Empirical Finance: Assignment 5

By Group 15

Lanlan Hou	2801069
Rick van der Ploeg	2782774
Sebastiaan van Dijen	2769505
Wa Chak Sou	2796840

School of Business and Economics

1st December 2025

QUESTION 1: Forecasting Value-at-Risk

a. AR(1)-GARCH(1,1) model estimation

For the final five years of the sample, we generate daily return and volatility forecasts using an AR(1)-GARCH(1,1) model. The model is re-estimated every 20 trading days using a rolling estimation window of sufficiently large size to ensure stable parameter estimation. This setup mimics how a risk manager would regularly update the model to incorporate new information while allowing the parameters to adjust to changing market conditions.

The conditional mean and variance dynamics of the model can be written as

$$r_t = \mu + \phi r_{t-1} + \varepsilon_t, \qquad \varepsilon_t = \sigma_t z_t, \qquad z_t \sim i.i.d.\ (0,1),$$

$$\sigma_t^2 = \omega + \alpha \varepsilon_{t-1}^2 + \beta \sigma_{t-1}^2,$$

where $\omega > 0$, $\alpha \geq 0$, $\beta \geq 0$, and typically $\alpha + \beta$ is close to one, indicating high persistence in volatility.

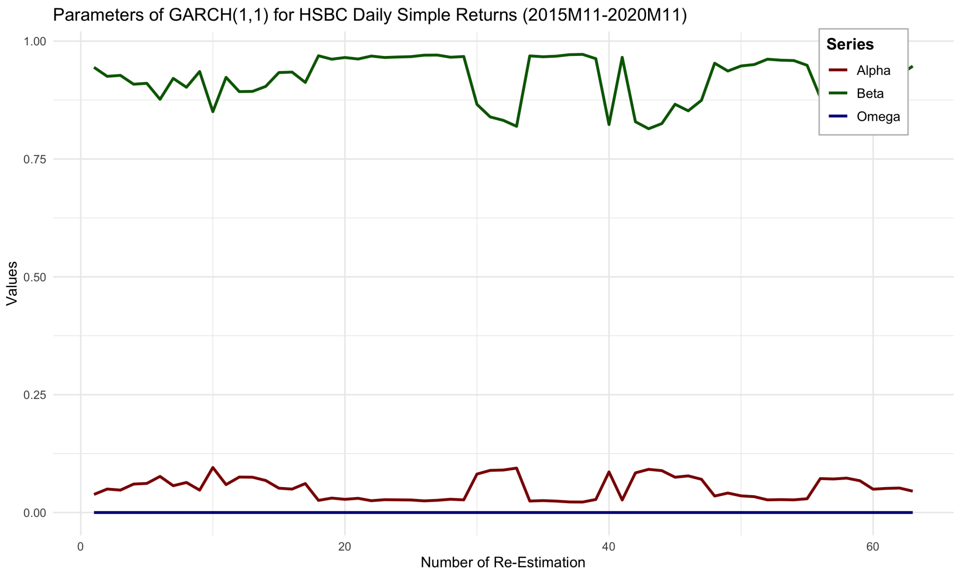

Figure 1.1.1 The estimated GARCH parameters across all re-estimations.

Figure 1.1.1 shows the evolution of the GARCH parameters across all rolling re-estimations. We observe that the parameters are not constant over time, which indicates that the volatility dynamics of the series change with market conditions. During periods of higher market turbulence, the short-run parameter α tends to increase, reflecting a stronger immediate impact of recent shocks on volatility. In calmer periods, the persistence parameter β tends to be relatively larger, implying a slower decay of volatility and smoother volatility dynamics. The constant term ω remains very close to zero all re-estimation. It indicates that the unconditional long-run component of volatility is negligible relative to past squared shocks and past volatility .

Overall, the sum $\alpha + \beta$ remains close to one throughout most of the sample. Movements in α are often accompanied by opposite movements in β. This is consistent with the requirement that the model remains stationary and confirms that volatility is highly persistent, which is usual for financial return series. The time variation in the parameters illustrates how the model adapts to different regimes in the data and highlights the importance of regularly updating the GARCH model when it is used for risk management purposes.

b. Value-at-Risk (VaR) for different volatility models

For the second five-year sample, we compute one day ahead VaR(0.05) forecasts using three different volatility models:

(i) The AR(1)–GARCH(1,1) model estimated in Question 1(a).

(ii) An EWMA model with decay parameter $\lambda = 0.94$, where the initial mean and variance are computed using the first 30 observations.

(iii) A Historical Simulation (HS) approach with a rolling window of $W = 1000$ daily returns.

Under each model, the one day ahead VaR(0.05) is obtained as the 5% lower quantile of the forecast return distribution. For the parametric models (AR–GARCH and EWMA), this is based on the conditional mean and variance together with a normality assumption for the standardized innovations. For Historical Simulation, the VaR is taken as the empirical 5% quantile of the returns in the rolling window.

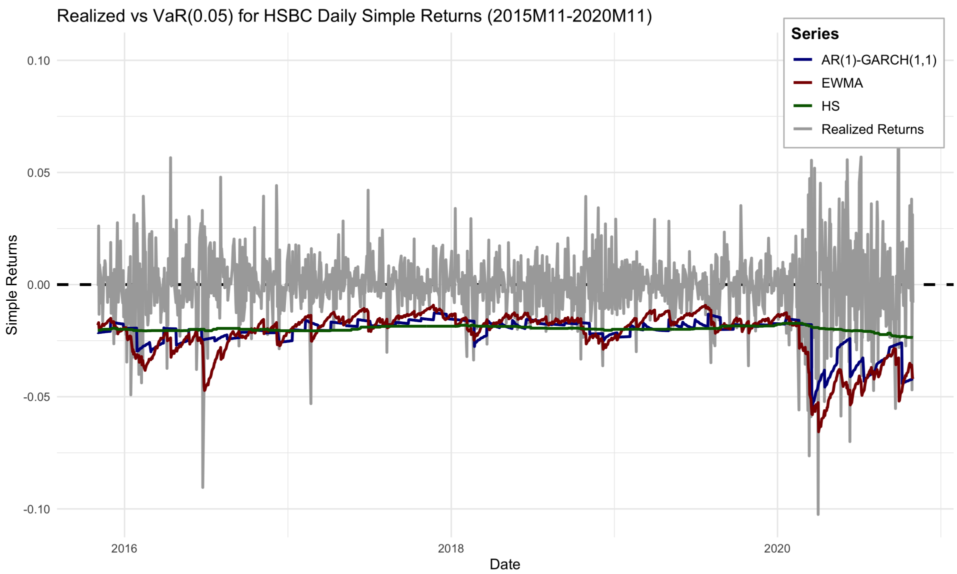

Figure 1.2.1 The VaR(0.05) series of different models against the realized returns.

Figure 1.2.1 plots the three VaR(0.05) series together with the realized returns. The AR–GARCH-based VaR series is relatively smooth and persistent, as the GARCH model captures volatility clustering through its lagged variance and squared residual terms. The EWMA VaR reacts more quickly to new large shocks, because recent observations receive more weight in the volatility estimate. The Historical Simulation VaR is the most irregular of the three, since it depends directly on the extreme returns contained in the rolling sample window.

Across all models, VaR levels increase during periods of higher market volatility and decrease in calmer periods, which is consistent with economic intuition. However, the magnitude and timing of these adjustments differ. This highlights that the choice of volatility model can have a substantial impact on the reported risk figures and therefor on risk management decisions.

Question 2: Backtesting Value-at-Risk

a. Violation Ratio and Unconditional coverage test

i. Violation Ratio

The violation ratio (VR) measures whether a VaR model systematically under- or over-predicts tail risk. It is defined as

$$\mathrm{VR} = \frac{\text{Observed number of VaR violations}}{\text{Expected number of violations}} = \frac{v_1}{p \times W_T}, \tag{1}$$

where $v_1 = \sum_{t=1}^{W_T} \eta_t$ denotes the total number of violations in the testing window of length W_T, and

$$\eta_t = \begin{cases} 1, & \text{if } y_t \leq -\mathrm{VaR}_t(p), \\ 0, & \text{if } y_t > -\mathrm{VaR}_t(p), \end{cases}$$

indicates whether a violation occurred at time t.

A violation ratio of $\mathrm{VR} = 1$ is the theoretical benchmark: the model produces exactly as many violations as expected under the nominal VaR level p.

- If $\mathrm{VR} > 1$, the model **underestimates risk**: more violations occur than expected.
- If $\mathrm{VR} < 1$, the model **overestimates risk**: fewer violations occur than expected.

A rule of thumb for deviation from 1:

- $\mathrm{VR} \in [0.8, 1.2]$: **good** forecast accuracy.
- $\mathrm{VR} < 0.5$ or $\mathrm{VR} > 1.5$: model is **imprecise**.

These ranges tighten as the testing window W_T increases, since sampling uncertainty decreases with more observations.

Model	**AR-GARCH**	**EWMA**	**HS**
Violation Ratio	1.191422	1.159651	1.461477

Table 2.1.1: Violation Ratio for Different VaR Models

Table 2.1.1 reveals the violation ratio for the three VaR(0.05) models. Both violation ratios of AR(1)-GARCH(1,1) and EWMA models are slightly over the expected ratio ($\mathrm{VR} = 1$) and are within the range of [0.8, 1.2]. These results suggest that there are slightly more violations than expected and both VaR models are slightly underestimates risk while the results are still acceptable. In contrast, the violation ratio of the Historic Simulation model (Non-parametric VaR) is much higher than expected. It indicates that the evaluation of VaR using Historic Simulation is incorrect, much more violations occur than expected as it underestimates risk.

ii. Unconditional Coverage (Bernoulli Coverage) Test

An Unconditional Coverage (Bernoulli Coverage) Test can be used to test whether theoretical violation probability p matches the empirically observed $\tilde{p}$. For VaR(0.05), a violation for 5% of the VaR(0.05) forecasts is expected to be observed:

- If $\hat{p} > p \Rightarrow$ Systematically underestimate risk.
- If $\hat{p} > p \Rightarrow$ Systematically overestimate risk.
- If $\hat{p} = p \Rightarrow$ Accurately capture risk.

To test whether $\hat{p} = p$, assume that each violation is an *i.i.d.* draw from a Bernoulli distribution. Notice that η_t takes on the value 1 with probability p and zero with probability 1-p. η_t is a sequence of Bernoulli-distributed random variables. Then, use the Bernoulli coverage test to ascertain the proportion of violations.

The null and alternative hypotheses for VaR violations are formulated as follows:

$$H_0 : \eta \sim B(p) \quad \text{vs.} \quad H_1 : \eta \sim B(\tilde{p}) \quad \text{where} \quad \tilde{p} \neq p$$

where B stands for the Bernoulli distribution. The test statistic can be computed as:

$$\text{LR} = 2(\log\mathcal{L}_U(\tilde{p}) - \log\mathcal{L}_R(p)) \sim \chi^2_{(1)}$$

Model	**AR-GARCH**	**EWMA**	**HS**
Test Statistic	2.293914	1.6100116	12.42976
p-value	0.129882	0.2044902	0.0004225472

Table 2.1.2: Bernoulli Test Results for VaR Backtesting

Table 2.1.2 reveals the results of Unconditional Coverage (Bernoulli) test for the three VaR(0.05) models. The p-values of both AR(1)-GARCH(1,1) and EWMA models are larger than the significant level. Both results lead to a failure to reject the null hypothesis of correct unconditional coverage, which is $H_0 : \eta \sim B(p)$. Conversely, the p-value of VaR model using Historic Simulation is significantly small. It rejects the null and accepts the alternative that the theoretical violation probability p does not match the empirical (observed) violation probability $\tilde{p}$.

b. Independence test and Joint test

iii. Independence test

Any two subsequent violations η_t and η_{t+1} should be be independent of each other. A violation today should not convey any information about the likelihood of a violation tomorrow. The independence test examines whether VaR violations occur independently over time.

The hypotheses for the test are formulated as follows:

H_0 : The probability of a violation tomorrow does not depend on today,
H_1 : The probability of a violation tomorrow depends on today (violation clustering).

The likelihood ratio test again is being used. Therefore, the test statistic can be computed as two times the difference between the unrestricted and restricted likelihood:

$$\text{LR} = 2(log\mathcal{L}_U(\hat{\sqcap}_1) - log\mathcal{L}_R(\hat{\sqcap}_0)) \sim \chi^2_{(1)}$$

Model	**AR-GARCH**	**EWMA**	**HS**
Test Statistic	1.4070027	0.4475540	5.56362180
p-value	0.2355546	0.5034988	0.01833744

Table 2.2.1: Independence Test Results for VaR Backtesting

Table 2.2.1 reveals the results of Independence test for the three VaR(0.05) models. Once again, the p-values of both AR(1)-GARCH(1,1) and EWMA models are larger than the significant level. Both results lead to a failure to reject the null hypothesis of independence, which means the probability of a violation tomorrow does not depend on today. Conversely, the p-value of

VaR model using Historic Simulation is significantly small. It rejects the null and accepts the alternative of violation clustering.

iv. Joint test

The joint test combines the unconditional coverage test and the independence test, allows to jointly test that if a VaR model satisfies both properties simultaneously, which means whether violations are significantly different from those expected and whether there is violation clustering.

The hypotheses for the test are formulated as follows:

H_0 : Correct violation rate and No clustering (Correct conditional coverage),

H_1 : Either unconditional coverage or independence (or both) is violated.

The test statistic can be computed as:

$$\mathrm{LR}_{(joint)} = \mathrm{LR}_{(coverage)} + \mathrm{LR}_{(independence)} \sim \chi^2_{(2)}$$

Model	**AR-GARCH**	**EWMA**	**HS**
Test Statistic	3.7009171	2.0575656	17.99338
p-value	0.1571651	0.3574418	0.0001238191

Table 2.2.2: Joint Test Results for VaR Backtesting

Table 2.2.2 reveals the results of Joint test for the three VaR(0.05) models. Since the joint test has less power to reject a VaR model which only satisfies one of the two properties, the individual tests in the previous parts should be used as the prior knowledge of the corresponding VaR model. The p-values of both AR(1)-GARCH(1,1) and EWMA models illustrate that both models satisfy both unconditional coverage and independence, which align with the previous test results. The VaR model using Historic Simulation again fails strongly for both unconditional coverage and independence.

Appendix

```r
# Load required packages
library(openxlsx)    # Excel file handling
library(lubridate)   # Date/time utilities
library(quantmod)    # Financial data & modeling
library(rugarch)     # GARCH/AR-GARCH models
library(ggplot2)     # Data visualization

raw <- read.xlsx('student_groups_stocks.xlsx', sheet = 1)

groupNumber <- 15
nameOfStock <- raw$Stock.Name[groupNumber]
startDate <- raw$Start.Date[groupNumber]
endDate <- raw$`End.Date.(+10y)`[groupNumber]

# compute the start date of the final five years
originalDate <- as.Date(startDate)
endDate_fiveYears <- originalDate %m+% years(5) + 1

# extract the daily stock price from source
getSymbols(nameOfStock, src = 'yahoo', from = startDate, to = endDate)
allDates <- as.Date(index(HSBC))

# compute the index of the start date of the final five years (including the day before)
endDate_fiveYears_index <- which(allDates == as.Date(endDate_fiveYears)) - 1
endDate_fiveYears <- allDates[endDate_fiveYears_index]

# extract the daily stock price of the last five years
lastFiveYears <- window(HSBC, start = endDate_fiveYears, end = endDate)

# extract the Adjusted Close for the last five years
adjustedPrice_lastFiveYears <- as.numeric(lastFiveYears[,'HSBC.Adjusted'])

compute_simpleReturns <- function(input_priceSeries){
  result <- numeric(length(input_priceSeries) - 1)

  for(i in (2:length(input_priceSeries))){
    result[i-1] <- (input_priceSeries[i] - input_priceSeries[i-1]) /
                    input_priceSeries[i-1]
  }

  return(result)
}

# compute the Daily Simple Returns for last five years
simpleReturns_lastFiveYears <- compute_simpleReturns(adjustedPrice_lastFiveYears)

# specify AR(1)-GARCH(1,1) model under normal distribution
ar1garch11_spec_norm <- ugarchspec(variance.model = list(model = 'sGARCH',
                                                         garchOrder = c(1, 1)),
                        mean.model = list(armaOrder = c(1, 0),
```

```
                         include.mean = TRUE),
                         distribution.model = 'norm')

# extract the date of the final five years (not including the day before)
date_lastFiveYear <- index(lastFiveYears)[2:1260]

testWindow <- length(date_lastFiveYear)

# initiate the estimation window and re-estimation duration
estimationWindow <- 1000
reestimationDuration <- 20

# compute the number of re-estimation
numberOfReestimation <- round(testWindow / reestimationDuration)

forecast_returns <- numeric(testWindow)
forecast_variance <- numeric(testWindow)

omega_list <- numeric(numberOfReestimation)
alpha_list <- numeric(numberOfReestimation)
beta_list <- numeric(numberOfReestimation)

# compute the returns and variance forecast for the next 5 years
for (i in 1:numberOfReestimation){
  # compute the start and end date for this estimation
  this_endDate_index <- endDate_fiveYears_index + reestimationDuration*(i-1) -1
  this_startDate_index <- this_endDate_index - estimationWindow
  this_endDate <- allDates[this_endDate_index]
  this_startDate <- allDates[this_startDate_index]

  # extract the data of this window
  thisWindow <- window(HSBC, start = this_startDate, end = this_endDate)

  # extract the Adjusted Close for the this estimation window
  adjustedPrice_thisWindow<- as.numeric(thisWindow[,'HSBC.Adjusted'])

  # compute the Daily Simple Returns for this estimation window
  simpleReturns_thisWindow <- compute_simpleReturns(adjustedPrice_thisWindow)

  # Fit AR(1)  GARCH  (1,1)
  ar1garch11_fit_norm <- ugarchfit(spec = ar1garch11_spec_norm,
                                   data = simpleReturns_thisWindow)

  # extract the estimated parameters
  mu_norm <- coef(ar1garch11_fit_norm)['mu']
  ar1_norm <- coef(ar1garch11_fit_norm)['ar1']
  omege_norm <- coef(ar1garch11_fit_norm)['omega']
  alpha1_norm <- coef(ar1garch11_fit_norm)['alpha1']
  beta1_norm <- coef(ar1garch11_fit_norm)['beta1']
  sum_alphaBeta <- alpha1_norm + beta1_norm

  # store and track the estimated GARCH parameters
  omega_list[i] <- omege_norm
  alpha_list[i] <- alpha1_norm
```

```
  beta_list[i] <- beta1_norm

  # extract the variables for forecasts computation
  returns_t <- simpleReturns_thisWindow[estimationWindow]
  eps_t <- ar1garch11_fit_norm@fit$residuals[estimationWindow]
  var_t <- ar1garch11_fit_norm@fit$var[estimationWindow]

  # compute the 1-step ahead variance
  oneAhead_var <- omege_norm + alpha1_norm * (eps_t^2) + beta1_norm * (var_t)

  for (j in 1:reestimationDuration){
    # compute the estimation time t
    this_t <- reestimationDuration*(i-1) + j

    if (this_t <= testWindow){
      # Forecast for Simple Returns
      forecast_returns[this_t] <- (sum(sapply(0:(j-1), function(k) ar1_norm^k)) * mu_norm) + ar1_norm^j * returns_t

      # Forecast for Variance
      if (j == 1) {
        forecast_variance[this_t] <- oneAhead_var
      } else {
        forecast_variance[this_t] <- (sum(sapply(0:(j-2),
                                           function(k) sum_alphaBeta^k))) * omege_norm + sum_alphaBeta^(j-1) * oneAhead_var
      }
    }
  }
}

parameter_df <- data.frame(omega = omega_list,
                           alpha = alpha_list,
                           beta = beta_list,
                           number = seq_along(omega_list))

ggplot(parameter_df, aes(x = number)) +
  geom_line(aes(y = omega, color = 'Omega'),
            linewidth = 1) +
  geom_line(aes(y = alpha, color = 'Alpha'),
            linewidth = 1) +
  geom_line(aes(y = beta, color = 'Beta'),
            linewidth = 1) +
  labs(title = 'Parameters of GARCH(1,1) for HSBC Daily Simple Returns (2015M11-2020M11)',
    x = 'Number of Re-Estimation', y = 'Values', color = 'Series') +
  scale_color_manual(values = c('Omega' = 'darkblue',
                                'Alpha' = 'darkred',
                                'Beta' = 'darkgreen')) +
  theme_minimal() +
  theme(legend.position = c(0.9, 0.9),
        legend.background = element_rect(fill = 'white', color = 'grey'),
        legend.title = element_text(size = 12, face = 'bold'),
        legend.text  = element_text(size = 10))
```

```
ggsave('figures/a_parameter_tracePlot.png', width = 10, height = 6, dpi
    = 300)
```

Figure 1: Question 1(a)

```
valueAtRisk_Parametric <- function(input_forecastReturns,
                                   input_forecastVariance,
                                   input_probability){
  valueAtRisk <- numeric(length(input_forecastReturns))

  for (i in 1:testWindow){
    valueAtRisk[i] <- -sqrt(input_forecastVariance[i]) * qnorm(input_
       probability) - input_forecastReturns[i]
  }

  return(valueAtRisk)
}

# compute the forecast of VaR for the last five years
valueAtRisk_NonParametric <- function(input_window, input_probability){
  VaR_hs <- numeric(length(simpleReturns_lastFiveYears))

  for(i in 1:testWindow){
    this_endDate_index <- endDate_fiveYears_index + (i-1) - 1
    this_startDate_index <- this_endDate_index - input_window
    this_endDate <- allDates[this_endDate_index]
    this_startDate <- allDates[this_startDate_index]

    duration <- window(HSBC, start = this_startDate, end = this_endDate
       )

    # extract the Adjusted Close for the next five years
    this_adjustedPrice <- as.numeric(duration[,'HSBC.Adjusted'])

    this_simpleReturns <- compute_simpleReturns(this_adjustedPrice)
    sorted_this_simpleReturns <- sort(this_simpleReturns)

    VaR_hs[i] <- -sorted_this_simpleReturns[input_window*input_
       probability]
  }
  return(VaR_hs)
}

computeEWMAVolatility <- function(input_lambda, input_y0,
                                  input_sigmaSquared0, input_returns){
  volatility <- numeric(length(input_returns))

  volatility[1] <- sqrt((1 - input_lambda) * input_y0^2 + input_lambda
     * input_sigmaSquared0)

  for (i in 2:(testWindow)){
    volatility[i] <- sqrt((1 - input_lambda) * (input_returns[i-1]^2) +
        input_lambda * volatility[i-1]^2)
  }

  return(volatility)
}

# specific the probability of losses
```

```
probability <- 0.05
# specific estimation window length for Historic Simulation (Non-Parametric)
hs_window <- 1000
# compute the EWMA volatility
lambda <- 0.94
initializeDuration <- 30
initialMean <- mean(simpleReturns_lastFiveYears[1:30])
initialVariance <- var(simpleReturns_lastFiveYears[1:30])
assumedMu_EWMA <- numeric(testWindow)

EWMA_volatility <- computeEWMAVolatility(input_lambda = lambda,
                                         input_y0 = initialMean,
                                input_sigmaSquared0 = initialVariance,
                                input_returns = simpleReturns_lastFiveYears)

VaR_GARCH <- valueAtRisk_Parametric(input_forecastReturns = forecast_returns,
                                input_forecastVariance = forecast_variance,
                                input_probability = probability)
VaR_HS <- valueAtRisk_NonParametric(input_window = hs_window,
                                    input_probability = probability)
VaR_EWMA <- valueAtRisk_Parametric(input_forecastReturns = assumedMu_EWMA,
                                input_forecastVariance = EWMA_volatility^2,
                                input_probability = probability)

VaR_df <- data.frame(date = date_lastFiveYear,
                     VaR_ARGARCH = -VaR_GARCH,
                     VaR_HS = -VaR_HS,
                     VaR_EWMA = -VaR_EWMA,
                     simpleReturns_realized = simpleReturns_lastFiveYears)

ggplot(VaR_df, aes(x = date)) +
  geom_hline(yintercept = 0, color = 'black',
             linewidth = 1, linetype = 'dashed') +
  geom_line(aes(y = simpleReturns_realized, color = 'Realized Returns'),
            linewidth = 1) +
  geom_line(aes(y = VaR_ARGARCH, color = 'AR(1)-GARCH(1,1)'),
            linewidth = 1) +
  geom_line(aes(y = VaR_EWMA, color = 'EWMA'),
            linewidth = 1) +
  geom_line(aes(y = VaR_HS, color = 'HS'),
            linewidth = 1) +
  labs(title = 'Realized vs VaR(0.05) for HSBC Daily Simple Returns (2015M11-2020M11)',
    x = 'Date', y = 'Simple Returns', color = 'Series') +
  scale_color_manual(values = c('Realized Returns' = 'darkgrey',
                                'AR(1)-GARCH(1,1)' = 'darkblue',
                                'EWMA' = 'darkred',
                                'HS' = 'darkgreen')) +
  theme_minimal() +
  theme(legend.position = c(0.9, 0.9),
```

```
        legend.background = element_rect(fill = 'white', color = 'grey'
           ),
        legend.title = element_text(size = 12, face = 'bold'),
        legend.text  = element_text(size = 10))
ggsave('figures/b_VaR_plot.png', width = 10, height = 6, dpi = 300)
```

Figure 2: Question 1(b)

```
computeViolationRatio <- function(input_probability,
                                  input_testWindow,
                                  input_VaR, input_realized){
  expectedViolation <- input_probability * input_testWindow
  eta <- numeric(input_testWindow)
  for(i in 1:input_testWindow){
    if(input_realized[i] <= -input_VaR[i]){
      eta[i] <- 1
    } else {
      eta[i] <- 0
    }
  }
  violationRatio <- sum(eta == 1) / expectedViolation

  return(list(eta=eta, violationRatio=violationRatio))
}

vr_ARGARCH <- computeViolationRatio(input_probability = probability,
                                    input_testWindow = testWindow,
                                    input_VaR = VaR_GARCH,
                                input_realized = simpleReturns_
                                   lastFiveYears)
vr_EWMA <- computeViolationRatio(input_probability = probability,
                                 input_testWindow = testWindow,
                                 input_VaR = VaR_EWMA,
                                 input_realized = simpleReturns_
                                    lastFiveYears)
vr_HS <- computeViolationRatio(input_probability = probability,
                               input_testWindow = testWindow,
                               input_VaR = VaR_HS,
                               input_realized = simpleReturns_
                                  lastFiveYears)

violationRatio_df = data.frame(Model = 'violationRatio',
                               AR_GARCH = vr_ARGARCH$violationRatio,
                               EWMA = vr_EWMA$violationRatio,
                               HS = vr_HS$violationRatio)
violationRatio_df

bern_test_function <- function(p, v) {
  # Number of successes and total trials
  num_successes <- sum(v)
  total_trials <- length(v)

  # Null hypothesis proportion
  p_null <- p

  # Observed proportion
  p_observed <- num_successes / total_trials

  # Likelihood under null hypothesis
```

```
  likelihood_null <- p_null^num_successes * (1 - p_null)^(total_trials
     - num_successes)

  # Likelihood under observed proportion
  likelihood_observed <- p_observed^num_successes * (1 - p_observed)^(
     total_trials - num_successes)

  # Calculate the test statistic (Likelihood Ratio Test statistic)
  test_statistic <- 2 * log(likelihood_observed / likelihood_null)

  # Compute p-value from chi-squared distribution with 1 degree of
     freedom
  p_value <- 1 - pchisq(test_statistic, df = 1)

  # Return the test statistic and p-value
  list(test_statistic = test_statistic, p_value = p_value)
}

bern_test_ARGARCH <- bern_test_function(probability, vr_ARGARCH$eta)
bern_test_EWMA <- bern_test_function(probability, vr_EWMA$eta)
bern_test_HS <- bern_test_function(probability, vr_HS$eta)

bern_test_df = data.frame(Model = c('TestStatistic','pValue'),
    AR_GARCH = c(bern_test_ARGARCH$test_statistic,bern_test_ARGARCH$p_
       value),
    EWMA = c(bern_test_EWMA$test_statistic,bern_test_EWMA$p_value),
    HS = c(bern_test_HS$test_statistic,bern_test_HS$p_value))

bern_test_df
```

Figure 3: Question 2(a)

```
ind_test_function <- function(violations) {
  # Calculate transition frequencies
  v00 <- sum(violations[-1] == 0 & violations[-length(violations)] ==
     0)
  v01 <- sum(violations[-1] == 1 & violations[-length(violations)] ==
     0)
  v10 <- sum(violations[-1] == 0 & violations[-length(violations)] ==
     1)
  v11 <- sum(violations[-1] == 1 & violations[-length(violations)] ==
     1)

  # Total number of transitions
  V <- v00 + v01 + v10 + v11

  # Calculate transition probabilities
  p_hat <- (v01 + v11) / V
  p01_hat <- v01 / (v00 + v01)
  p11_hat <- v11 / (v10 + v11)

  # Calculate likelihoods under the null hypothesis of independence
  LR <- (1 - p_hat)^(v00 + v10) * p_hat^(v01 + v11)

  # Calculate likelihoods under the observed transition probabilities
  LU <- (1 - p01_hat)^v00 * p01_hat^v01 * (1 - p11_hat)^v10 * p11_hat^
     v11

  # Calculate the log likelihood ratio
```

```r
  test_statistic <- 2 * log(LU / LR)

  # Compute p-value from chi-squared distribution with 1 degree of freedom
  p_value <- 1 - pchisq(test_statistic, df = 1)

  # Return the test statistic and p-value
  list(test_statistic = test_statistic, p_value = p_value)
}

ind_test_ARGARCH <- ind_test_function(vr_ARGARCH$eta)
ind_test_EWMA <- ind_test_function(vr_EWMA$eta)
ind_test_HS <- ind_test_function(vr_HS$eta)

ind_test_df = data.frame(Model = c('TestStatistic','pValue'),
      AR_GARCH = c(ind_test_ARGARCH$test_statistic,ind_test_ARGARCH$p_value),
      EWMA = c(ind_test_EWMA$test_statistic,ind_test_EWMA$p_value),
      HS = c(ind_test_HS$test_statistic,ind_test_HS$p_value))

ind_test_df

joint_test <- function(input_LR_coverage, input_LR_independence){
  # Calculate the new test statistic
  new_LR <- input_LR_coverage + input_LR_independence

  # Compute p-value from chi-squared distribution with 2 degree of freedom
  p_value <- 1 - pchisq(new_LR, df = 2)

  # Return the test statistic and p-value
  list(test_statistic = new_LR, p_value = p_value)
}

joint_test_ARGARCH <- joint_test(
  input_LR_coverage = bern_test_ARGARCH$test_statistic,
  input_LR_independence = ind_test_ARGARCH$test_statistic)
joint_test_EWMA <- joint_test(
  input_LR_coverage = bern_test_EWMA$test_statistic,
  input_LR_independence = ind_test_EWMA$test_statistic)
joint_test_HS <- joint_test(
  input_LR_coverage = bern_test_HS$test_statistic,
  input_LR_independence = ind_test_HS$test_statistic)

joint_test_df = data.frame(Model = c('TestStatistic','pValue'),
  AR_GARCH = c(joint_test_ARGARCH$test_statistic,joint_test_ARGARCH$p_value),
  EWMA = c(joint_test_EWMA$test_statistic,joint_test_EWMA$p_value),
  HS = c(joint_test_HS$test_statistic,joint_test_HS$p_value))

joint_test_df
```

Figure 4: Question 2(b)