = mydf)  
print(summary(model))

##   
## Call:  
## lm(formula = r ~ lag\_r + D + interaction, data = mydf)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.113034 -0.015976 -0.000879 0.017618 0.104950   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 0.0008272 0.0013485 0.613 0.540  
## lag\_r -0.0007177 0.0451246 -0.016 0.987  
## D -0.0019882 0.0057246 -0.347 0.729  
## interaction 0.2360176 0.2179932 1.083 0.279  
##   
## Residual standard error: 0.02928 on 496 degrees of freedom  
## Multiple R-squared: 0.002877, Adjusted R-squared: -0.003154   
## F-statistic: 0.477 on 3 and 496 DF, p-value: 0.6984

## Step 8: F-Test

restricted\_model <- lm(r ~ lag\_r, data = mydf)  
print(anova(restricted\_model, model))

## Analysis of Variance Table  
##   
## Model 1: r ~ lag\_r  
## Model 2: r ~ lag\_r + D + interaction  
## Res.Df RSS Df Sum of Sq F Pr(>F)  
## 1 498 0.42632   
## 2 496 0.42514 2 0.0011826 0.6898 0.5021

## Step 9: ADF Unit Root Tests

adf\_full <- ur.df(mydf$r, type = "drift", selectlags = "AIC")  
pre\_event <- mydf[mydf$D == 0, "r"]  
post\_event <- mydf[mydf$D == 1, "r"]  
adf\_pre <- ur.df(pre\_event, type = "drift", selectlags = "AIC")  
adf\_post <- ur.df(post\_event, type = "drift", selectlags = "AIC")  
  
print(summary(adf\_full))

##   
## ###############################################   
## # Augmented Dickey-Fuller Test Unit Root Test #   
## ###############################################   
##   
## Test regression drift   
##   
##   
## Call:  
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.113047 -0.015607 -0.001365 0.017585 0.105125   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.0006439 0.0013119 0.491 0.624   
## z.lag.1 -0.9936564 0.0635500 -15.636 <2e-16 \*\*\*  
## z.diff.lag -0.0050363 0.0448636 -0.112 0.911   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.02926 on 495 degrees of freedom  
## Multiple R-squared: 0.4997, Adjusted R-squared: 0.4977   
## F-statistic: 247.2 on 2 and 495 DF, p-value: < 2.2e-16  
##   
##   
## Value of test-statistic is: -15.6358 122.2405   
##   
## Critical values for test statistics:   
## 1pct 5pct 10pct  
## tau2 -3.44 -2.87 -2.57  
## phi1 6.47 4.61 3.79

print(summary(adf\_pre))

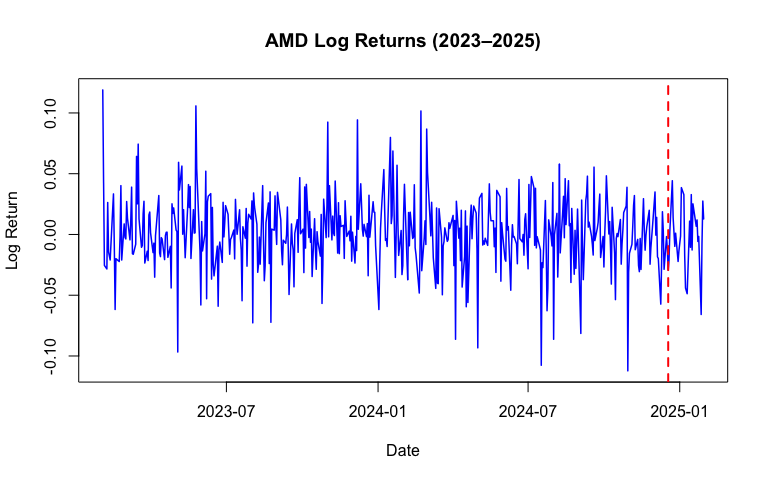
##   
## ###############################################   
## # Augmented Dickey-Fuller Test Unit Root Test #   
## ###############################################   
##   
## Test regression drift   
##   
##   
## Call:  
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.112903 -0.015397 -0.001421 0.017705 0.104986   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.0007917 0.0013590 0.583 0.560   
## z.lag.1 -0.9986469 0.0657518 -15.188 <2e-16 \*\*\*  
## z.diff.lag -0.0113810 0.0461467 -0.247 0.805   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.02944 on 467 degrees of freedom  
## Multiple R-squared: 0.5055, Adjusted R-squared: 0.5033   
## F-statistic: 238.7 on 2 and 467 DF, p-value: < 2.2e-16  
##   
##   
## Value of test-statistic is: -15.1881 115.3397   
##   
## Critical values for test statistics:   
## 1pct 5pct 10pct  
## tau2 -3.44 -2.87 -2.57  
## phi1 6.47 4.61 3.79

print(summary(adf\_post))

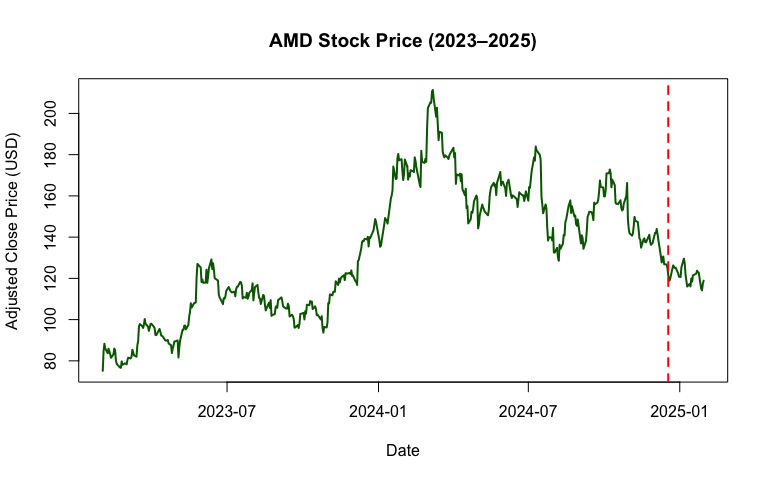
##   
## ###############################################   
## # Augmented Dickey-Fuller Test Unit Root Test #   
## ###############################################   
##   
## Test regression drift   
##   
##   
## Call:  
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.06628 -0.01414 0.00239 0.01486 0.03938   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.0004538 0.0052541 -0.086 0.93192   
## z.lag.1 -0.9756828 0.2554470 -3.820 0.00088 \*\*\*  
## z.diff.lag 0.2194142 0.2033884 1.079 0.29186   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.02655 on 23 degrees of freedom  
## Multiple R-squared: 0.4328, Adjusted R-squared: 0.3835   
## F-statistic: 8.775 on 2 and 23 DF, p-value: 0.001472  
##   
##   
## Value of test-statistic is: -3.8195 7.3602   
##   
## Critical values for test statistics:   
## 1pct 5pct 10pct  
## tau2 -3.58 -2.93 -2.60  
## phi1 7.06 4.86 3.94

## Step 10–11: Save Log Return and Price Plots

logret\_df <- data.frame(Date = index(log\_returns), Return = as.numeric(log\_returns))  
plot(logret\_df$Date, logret\_df$Return, type = "l",  
 main = "AMD Log Returns (2023–2025)",  
 xlab = "Date", ylab = "Log Return",  
 col = "blue", lwd = 1.5)  
abline(v = as.Date("2024-12-18"), col = "red", lty = 2, lwd = 2)



price\_df <- data.frame(Date = index(prices), Price = as.numeric(prices))  
plot(price\_df$Date, price\_df$Price, type = "l",  
 main = "AMD Stock Price (2023–2025)",  
 xlab = "Date", ylab = "Adjusted Close Price (USD)",  
 col = "darkgreen", lwd = 2)  
abline(v = as.Date("2024-12-18"), col = "red", lty = 2, lwd = 2)



## Step 12: Volatility Structural Break Model

squared\_r <- mydf$r^2  
lag\_squared\_r <- stats::lag(squared\_r, k = 1)  
interaction\_vol <- mydf$D \* lag\_squared\_r  
  
vol\_data <- data.frame(  
 r2 = squared\_r,  
 lag\_r2 = lag\_squared\_r,  
 D = mydf$D,  
 interaction = interaction\_vol  
)  
vol\_data <- na.omit(vol\_data)  
  
vol\_model <- lm(r2 ~ lag\_r2 + D + interaction, data = vol\_data)  
print(summary(vol\_model))

## Warning in summary.lm(vol\_model): essentially perfect fit: summary may be  
## unreliable

##   
## Call:  
## lm(formula = r2 ~ lag\_r2 + D + interaction, data = vol\_data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.041e-18 -7.210e-20 -4.730e-20 -2.130e-20 2.297e-17   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 4.736e-19 5.369e-20 8.821e+00 <2e-16 \*\*\*  
## lag\_r2 1.000e+00 2.858e-17 3.498e+16 <2e-16 \*\*\*  
## D -5.101e-20 2.455e-19 -2.080e-01 0.835   
## interaction -8.666e-18 2.049e-16 -4.200e-02 0.966   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.036e-18 on 496 degrees of freedom  
## Multiple R-squared: 1, Adjusted R-squared: 1   
## F-statistic: 4.163e+32 on 3 and 496 DF, p-value: < 2.2e-16

## Step 13: Summary Statistics

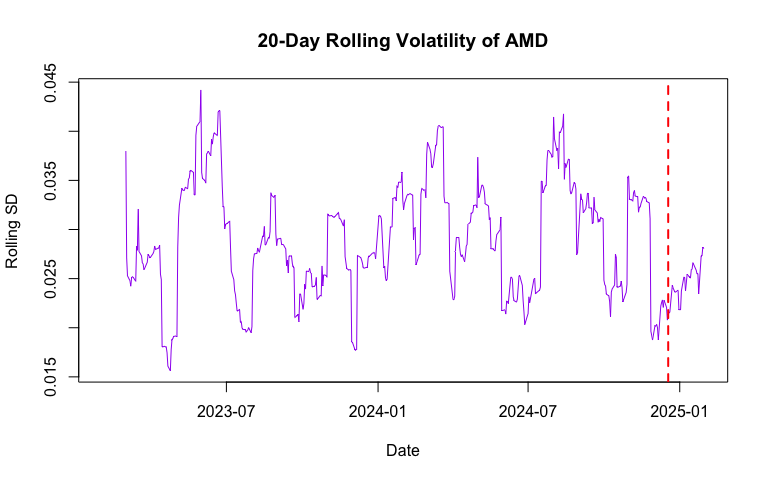
summary\_stats <- data.frame(  
 Mean = c(mean(pre\_event), mean(post\_event)),  
 SD = c(sd(pre\_event), sd(post\_event)),  
 Skewness = c(skewness(pre\_event), skewness(post\_event)),  
 Kurtosis = c(kurtosis(pre\_event), kurtosis(post\_event))  
)  
rownames(summary\_stats) <- c("Pre-event", "Post-event")  
knitr::kable(summary\_stats, caption = "Summary Statistics: Pre vs Post Event")

Summary Statistics: Pre vs Post Event

|  | Mean | SD | Skewness | Kurtosis |
| --- | --- | --- | --- | --- |
| Pre-event | 0.0008264 | 0.0294042 | -0.1523071 | 4.752419 |
| Post-event | -0.0018045 | 0.0264925 | -0.3724834 | 2.909868 |

## Step 14: Rolling Volatility Plot

rolling\_sd <- rollapply(log\_returns, width = 20, FUN = sd, align = "right", fill = NA)  
rolling\_df <- data.frame(Date = index(rolling\_sd), Vol = as.numeric(rolling\_sd))  
plot(rolling\_df$Date, rolling\_df$Vol, type = "l", col = "purple",  
 main = "20-Day Rolling Volatility of AMD",  
 xlab = "Date", ylab = "Rolling SD")  
abline(v = as.Date("2024-12-18"), col = "red", lty = 2, lwd = 2)



## Step 15: Event Window Plot

event\_center <- as.Date("2024-12-18")  
window\_size <- 5  
event\_idx <- which(index(log\_returns) == event\_center)  
event\_window\_idx <- (event\_idx - window\_size):(event\_idx + window\_size)  
event\_window\_idx <- event\_window\_idx[event\_window\_idx > 0 & event\_window\_idx <= length(log\_returns)]  
event\_window\_returns <- log\_returns[event\_window\_idx]  
plot(index(event\_window\_returns), coredata(event\_window\_returns), type = "l",  
 main = "Log Returns Around Fed Announcement (±5 Days)",  
 col = "orange", lwd = 2,  
 xlab = "Date", ylab = "Log Return")  
abline(v = event\_center, col = "red", lty = 2, lwd = 2)

