Testing the Weak-Form Efficient Market Hypothesis: Evidence from Bitcoin Daily Prices

Joe White

Miami University

December 10, 2025

Introduction

The Efficient Market Hypothesis (EMH) is one of the most influential ideas in modern financial economics, and it provides the foundation for understanding how asset prices incorporate information. Under the weak form of EMH, historical price data and past returns should not help investors predict future price movements. If markets are efficient in this sense, return patterns such as momentum, reversals, or seasonality should not provide a reliable way to earn above-average risk-adjusted returns.

This project examines whether Bitcoin's daily price behavior is consistent with these predictions by testing for return predictability, autocorrelation, and randomness in the time-series data. Before turning to the empirical analysis, I provide a brief literature review of Burton Malkiel's discussion of EMH and its critics, which highlights the core arguments for and against return predictability in financial markets.

Literature Review

Malkiel's article provides an overview of the Efficient Market Hypothesis, its development, and the main empirical challenges raised by critics. He begins by explaining the central idea behind EMH: financial markets rapidly incorporate new information into prices, making it extremely difficult for investors to consistently outperform the market using either technical analysis (trading based on past prices and charts) or fundamental analysis (trading based on financial statements and valuation ratios). If prices fully reflect available

information, then they should follow what is known as a "random walk," meaning that future price changes depend only on new, unpredictable information rather than on past price patterns.

Malkiel reviews a wide range of empirical studies that appear to challenge the randomness of returns. These include short-term momentum effects, where prices continue moving in the same direction for brief periods, and long-term reversals, where stocks that perform very poorly eventually outperform. He also discusses valuation-based predictability, such as dividend yields or price–earnings ratios forecasting long-run returns, as well as well-known market "anomalies" like the January effect or day-of-the-week patterns. Behavioral economists argue that psychological factors—overconfidence, herd behavior, or underreaction to news—can create predictable return patterns inconsistent with strict market efficiency.

However, Malkiel emphasizes that even when predictable patterns are documented statistically, they rarely translate into profitable trading strategies. Any predictable pattern is usually too small to survive real-world trading costs, and once an anomaly becomes widely known, it tends to shrink or disappear as investors try to exploit it. Many patterns also arise from data mining, survivorship bias, or specific modeling choices. Importantly, he notes that professional portfolio managers—those best positioned to exploit inefficiencies—consistently fail to outperform passive index funds after adjusting for risk, which supports the idea that markets remain broadly efficient.

Malkiel concludes that while markets are not perfectly rational and occasionally display bubbles or mispricing, these episodes do not provide systematic opportunities for earning abnormal returns. In this sense, the weak-form EMH remains largely intact: even if some return predictability exists in theory, it is not strong, stable, or exploitable enough to undermine market efficiency. This perspective directly informs the empirical analysis in this project. If EMH holds for Bitcoin, its daily returns should behave much like noise—showing little autocorrelation and no stable patterns that could be used to forecast future price movements.

Data and Statistical Analysis (Sections 1 and 2)

For the empirical analysis, I use daily U.S. dollar prices for Bitcoin from the Federal Reserve Bank of St. Louis (FRED) database. The specific series is the closing price of Bitcoin in U.S. dollars with the code `CBBTCUSD`. I download the full available sample of daily observations and load it into `R` using the `quantmod` package. The raw price series begins in December 2014 and extends through the most recent available trading day at the time of the analysis.

To study the behavior of returns, I transform the price series into daily log returns. The return on day t is computed as

$$r_t = \log(P_t) - \log(P_{t-1}),$$

where P_t is the Bitcoin price on day t. This transformation has two advantages: it scales returns in percentage terms and makes them easier to compare across time when the level of the price changes dramatically.

Figure 1 shows the time series of the Bitcoin price and the corresponding daily log return. The price plot highlights large swings over time, including long periods of rapid increase and sharp crashes. In contrast, the daily returns fluctuate tightly around zero and alternate frequently between positive and negative values, which is visually consistent with a noisy process.

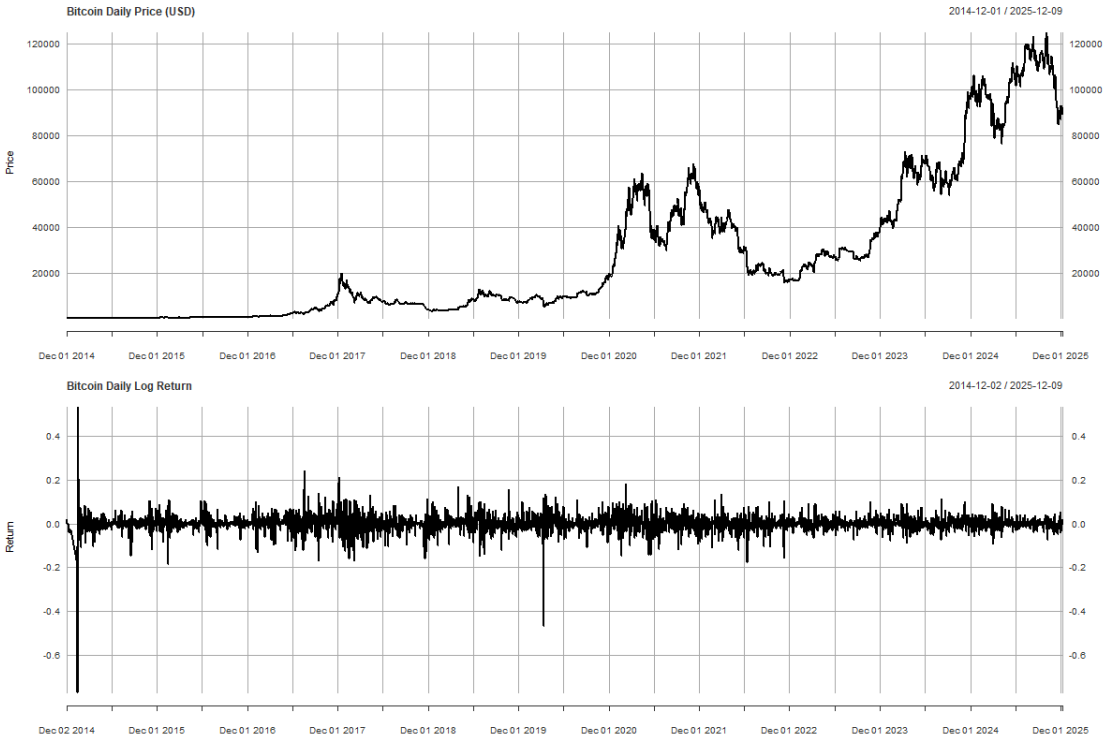

Figure 1: Bitcoin daily price (top) and daily log return (bottom).

Table 1 reports descriptive statistics for the price and daily log return. The average Bitcoin price over the sample is about $27,241, with a very large standard deviation of roughly $31,562. The minimum observed price is $120, and the maximum exceeds $124,000, reflecting the extreme growth and volatility of Bitcoin over the past decade.

The behavior of returns is quite different. The mean daily log return is about 0.0014, which corresponds

to roughly 0.14% per day. The standard deviation of returns is about 0.0386, or 3.86% per day, indicating substantial day-to-day volatility. The minimum daily return in the sample is approximately -0.77, and the maximum is about 0.53, meaning that on some days the price fell by more than 70% or rose by more than 50%. These large swings reflect the speculative and risky nature of Bitcoin trading.

	mean	sd	min	q25.25%	median	q75.75%	max
Price	27240.5659	31561.5234	120.0000	3624.7425	11199.0250	42497.4575	124720.0900
Return	0.0014	0.0386	-0.7732	-0.0124	0.0011	0.0161	0.5317

Table 1: Descriptive Statistics for Bitcoin Price and Daily Log Return

To formally evaluate whether the price and return series are stationary, I conduct Augmented Dickey–Fuller (ADF) unit root tests. For the price series, the ADF test statistic is approximately -1.96 with a p-value of about 0.59. This means we cannot reject the hypothesis that the Bitcoin price contains a unit root and is non-stationary. In plain terms, the price level tends to wander over time rather than fluctuating around a stable long-run mean.

For the daily returns, the ADF test statistic is approximately -45.6 with a p-value smaller than 0.01. In this case, we strongly reject the presence of a unit root and conclude that the return series is stationary. This result is consistent with the weak-form EMH: even though the price itself drifts and trends over time, the day-to-day returns behave more like a stable, mean-reverting noise process.

Rolling AR(1) Analysis (Section 3)

To investigate short-run predictability in Bitcoin returns, I estimate a simple autoregressive model of order one, AR(1), on rolling windows of the data. The model is

$$r_t = \phi_0 + \phi_1 r_{t-1} + \varepsilon_t,$$

where r_t is the daily log return on day t, r_{t-1} is the previous day's return, and ε_t is a random error term. The key parameter of interest is ϕ_1, which measures whether today's return tends to move in the same direction as yesterday's return. Under the weak-form EMH, ϕ_1 should be close to zero, meaning that knowing yesterday's return does not help predict today's return.

Rather than estimating this model once for the full sample, I use rolling windows of 28 consecutive trading days. For each window, I run the AR(1) regression and record the t-statistic for ϕ_1. This produces a

time series of t-values that shows how the evidence for or against predictability changes over time.

Figure 2 plots the t-statistic for ϕ_1 across rolling windows, along with horizontal lines at -1.96 and $+1.96$. These lines correspond to the 5% significance cutoff in a standard normal distribution. When the t-statistic falls between -1.96 and $+1.96$, we cannot reject the hypothesis that $\phi_1 = 0$ at the 5% level. When the t-statistic lies outside this band, the lagged return appears significantly predictive within that window.

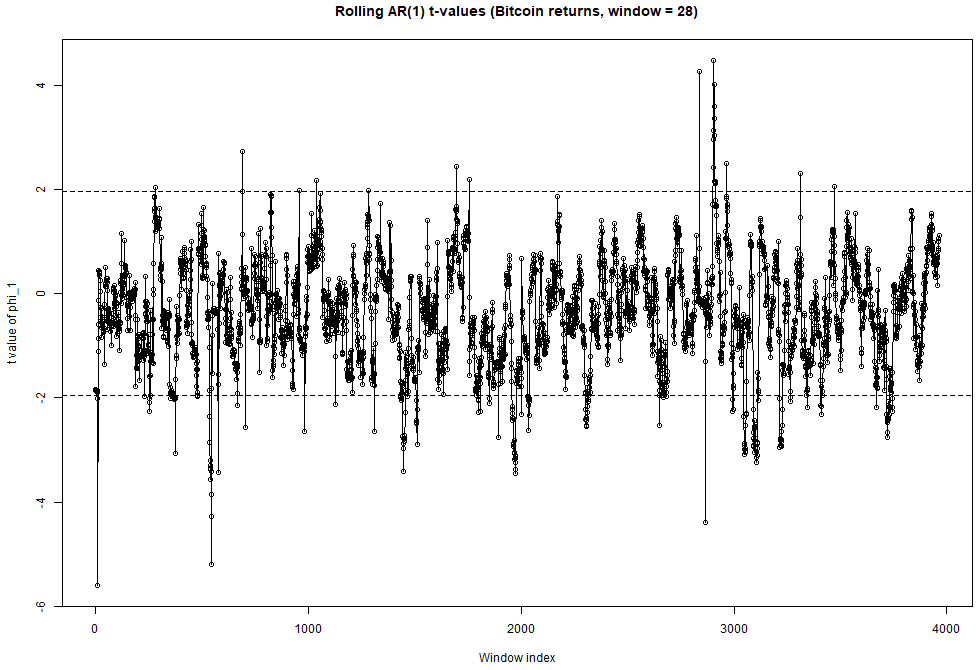

Figure 2: Rolling 28-day AR(1) t-statistics for the lagged return coefficient.

Overall, most of the t-statistics cluster around zero and remain inside the $[-1.96, 1.96]$ band. There are some windows where the t-values temporarily move outside the band, indicating short periods where lagged returns appear statistically significant. However, these episodes are sporadic and do not persist for long stretches of time.

From a non-specialist perspective, the main message is that Bitcoin returns do not show strong or stable predictability from one day to the next. Any apparent predictability that does appear is localized to short windows and could easily be the result of random fluctuations rather than a reliable trading rule. This pattern is broadly consistent with the weak-form EMH, which allows for occasional deviations from randomness but rules out a stable, exploitable relationship between past and future returns.

Rolling Box–Ljung Tests (Section 4)

As a second approach to testing for return predictability, I apply the Box–Ljung test for autocorrelation to rolling windows of the daily returns. For each 28-day window, I test whether the returns are uncorrelated at lag 1. The null hypothesis of the Box–Ljung test is that there is no autocorrelation; a low p-value (below 0.05) indicates evidence of autocorrelation, while a high p-value suggests that the returns look like uncorrelated noise.

Using the same rolling windows as in the AR(1) analysis, I compute the Box–Ljung p-value for each window and plot the resulting time series in Figure 3. The horizontal line at 0.05 marks the usual 5% significance level. Windows with p-values below this line reject the null hypothesis of no autocorrelation; those above the line do not.

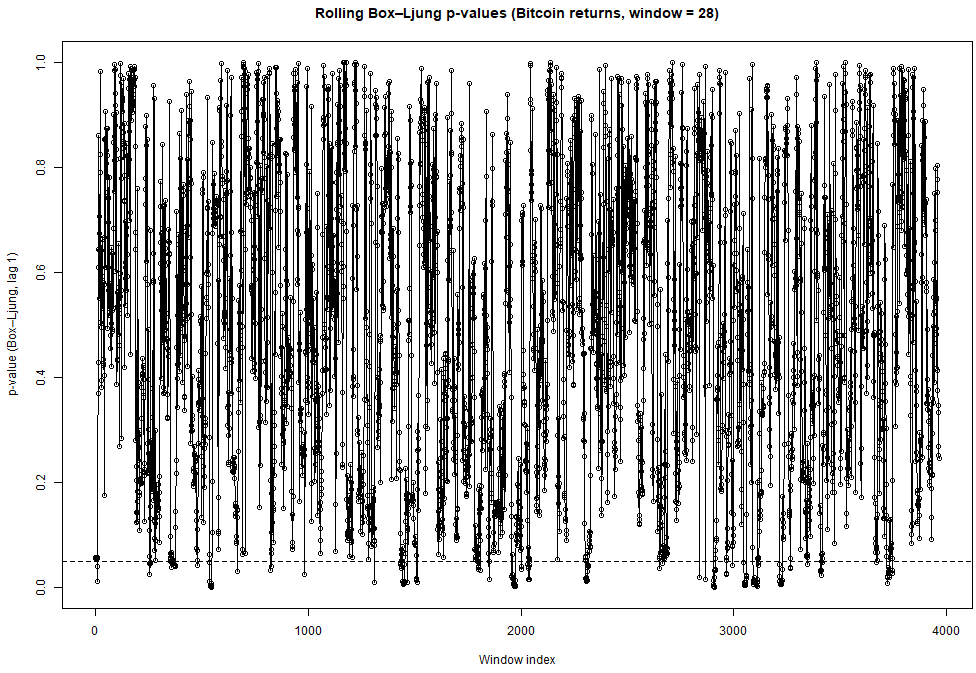

Figure 3: Rolling 28-day Box–Ljung p-values for daily Bitcoin returns (lag 1).

Most of the time, the p-values lie above 0.05, meaning that we do not find strong evidence of autocorrelation in the returns. There are some windows where the p-values dip below 0.05, indicating short periods where returns show statistically significant dependence over time. However, just as in the AR(1) analysis, these departures from the null are isolated rather than persistent.

For a non-specialist reader, the takeaway is that the Box–Ljung test mostly sees Bitcoin returns as behav-

ing like random noise, with only occasional episodes of detectable autocorrelation. This again lines up with the weak-form EMH: there is no consistent pattern in the data that would allow an investor to build a simple "look at yesterday's return" trading strategy and earn reliably superior profits.

Conclusion

This project uses daily Bitcoin price data from FRED to test the weak-form Efficient Market Hypothesis. After converting prices to daily log returns, I find that the price series is non-stationary, while the return series is stationary, which is consistent with standard financial theory. Descriptive statistics show that Bitcoin is extremely volatile, with large swings in both prices and daily returns.

Rolling AR(1) regressions and Box–Ljung tests provide a more detailed look at short-run predictability. Both approaches suggest that while there are occasional short periods with statistically significant autocorrelation, these episodes are not long-lasting and do not form a stable pattern over time. Most windows show no strong evidence that yesterday's return helps to predict today's return.

Taken together, the empirical results for Bitcoin daily returns are broadly consistent with the weak-form EMH. The return series behaves much like a noisy, mean-reverting process with limited and unstable predictability. Although Bitcoin is an unconventional and highly volatile asset, its daily return behavior does not strongly contradict the idea that past price information alone cannot be used to systematically "beat the market."

Appendix: R Code for Part II

The R code used to download the data, compute returns, produce the plots, and generate the LaTeX table for descriptive statistics is included below.

```
############################################################
## ECO 420Y Project - Testing EMH for Bitcoin (Part II)
## R code for Sections -14
############################################################

## Packages
library(quantmod)
```

```r
library(tseries)
library(xtable)

## Paths
base_dir <- "C:/Repositories/eco-420y-financial-economics/efficient_market_
    hypothesis"
plot_dir <- file.path(base_dir, "plots")

dir.create(base_dir, showWarnings = FALSE, recursive = TRUE)
dir.create(plot_dir, showWarnings = FALSE, recursive = TRUE)

################################################################
## Section 1: Download Bitcoin price data from FRED
################################################################

sym <- "CBBTCUSD"
getSymbols(sym, src = "FRED")

btc_price_xts <- get(sym)
colnames(btc_price_xts) <- "Price"
btc_price_xts <- na.omit(btc_price_xts)

head(btc_price_xts)

################################################################
## Section 2: Statistical analysis
## - daily returns
## - plots
## - descriptive statistics
## - unit root tests
################################################################

## Daily log returns
```

```r
btc_ret_xts <- diff(log(btc_price_xts$Price))
btc_ret_xts <- na.omit(btc_ret_xts)
colnames(btc_ret_xts) <- "Return"

btc_price <- as.numeric(btc_price_xts$Price)
btc_ret   <- as.numeric(btc_ret_xts$Return)

## Plot price and return (saved to file)
png(file = file.path(plot_dir, "bitcoin_price_return.png"),
    width = 1200, height = 800)
par(mfrow = c(2, 1))

plot(btc_price_xts,
     main = "Bitcoin Daily Price (USD)",
     ylab = "Price",
     xlab = "Date")

plot(btc_ret_xts,
     main = "Bitcoin Daily Log Return",
     ylab = "Return",
     xlab = "Date")

par(mfrow = c(1, 1))
dev.off()

## Descriptive statistics
descriptive_stats <- function(x) {
  x <- x[is.finite(x)]
  c(
    mean   = mean(x),
    sd     = sd(x),
    min    = min(x),
    q25    = quantile(x, 0.25),
```

```
    median = median(x),
    q75    = quantile(x, 0.75),
    max    = max(x)
  )
}

price_stats  <- descriptive_stats(btc_price)
return_stats <- descriptive_stats(btc_ret)

stats_table <- rbind(Price = price_stats,
                     Return = return_stats)
round(stats_table, 4)

## LaTeX table for descriptive statistics
latex_tab <- xtable(
  stats_table,
  caption = "Descriptive Statistics for Bitcoin Price and Daily Log Return",
  label   = "tab:btc_stats",
  digits  = 4
)

tex_path <- file.path(base_dir, "btc_descriptive_stats.tex")

# Capture the LaTeX table as a character vector
latex_lines <- capture.output(
  print(latex_tab, type = "latex", include.rownames = TRUE)
)

# Write those lines to the .tex file (same style as your TWFE tables)
writeLines(latex_lines, tex_path, useBytes = TRUE)


## ADF tests for price and return
```

```r
adf_price  <- adf.test(btc_price, alternative = "stationary", k = 1)
adf_price

adf_return <- adf.test(btc_ret, alternative = "stationary", k = 1)
adf_return

################################################################
## Section 3: Rolling AR(1) t-values (window size = 28)
## y_t = phi_0 + phi_1 y_{t-1} + error_t
################################################################

r      <- btc_ret
n      <- length(r)
rlag1 <- c(NA, r[1:(n - 1)])

d <- data.frame(r = r, rlag1 = rlag1)
d <- na.omit(d)
d$tr <- 1:nrow(d)

window <- 28
Td     <- nrow(d)

t_values <- numeric(Td - window + 1)

j <- 1
while (j + window - 1 <= Td) {
  subd <- subset(d, tr >= j & tr <= j + window - 1)
  m    <- lm(r ~ rlag1, data = subd)
  t_values[j] <- summary(m)$coef["rlag1", "t value"]
  j <- j + 1
}

## Save rolling AR(1) t-values plot
```

```r
png(file = file.path(plot_dir, "bitcoin_rolling_tvalues.png"),
    width = 1000, height = 700)

plot(t_values, type = "o",
     xlab = "Window index",
     ylab = "t value of phi_1",
     main = "Rolling AR(1) t-values (Bitcoin returns, window = 28)")
abline(h = -1.96, lty = 2)
abline(h =  1.96, lty = 2)

dev.off()

################################################################
## Section 4: Rolling -BoxLjung p-values (window size = 28)
################################################################

r_full   <- btc_ret
n_full   <- length(r_full)
n_win    <- n_full - window + 1
p_values <- numeric(n_win)

j <- 1
while (j + window - 1 <= n_full) {
  subr <- r_full[j:(j + window - 1)]
  bj   <- Box.test(subr, lag = 1, type = "Ljung-Box")
  p_values[j] <- bj$p.value
  j <- 1 + j
}

## Save rolling -BoxLjung p-values plot
png(file = file.path(plot_dir, "bitcoin_rolling_pvalues.png"),
    width = 1000, height = 700)

```

```r
plot(p_values, type = "o",
     xlab = "Window index",
     ylab = "p-value (-BoxLjung, lag 1)",
     main = "Rolling -BoxLjung p-values (Bitcoin returns, window = 28)",
     ylim = c(0, 1))
abline(h = 0.05, lty = 2)

dev.off()
```