MATH60633 – TP2

Achille Leturger, Philippe Gagné et Tarak Ben Abda

2024-04-16

Description

L'objectif est de mettre en oeuvre (une partie) du cadre de gestion des risques pour estimer le risque d'un portfefeuille d'options d'achat européennes en tenant compte des facteurs de risque tels que le sous-jacente et la volatilité implicite.

Partie 1 - Données

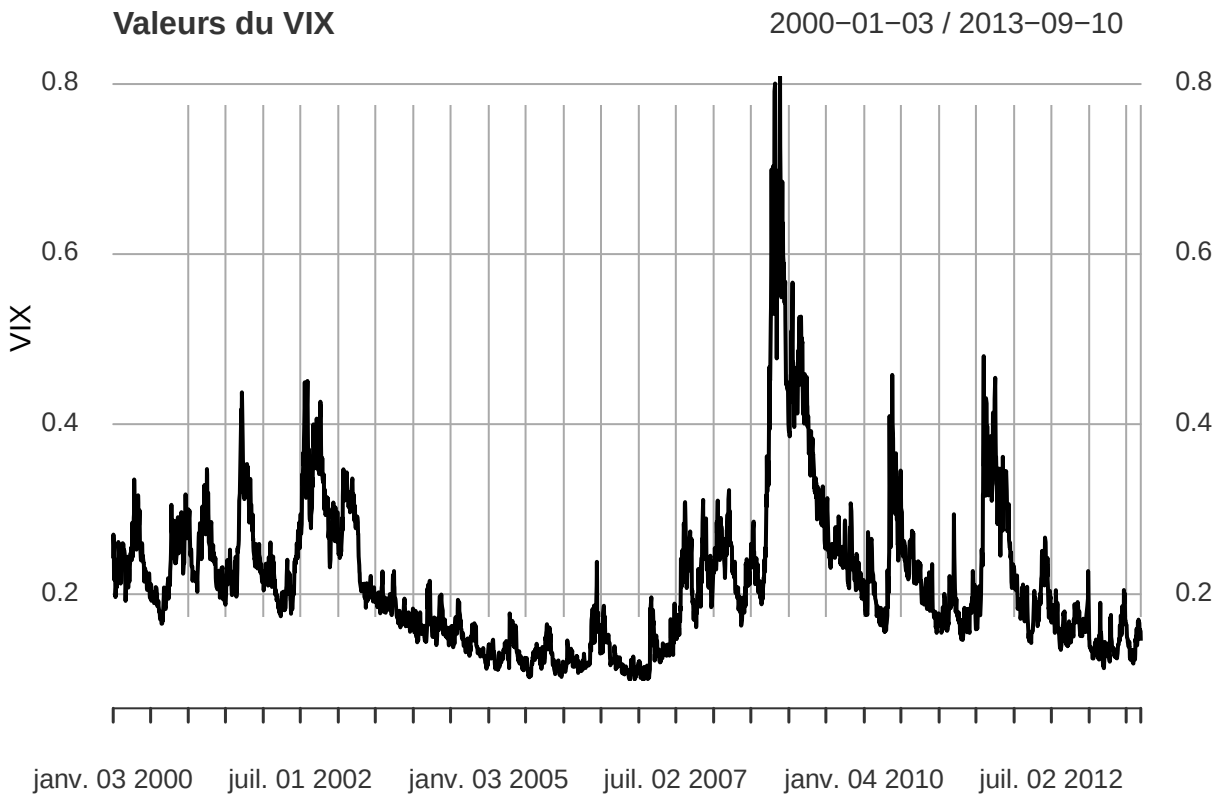
1. Chargez la base de données Market

```
load(file = here("data","Market.rda"))
```

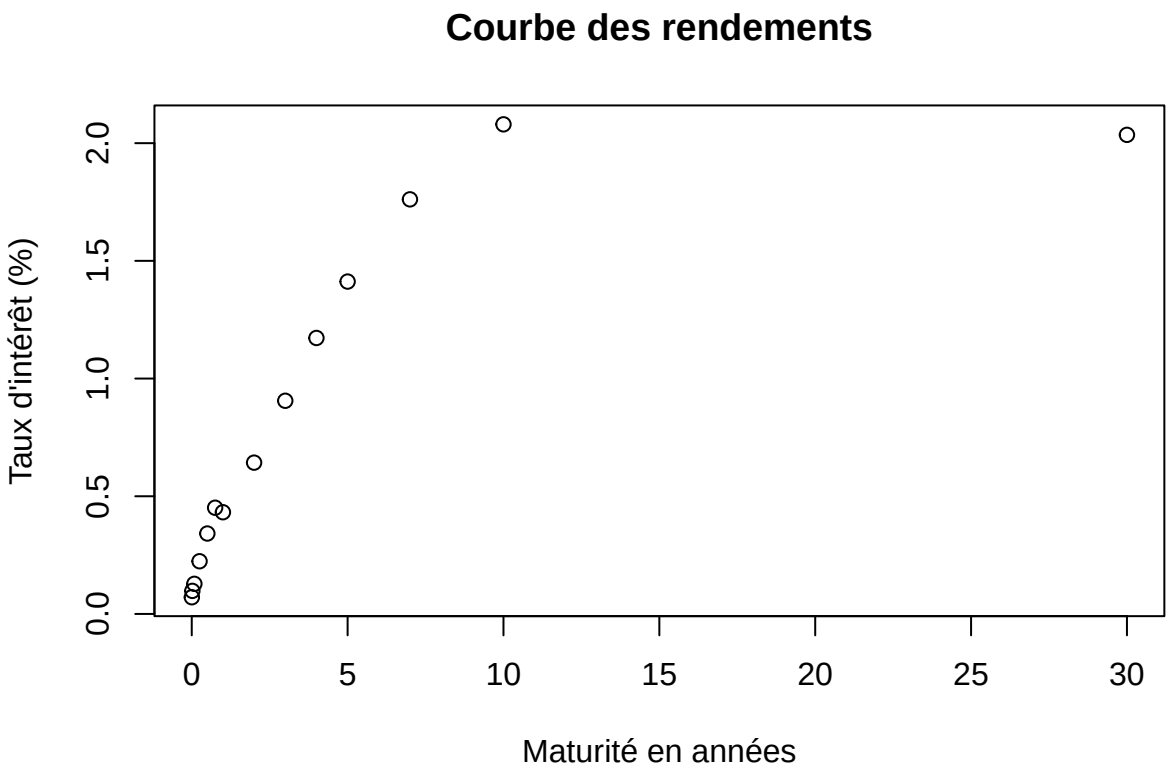
2. Identifiez le prix du S&P 500, l'indice VIX, la structure des taux d'intérêt (actuels et passés), et les options négociées (calls et puts).

```
sp500_prices <- Market$sp500  # Prix du S&P 500
vix          <- Market$vix    # Valeurs du VIX
rf           <- Market$rf     # Taux sans risque
rf_maturity  <- as.numeric(names(rf))  # Maturités des taux sans risque
calls        <- as.data.frame(Market$calls)  # Données des options d'achat
puts         <- as.data.frame(Market$puts)   # Données des options de vente

# Ci-dessous, nous traçons les prix du S&P 500 et les valeurs du VIX sur la période observée.
plot(sp500_prices, main = "Prix du S&P 500", xlab = "Temps", ylab = "Prix")
```


```r
plot(vix, main = "Valeurs du VIX", xlab = "Temps", ylab = "VIX")
```




```r
# Ensuite, nous visualisons les taux annualisés de la courbe des rendements fournie.
plot(x    = rf_maturity,
     y    = rf*100,
     main = "Courbe des rendements",
     xlab = "Maturité en années",
     ylab = "Taux d'intérêt (%)")
```



Partie 2 - Fixation du prix d'un portefeuille d'options

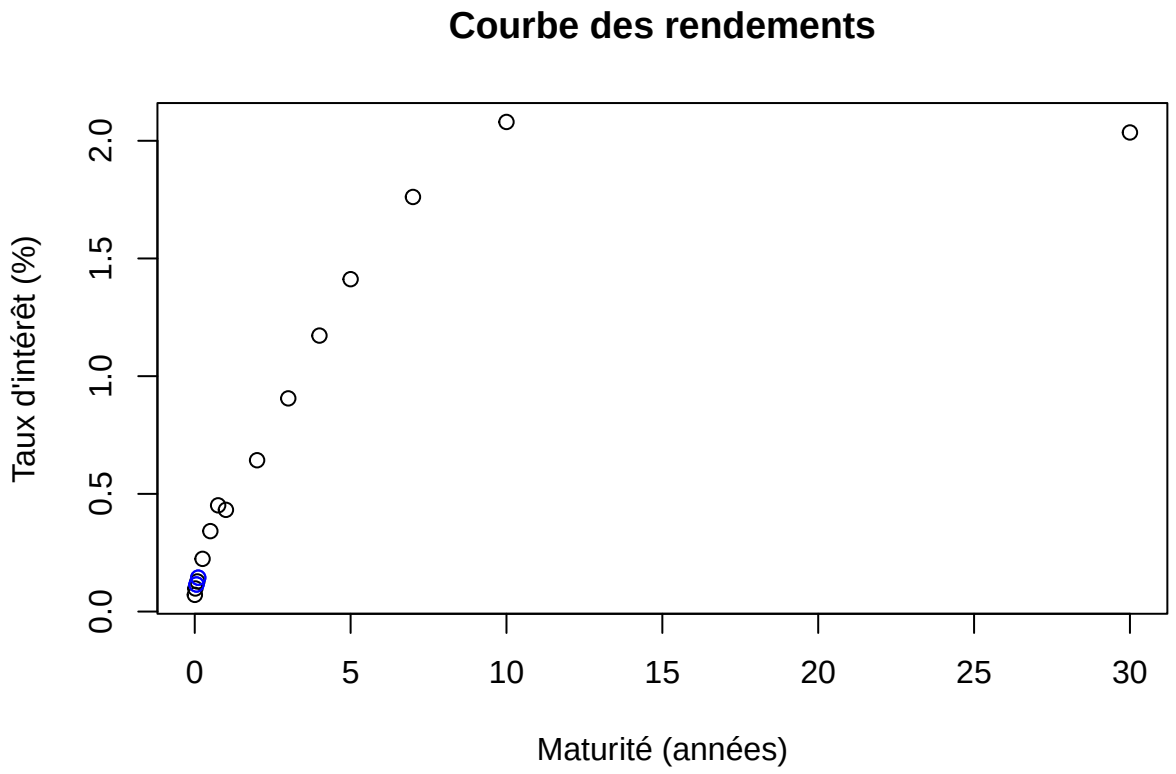
1. Supposez le portefeuille suivant d'options d'achat européennes : 1x strike K = 1600 échéance 20 jours, 1x strike K = 1650 échéance 20 jours, 1x strike K = 1750 échéance 40 jours et 1x strike K = 1800 échéance 40 jours.

```
days_year_tau <- 250  # Nombre de jours de trading par an pour le calcul de tau
days_year_r   <- 360  # Nombre de jours par an pour le calcul des taux
days_book     <- c(20, 20, 40, 40)  # Durées des options en jours
last_price    <- as.numeric(sp500_prices[length(sp500_prices)])  # Dernier prix du S&P 500
last_vix      <- as.numeric(vix[length(vix)])  # Dernière valeur du VIX

S     <- rep(last_price, 4)  # Prix sous-jacent répété pour chaque option
K     <- c(1600, 1650, 1750, 1800)  # Prix d'exercice des options
tau   <- days_book/days_year_tau  # Conversion des jours en années fractionnaires pour tau
vol   <- rep(last_vix, 4)  # Volatilité répétée pour chaque option
r     <- rep(NA, 4)  # Taux d'intérêt initialisés à NA
price <- rep(NA, 4)  # Prix initialisés à NA

init_book <- data.frame(S, K, tau, vol, r, price)  # Création du dataframe pour le portefeuille d'optio
```

2. Trouvez le prix de ce portefeuille en fonction du dernier prix du sous-jacent et de la dernière volatilité implicite (prenez le VIX pour toutes les options). Utilisez la formule de Black-Scholes pour évaluer les options. Prenez la structure actuelle des taux d'intérêt et interpolez linéairement pour trouver les taux

correspondants. Utilisez 360 jours/an pour la structure des taux et 250 jours/an pour la maturité des options.

```r
# Interpolation linéaire de la structure à terme pour trouver les taux sans risque pour notre portefeui
init_book$r <- approx(x    = rf_maturity,
                      y    = rf,
                      xout = days_book/days_year_r)$y  # Interpolation des taux pour les durées des opt

# Le graphique ci-dessous montre les taux sans risque utilisés pour notre portefeuille en bleu, comparé
plot(x    = rf_maturity,
     y    = rf*100,
     main = "Courbe des rendements",
     xlab = "Maturité (années)",
     ylab = "Taux d'intérêt (%)")
points(x   = days_book/days_year_r,
       y   = init_book$r*100,
       col = "blue")  # Points pour les taux interpolés
```




```r
# Tarification de notre portefeuille d'options d'achat selon la formule de Black-Scholes, et calcul de
init_book$price <- blackscholes(callput = 1,
                                S0      = init_book$S,
                                K       = init_book$K,
                                r       = init_book$r,
                                time    = init_book$tau,
                                vola    = init_book$vol)$Price  # Calcul des prix selon Black-Scholes
```

```r
init_book_value <- sum(init_book$price)  # Somme des prix pour obtenir la valeur du portefeuille

init_book_value  # Affichage de la valeur totale du portefeuille d'options
```

```
## [1] 156.975
```

Partie 3 - Un facteur de risque et un modèle gaussien univarié

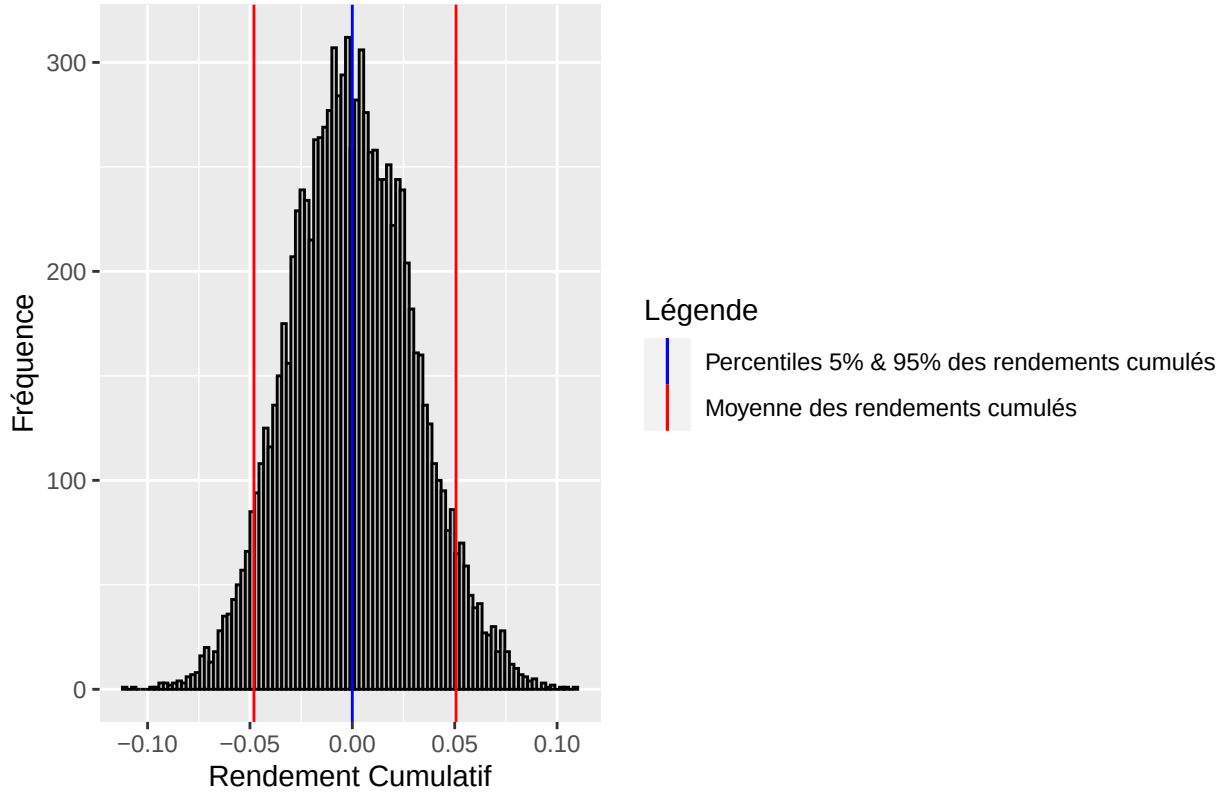
1. Calculez les rendements logarithmiques quotidiens de l'action sous-jacente.

```r
logrets_sp500 <- Return.calculate(prices = sp500_prices,
                                  method = "log")

logrets_sp500 <- logrets_sp500[-1]

# Estimation de la moyenne et de l'écart-type des rendements logarithmiques de l'S&P 500.
mean_rets_sp500 <- mean(logrets_sp500)  # Calcul de la moyenne des rendements logarithmiques
sd_rets_sp500 <- sd(logrets_sp500)      # Calcul de l'écart-type des rendements logarithmiques
```

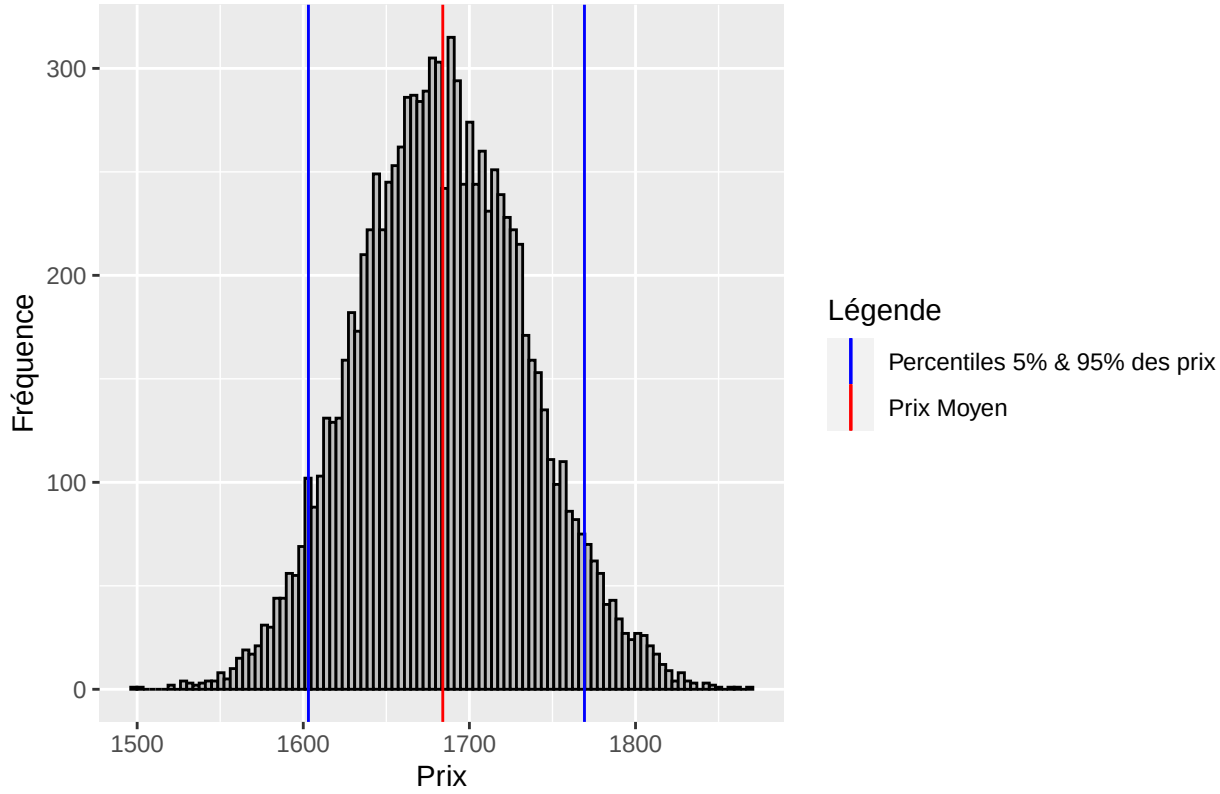
2. Supposez qu'ils sont normalement distribués de manière iid.
3. Générez 10 000 scénarios pour le prix sous-jacent à une semaine d'avance (cinq jours) en utilisant la distribution normale ajustée aux invariants passés.

```r
n_sim <- 10000  # Nombre de simulations
n_days <- 5  # Nombre de jours à simuler

set.seed(555) # Pour la reproductibilité du code

# Simulation des prix de l'S&P 500 selon un modèle gaussien univarié
sim_sp500_univ <- f_simulate_univ_gauss(n_sim      = n_sim,
                                        n_days     = n_days,
                                        mean       = mean_rets_sp500,  # Moyenne des rendements du S&P
                                        sd         = sd_rets_sp500,    # Écart-type des rendements du S
                                        lastprice  = last_price)       # Dernier prix connu du S&P 500
sim_rets_sp500_univ   <- unlist(sim_sp500_univ[1])  # Rendements simulés
sim_prices_sp500_univ <- unlist(sim_sp500_univ[2])  # Prix simulés

# Visualisation des rendements cumulés simulés et des prix.
f_plot_sim_univ_gauss(returns = sim_rets_sp500_univ,
                      prices  = sim_prices_sp500_univ)
```

fs Simulés sur Une Semaine de l'S&P 500 − Modèle Gaussien Univ.



és Simulés sur Une Semaine de l'S&P 500 − Modèle Gaussien Univ.




```
# Création du portefeuille d'options à une semaine.

# Mise à jour des jours restants pour chaque option d'achat
new_days_book <- days_book - n_days

# Suppression des colonnes inutiles pour l'instant
new_book <- init_book[c("K","tau","vol","r")]

# Mise à jour du temps jusqu'à maturité
new_book$tau <- new_days_book/days_year_tau

# Réalisation de nouvelles interpolations pour trouver les nouveaux taux sans risque
new_book$r <- approx(x    = rf_maturity,
                     y    = rf,
                     xout = new_days_book/days_year_r)$y

# Calcul des prix simulés à une semaine des options et des valeurs simulées du portefeuille.
new_book_values_univ <- f_Get_Book_values_univ(S    = sim_prices_sp500_univ,
                                               book = new_book)
```

4. Déterminez la distribution P&L du portefeuille d'options, en utilisant les valeurs sous-jacentes simulées. Supposez que la volatilité implicite reste la même. Prenez des taux interpolés pour la structure des taux.

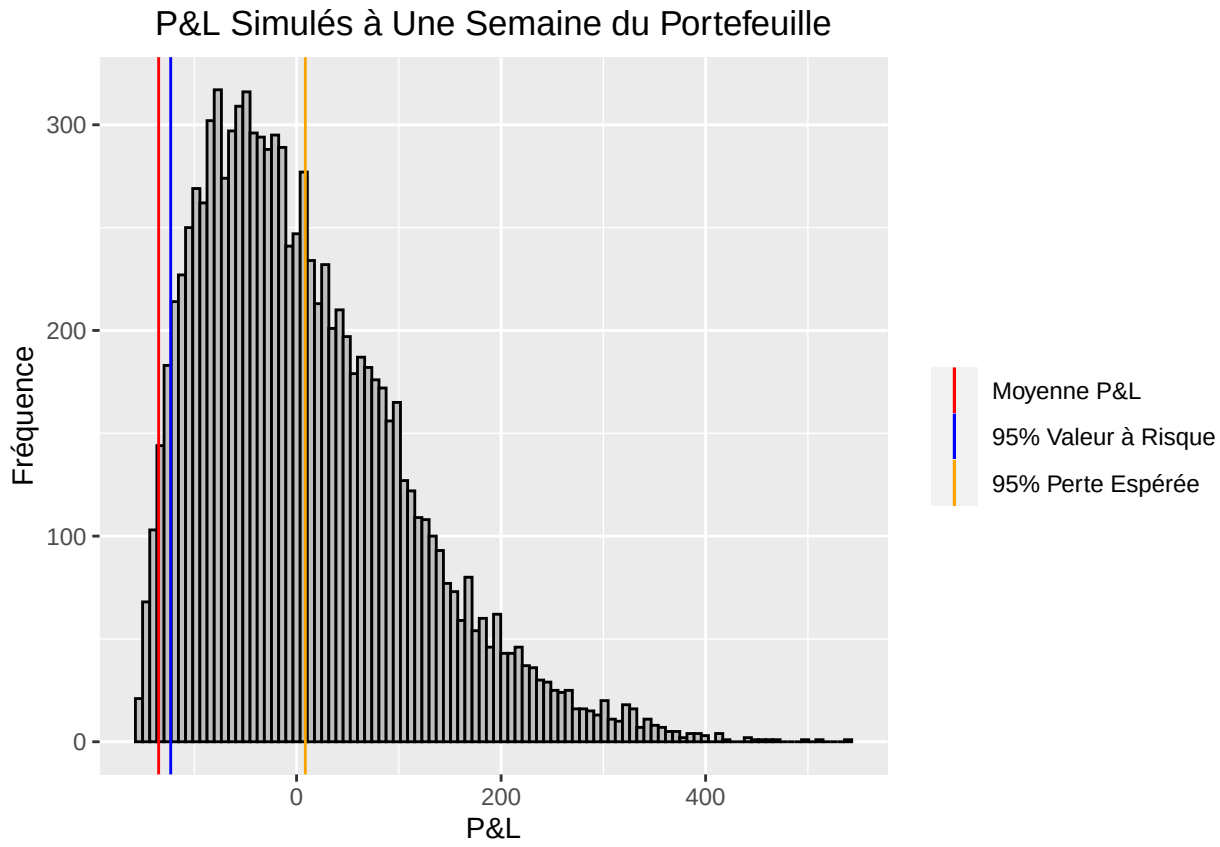
```r
pnl_univ_gauss <- new_book_values_univ - init_book_value  # Calcul des P&L
```

5. Calculez la VaR95 et l'ES95.

```r
level <- 0.95  # Niveau de confiance
PnLstats  <- data.frame(row.names = c("VaR", "ES"))  # Création d'un dataframe pour stocker les résulta

VaR95_univ_gauss <- f_VaR(pnl_univ_gauss, level)  # Calcul de la Valeur à Risque à 95%
ES95_univ_gauss  <- f_ES(pnl_univ_gauss, level)  # Calcul de la Perte Espérée à 95%

PnLstats["Univariate Gaussian"] <- c(VaR95_univ_gauss, ES95_univ_gauss)  # Enregistrement des résultats

PnLstats["Univariate Gaussian"]
```

```
##     Univariate Gaussian
## VaR           -123.1081
## ES            -134.9003
```

```r
# Visualisation de la distribution des P&L et des mesures de risque correspondantes.
f_Visualize_Portfolio_stats(data = pnl_univ_gauss,
                            VaR  = VaR95_univ_gauss,
                            ES   = ES95_univ_gauss)
```



Partie 4 - Deux facteurs de risque et modèle gaussien bivarié

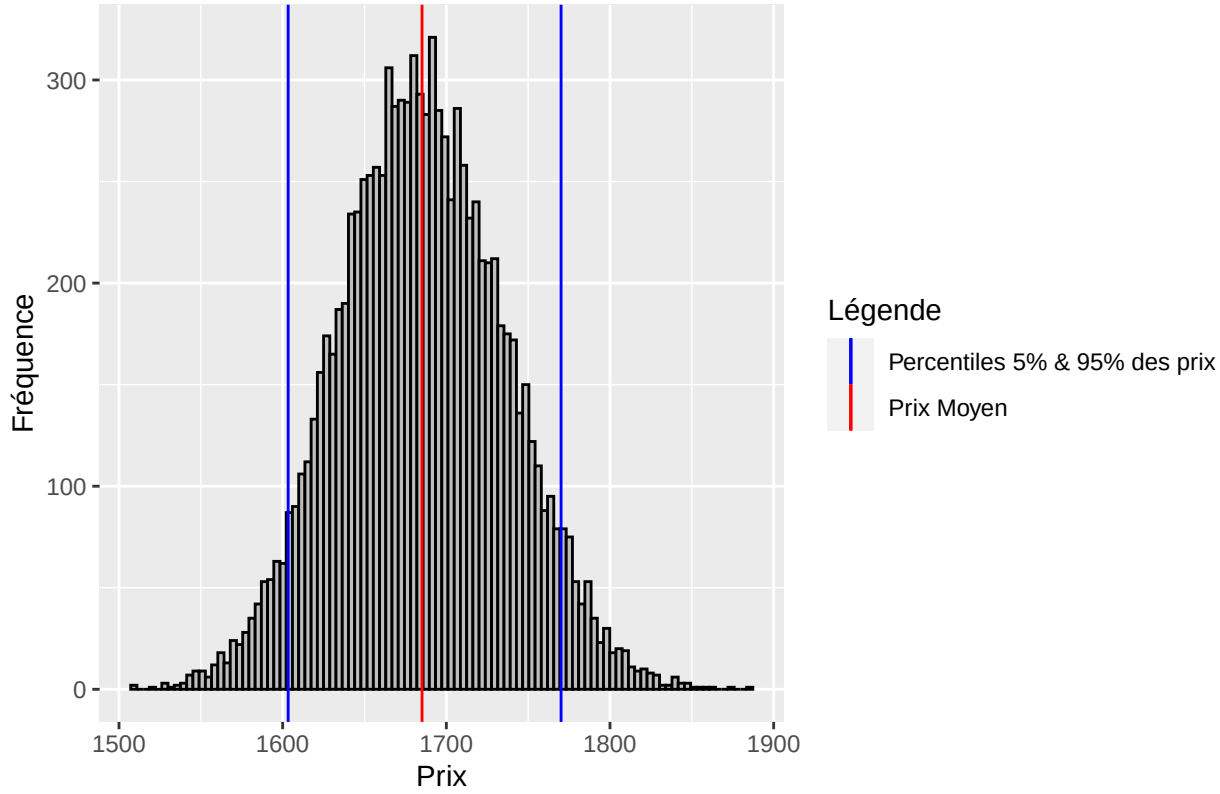
1. Calculez les rendements logarithmiques quotidiens de l'action sous-jacente.
2. Calculez les rendements logarithmiques quotidiens du VIX.

```
logrets_vix <- Return.calculate(prices = vix,
                                method = "log")
logrets_vix <- logrets_vix[-1]

mean_rets_vix <- mean(logrets_vix)

logrets <- cbind(logrets_sp500, logrets_vix)
mu      <- c(mean_rets_sp500, mean_rets_vix)
sigma   <- cov(logrets)
```

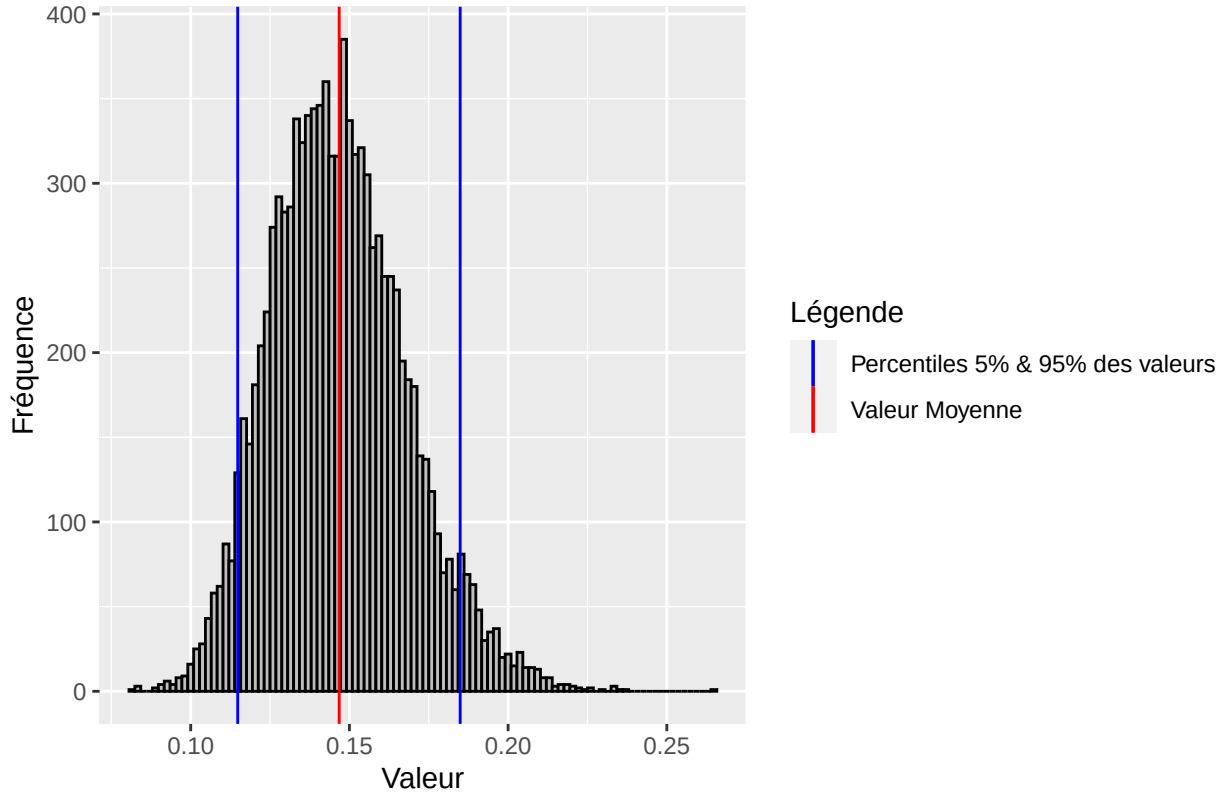
3. Supposons qu'ils sont normalement distribués de manière invariante.
4. Générez 10 000 scénarios pour le prix du sous-jacent à une semaine d'avance et la valeur du VIX à une semaine d'avance en utilisant la distribution normale ajustée aux facteurs de risque passés.

```
set.seed(555)  # Pour la reproductibilité du code

sim_multiv <- f_simulate_multiv_gauss(n_sim      = n_sim,
                                      n_days     = n_days,
                                      mu         = mu,
                                      sigma      = sigma,
                                      lastprice1 = last_price,
                                      lastprice2 = last_vix)
sim_prices_sp500_mv <- unlist(sim_multiv[1])
sim_vix_mv          <- unlist(sim_multiv[2])

# Visualiser les rendements cumulatifs simulés et les prix de l'S&P 500 ainsi que les valeurs du VIX, e

f_plot_sim_multiv_gauss(prices  = sim_prices_sp500_mv,
                        vix     = sim_vix_mv)
```

ès Simulés sur Une Semaine de l'S&P 500 − Modèle Gaussien Multiv.



Simulées sur Une Semaine du VIX − Modèle Gaussien Multiv.



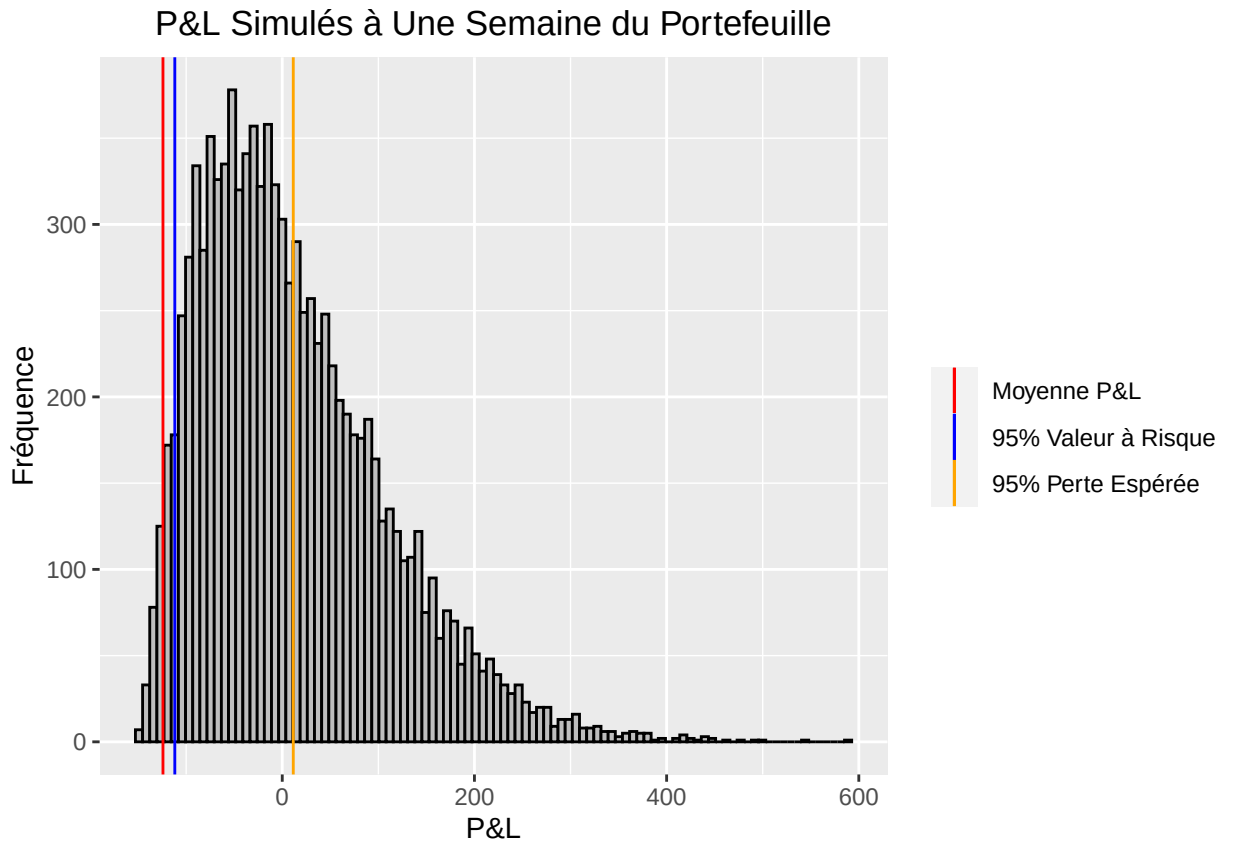

```
# Obtenir les nouvelles valeurs du portefeuille de la simulation sous la distribution multivariée.

new_book_values_mv <- f_Get_Book_values(S = sim_prices_sp500_mv,
                                        book = new_book,
                                        vol = sim_vix_mv)
```

5. Déterminez la distribution des pertes et profits du portefeuille d'options, en utilisant les valeurs simulées. Prenez des taux interpolés pour la structure des taux.

```
pnl_mv_gauss <- new_book_values_mv - init_book_value

# Calcul de la VaR95 et l'ES95.

VaR95_mv_gauss <- f_VaR(pnl_mv_gauss, level)
ES95_mv_gauss  <- f_ES(pnl_mv_gauss, level)
PnLstats["Multivariate Gaussian"] <- c(VaR95_mv_gauss, ES95_mv_gauss)

PnLstats["Multivariate Gaussian"]
```

```
##     Multivariate Gaussian
## VaR             -111.8175
## ES              -124.1529
```

```r
# Visualisation de la distribution des P&L et des mesures de risque correspondantes.

f_Visualize_Portfolio_stats(data = pnl_mv_gauss,
                            VaR  = VaR95_mv_gauss,
                            ES   = ES95_mv_gauss)
```



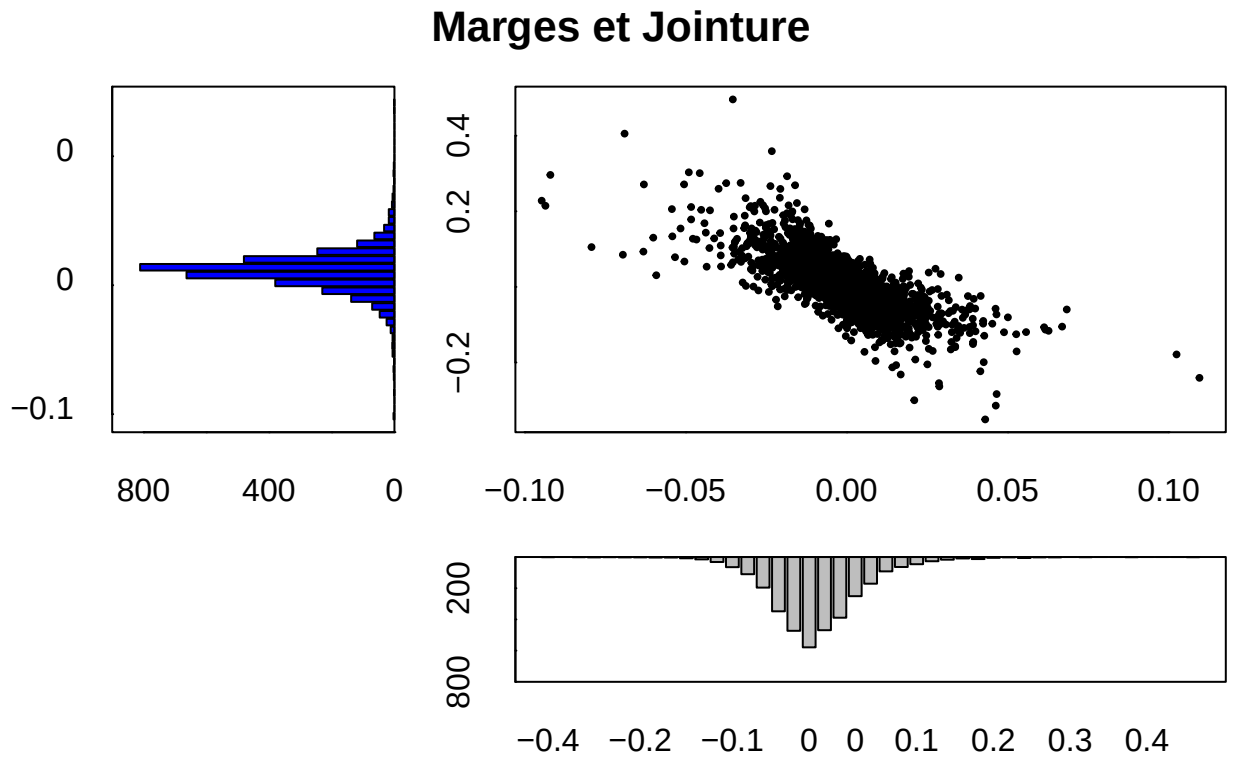
Nous pouvons remarquer que la VaR et l'ES donne des perte plus faible dans le cas de la distribution gaussienne multivariée que dans le cas de la distribution gaussienne univariée. Cela est dû à la corrélation négative entre les rendements de l'S&P 500 et du VIX, qui est prise en compte dans le modèle multivarié. Cette corrélation négative permet de limiter la baisse de prix liée aux rendements très faibles par l'augmentation du VIX et donc de la volatilité des options.

Partie 5 - Deux facteurs de risque et modèle copule-marginal (Student-t et copule gaussienne)

1. Procédez comme précédemment, mais supposez maintenant que le premier invariant est généré à l'aide d'une distribution de Student-t avec = 10 degrés de liberté et que le deuxième invariant est généré en utilisant une distribution de Student-t avec = 5 degrés de liberté. Supposons que la copule normale pour fusionner les marginales. Recalculez le P&L et les mesures de risque.

- Ajustement d'une copule : Observons d'abord un nuage de points pour vérifier s'il existe des corrélations linéaires claires entre les deux vecteurs.

```r
f_display_copula_margins(data   = cbind(as.numeric(logrets_sp500), as.numeric(logrets_vix)),
                         labels = c("SP500", "VIX"),
                         lim1   = 900,
                         lim2   = 800)
```



Nous pouvons observer une relation négative entre les rendements logarithmiques du S&P 500 et du VIX. Cela suggère que les deux variables sont corrélées négativement. Ainsi, la copule Gaussienne tiendra compte de cette relation dans la modélisation des distributions uniformes.

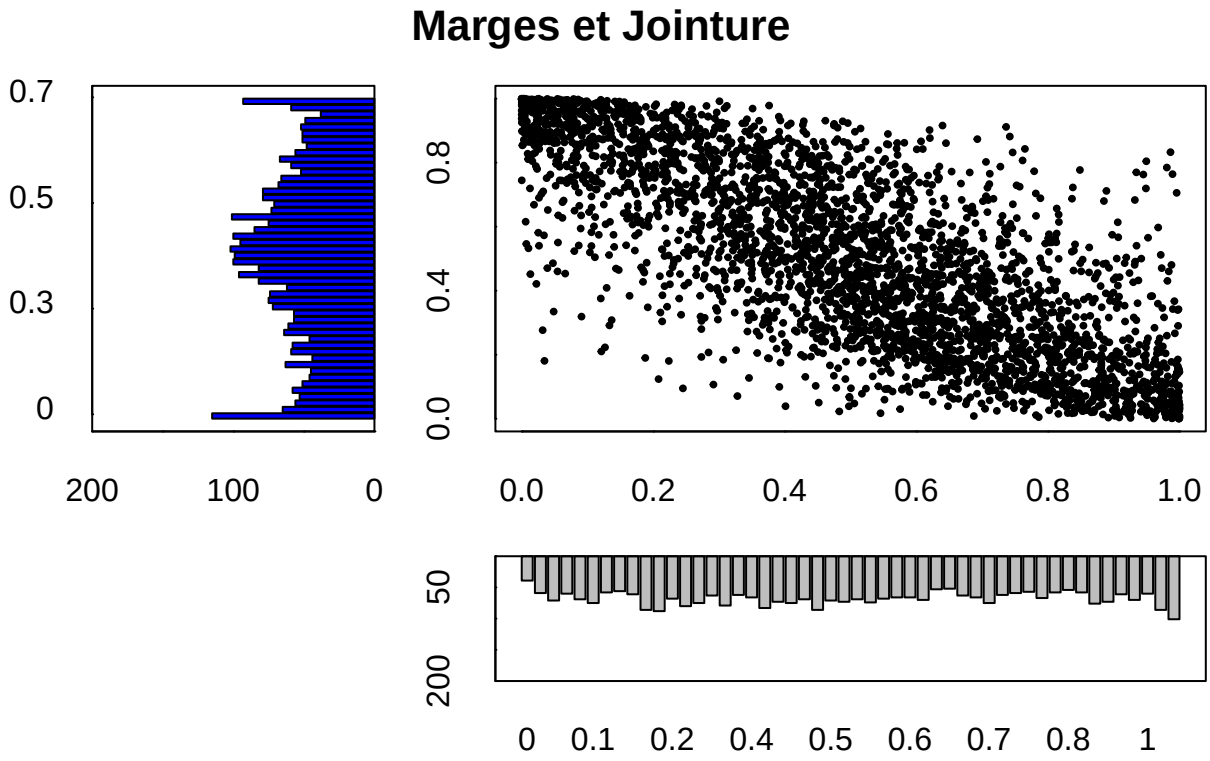
Nous utiliserons une Copule Gaussienne avec des marges de distribution *student-t* :

- $\nu = 10$ pour les rendements logarithmiques du SP500
- $\nu = 5$ pour les rendements logarithmiques du VIX.

Nous commencerons par calibrer la Copule Gaussienne.

```r
# Définir les degrés de liberté
df_sp500 <- 10
df_vix   <- 5

transformed_sp500 <- f_transform_to_unif(data    = logrets_sp500,
                                         density = "student",
                                         df      = df_sp500)

transformed_vix <- f_transform_to_unif(data    = logrets_vix,
                                       density = "student",
```

```
                                    df      = df_vix)

u_sp500 <- unlist(transformed_sp500[1])
u_vix <- unlist(transformed_vix[1])

params_sp500 <- unlist(transformed_sp500[2])
params_vix <- unlist(transformed_vix[2])

u_data  <- cbind(u_sp500, u_vix)

# Visualisons maintenant la distribution des données transformées.

f_display_copula_margins(data   = u_data,
                         labels = c("SP500", "VIX"),
                         lim1   = 200,
                         lim2   = 200)
```



En transformant les données, nous avons pu identifier (visuellement, à confirmer par la suite) une corrélation négative entre nos nouvelles variables. Nous pouvons nous attendre à ce que le coefficient de corrélation soit négatif, comme le montre la relation généralement négative entre les données uniformes du VIX et du S&P 500, vue dans le nuage de points ci-dessus (figure en haut à droite).

Vérifions si nos données transformées sont effectivement uniformément distribuées :

```
ks.test(u_sp500, "punif")
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  u_sp500
## D = 0.048286, p-value = 2.497e-07
## alternative hypothesis: two-sided
```

```r
ks.test(u_vix, "punif")
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  u_vix
## D = 0.020339, p-value = 0.1191
## alternative hypothesis: two-sided
```

Puisque la valeur-p des données échantillonnées des rendements logarithmiques du S&P 500 est significativement inférieure à 5%, nous avons suffisamment de preuves pour dire que les données de l'échantillon uniforme ne proviennent pas d'une distribution Uniforme. Cependant, comme la valeur-p des données de l'échantillon uniforme des rendements logarithmiques du VIX est plus que le double de 5%, nous pouvons conclure qu'elle suit effectivement une distribution Uniforme.

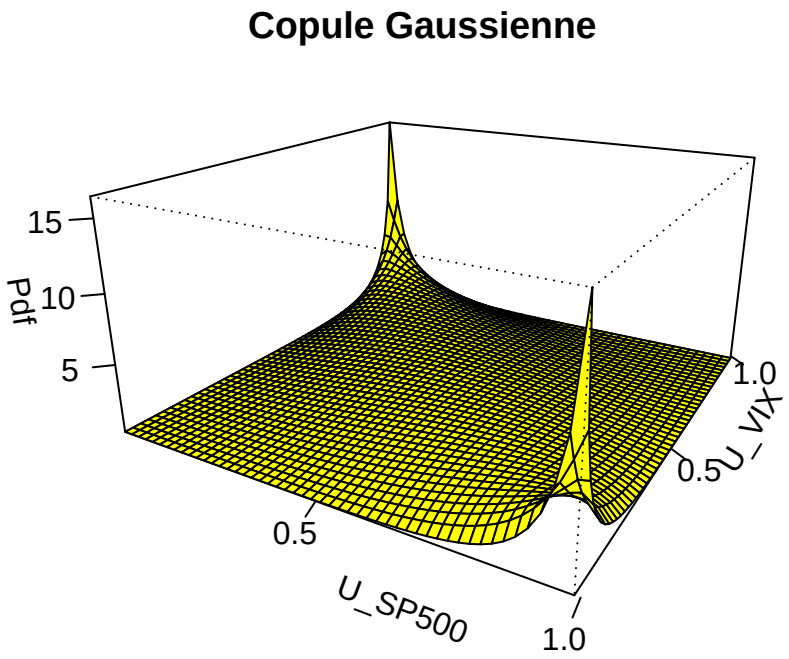
```r
# Définir la Copule Normale à deux dimensions
C <- normalCopula(dim = 2)

# Ajustement de la Copule à nos données d'échantillon uniformes
fit <- fitCopula(copula = C, data = u_data, method = "ml")
summary(fit)
```

```
## Call: fitCopula(C, data = u_data, ... = pairlist(method = "ml"))
## Fit based on "maximum likelihood" and 3409 2-dimensional observations.
## Normal copula, dim. d = 2
##       Estimate Std. Error
## rho.1  -0.7681      0.006
## The maximized loglikelihood is 1577
## Optimization converged
## Number of loglikelihood evaluations:
## function gradient
##       44       44
```

Le résumé ci-dessus confirme que la fonction d'optimisation par estimation du maximum de vraisemblance a convergé et que le coefficient de corrélation entre les rendements du VIX et du S&P 500 est $\rho = -0.7681$, ce qui confirme notre attente initiale mentionnée précédemment.

```r
# Extrayons maintenant le coefficient de corrélation de notre optimisation

rho <- coef(fit)[["rho.1"]]

Sigma <- f_copula_cov_mat(rho)
mu    <- c(mean_rets_sp500, mean_rets_vix) # vecteur de moyennes

# Obtenir la Copule Gaussienne
```

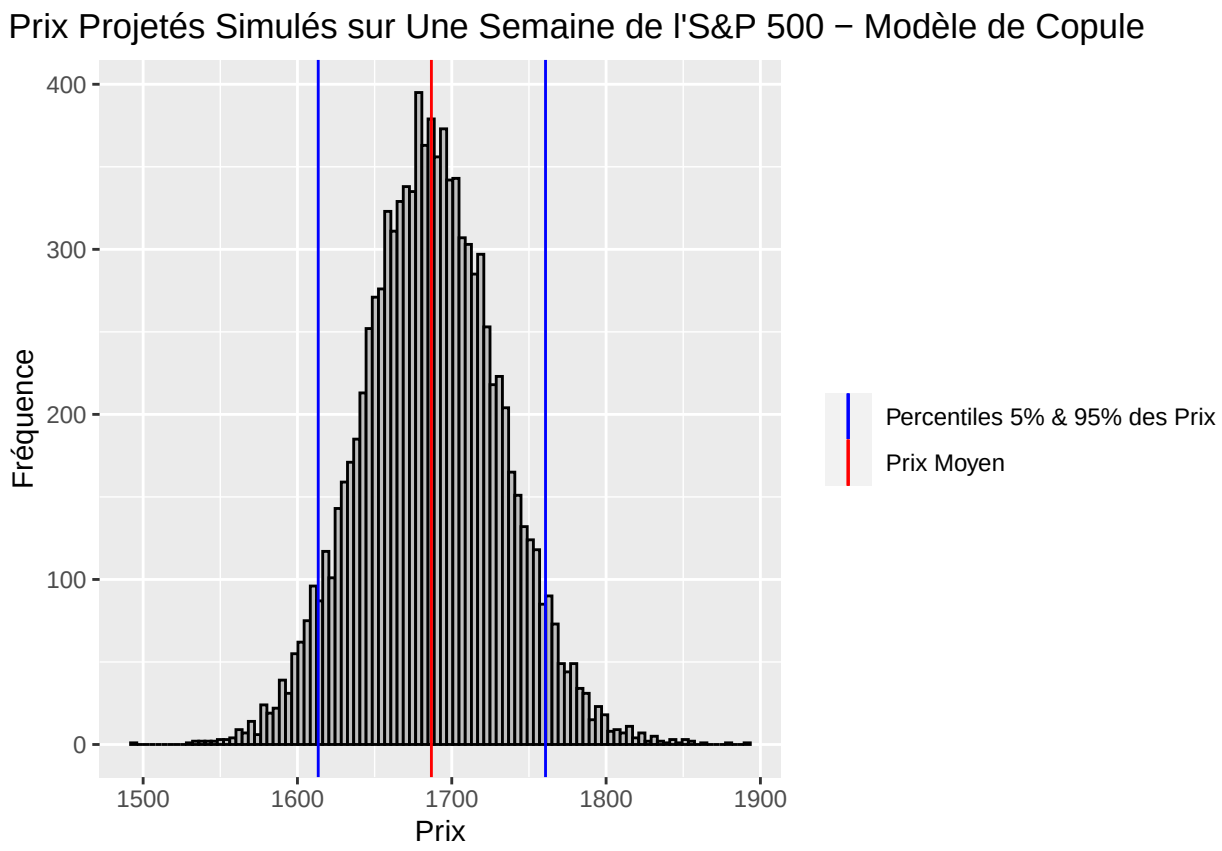
```r
f_my_copula <- function(w) {
  f_normal_copula_pdf(w, mu, Sigma)
}

grid_1 <- grid_2 <- seq(from = 0.01, to = 0.99, length.out = 50)

# Afficher la copule résultante
f_display_copula(f_my_copula, grid_1, grid_2)
title('Copule Gaussienne')
```

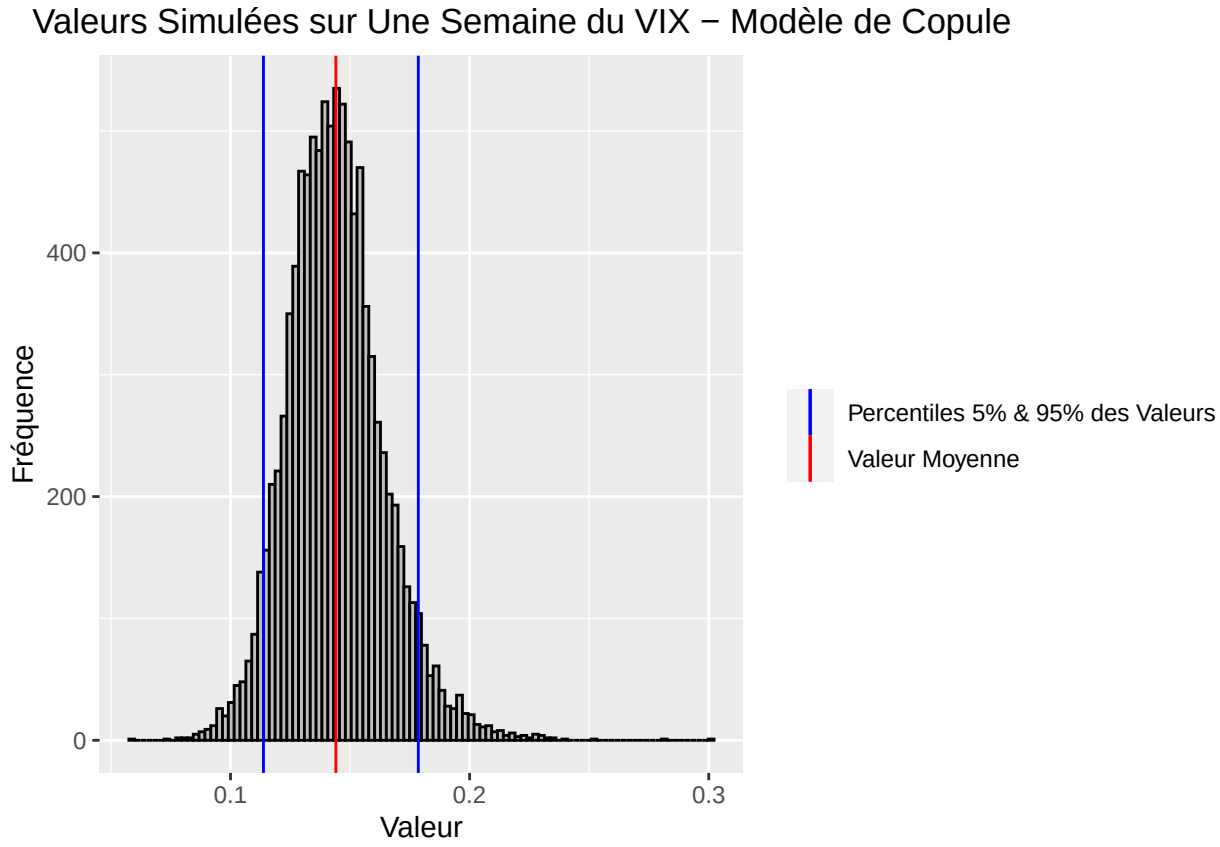


- Simuler les prix : Nous pouvons maintenant simuler les marginales et, par conséquent, calculer également les nouveaux prix du VIX et du S&P 500 sous cette distribution.

```r
set.seed(555) # pour la reproductibilité

sim_prices_sp500_Copula <- rep(NA, n_sim)  # Initialisation du vecteur pour stocker les prix simulés du

sim_prices_vix_Copula <- foreach(i = 1:n_sim, .combine = 'c') %do% {
  # Générer des échantillons pour la copule
  U_sim <- rCopula(n_days, fit@copula)
  # Transformer les tirages uniformément distribués des marginales en une distribution de Student
  logrets_sp500_sim   <- qstd(U_sim[,1],
                              mean = params_sp500[1],
                              sd   = params_sp500[2],
                              nu   = df_sp500)
```

```r
  logrets_vix_sim     <- qstd(U_sim[,2],
                              mean = params_vix[1],
                              sd   = params_vix[2],
                              nu   = df_vix)

  # Calculer les valeurs simulées du S&P 500
  sim_prices_sp500_Copula[i] <- exp(sum(logrets_sp500_sim)) * last_price
  # Calculer les valeurs simulées du VIX
  exp(sum(logrets_vix_sim)) * last_vix
}

# Visualiser les prix simulés pour le S&P 500 et le VIX en utilisant la copule
f_plot_sim_copula(prices = sim_prices_sp500_Copula,
                  vix    = sim_prices_vix_Copula)
```



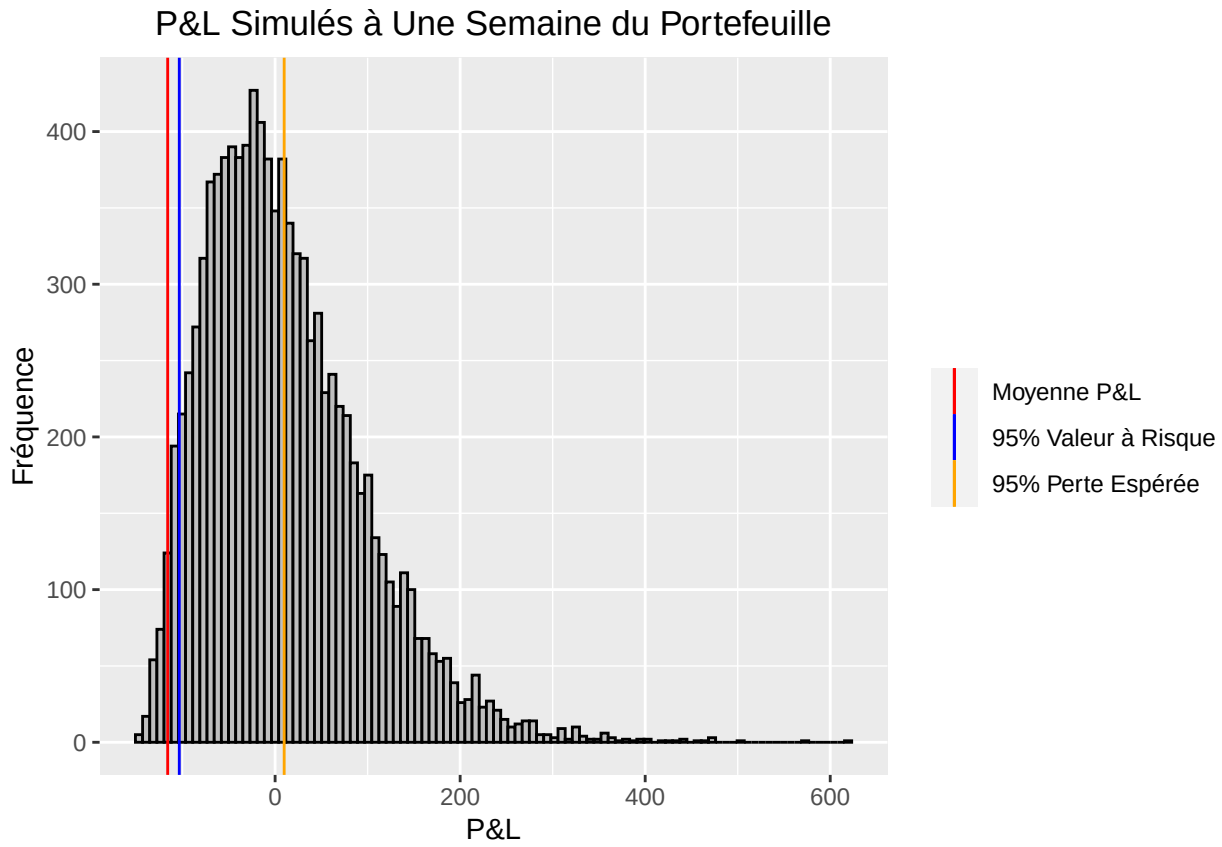


- Distribution des P&L (Profits et Pertes) :

```
# Obtention des nouvelles valeurs du portefeuille à partir des prix simulés sous la copule
new_book_values_Copula <- f_Get_Book_values(S = sim_prices_sp500_Copula,
                                            book = new_book,
                                            vol = sim_prices_vix_Copula)

# Calcul des profits et pertes
pnl_Copula <- new_book_values_Copula - init_book_value

# Calcul de la Valeur à Risque (VaR) et de la Perte Espérée (Expected Shortfall, ES) du portefeuille
VaR95_book_Copula <- f_VaR(pnl_Copula, level)
ES95_book_Copula  <- f_ES(pnl_Copula, level)
PnLstats["Copula Marginals"] <- c(VaR95_book_Copula, ES95_book_Copula)

PnLstats["Copula Marginals"]
```

```
##     Copula Marginals
## VaR        -103.5507
## ES         -116.0720
```

```
# Visualisation des résultats des P&L
f_Visualize_Portfolio_stats(data = pnl_Copula,
                            VaR = VaR95_book_Copula,
                            ES = ES95_book_Copula)
```



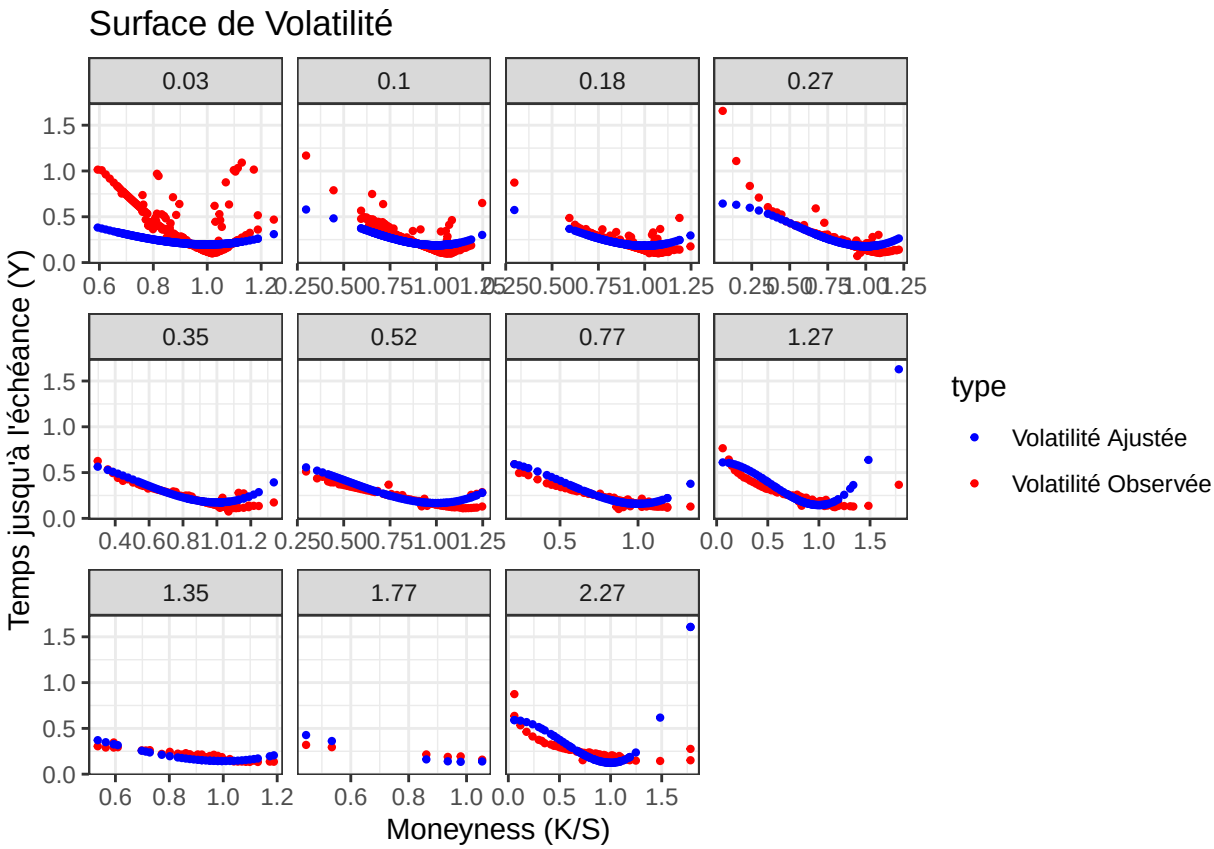
Nous pouvons remarquer que la VaR et la CVaR montre que les pertes sont plus faibles dans le cas de la distribution copule-marginal que dans le cas de la distribution gaussienne multivariée et de la gaussienne univariée. Cela est dû à la prise en compte de la corrélation négative entre les rendements de l'S&P 500 et du VIX, et à une meilleure modélisation du risque de queues en comparaison à la gaussienne multivariée.

```
## Partie 6 - Surface de volatilité
```

1. Ajustez une surface de volatilité aux volatilités implicites observées sur le marché (options d'achat et de vente négociées). Minimisez la distance absolue entre les volatilités implicites du marché et les volatilités implicites du modèle.

```
# Combinaison des options d'achat (calls) et de vente (puts) dans un seul dataframe
calls_and_puts <- rbind(calls, puts)

# Extraction des colonnes pertinentes
k         <- calls_and_puts[,1]  # Prix d'exercice
tau       <- calls_and_puts[,2]  # Temps jusqu'à l'échéance
market_IV <- calls_and_puts[,3]  # Volatilité implicite observée sur le marché

# Définition du prix actuel de l'action et calcul de la moneyness de chaque option
s <- last_price
m <- k / s  # Moneyness

alpha0 <- c(0,0,0,last_vix)  # Initialisation des paramètres du modèle [alpha1, alpha2, alpha3, alpha4]
# Minimiser la distance absolue entre les volatilités implicites du marché et celles du modèle
tmp <- optim(par       = alpha0,
```

```
              fn        = f_minimize_IV,
              m         = m,
              tau       = tau,
              market_IV = market_IV)
alpha <- tmp$par

# Calcul de la surface de volatilité implicite du modèle
model_IV <- f_parametric_surface(m     = m,
                                 tau   = tau,
                                 alpha = alpha)

# Tracer la surface de volatilité ajustée et observée
# Ajout de la volatilité implicite ajustée aux données des options
calls_and_puts$model_IV <- model_IV

# Création d'un dataframe vide pour stocker les données de tracé
plot_data <- data.frame()

# Boucle à travers les valeurs uniques de tau
unique_tau <- unique(calls_and_puts[, 2])
for (i in unique_tau) {
  unique_calls_puts <- calls_and_puts[tau == i, ]

  # Ajout de la volatilité implicite observée aux données de tracé
  plot_data <- rbind(plot_data, data.frame(moneyness = unique_calls_puts[, 1] / s,
                                           type = "Volatilité Observée",
                                           IV = unique_calls_puts[, 3],
                                           tau = i))

  # Ajout de la volatilité implicite ajustée aux données de tracé
  plot_data <- rbind(plot_data, data.frame(moneyness = unique_calls_puts[, 1] / s,
                                           type = "Volatilité Ajustée",
                                           IV = unique_calls_puts[, 4],
                                           tau = i))
}

# Création du graphique
ggplot(plot_data, aes(x = moneyness, y = IV, color = type)) +
  geom_point(size = 0.8) +
  scale_color_manual(values = c("Volatilité Observée" = "red", "Volatilité Ajustée" = "blue")) +
  facet_wrap(~ round(tau,2), scales = "free_x") +
  labs(title = "Surface de Volatilité", x = "Moneyness (K/S)", y = "Temps jusqu'à l'échéance (Y)") +
  theme_bw()
```



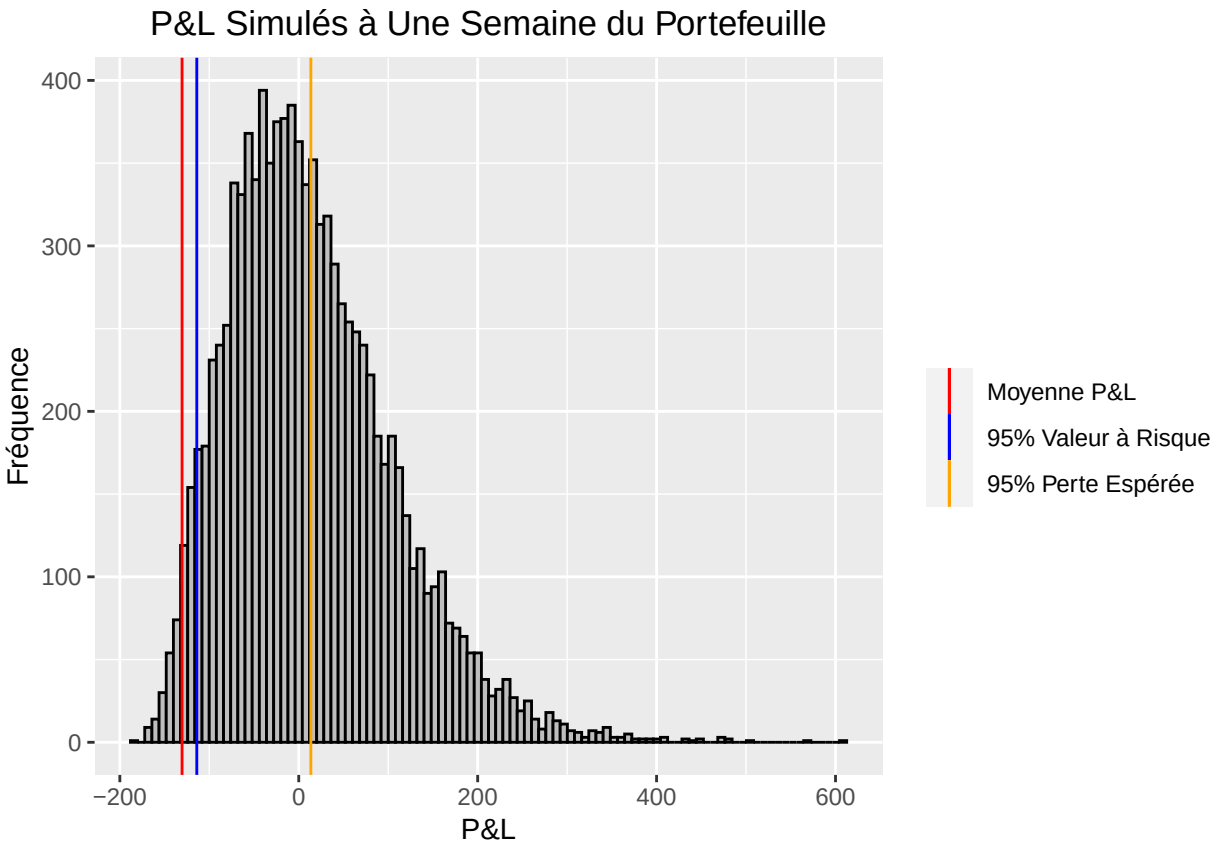
2. Réévaluez le portefeuille dans une semaine en supposant le même modèle paramétrique mais décalé de la différence de volatilité implicite de l'ATM à un an (i.e., vous gardez la différence entre le VIX et la volatilité implicite ATM à un an constante lors des simulations du VIX).

```
# Calibrer la volatilité implicite ajustée

vol_adj <- sim_prices_vix_Copula - (alpha[1] + alpha[4])  # Calcul de l'ajustement de la volatilité imp

# Calculer la moneyness pour chaque option
K_rep <- matrix(data = rep(new_book$K, each = n_sim),
                ncol = nrow(new_book))  # Répéter les prix d'exercice pour chaque simulation
m_sim <- K_rep / matrix(data = sim_prices_sp500_Copula,
                        nrow = n_sim,
                        ncol = nrow(new_book))  # Calculer la moneyness pour les prix simulés
tau_sim <- matrix(data = rep(new_book$tau, each = n_sim),
                  ncol = nrow(new_book))  # Répéter les temps jusqu'à maturité pour chaque simulation

# Calculer la surface de volatilité implicite du modèle pour chaque option
model_sim_IV <- f_parametric_surface(m = m_sim, tau = tau_sim, alpha)
model_sim_IV <- model_sim_IV  + vol_adj  # Ajouter la volatilité à la monnaie à la volatilité du modèle

# Re-tarifer chaque option du portefeuille
new_book_prices_IV <- matrix(NA, nrow = n_sim, ncol = ncol(new_book))

for (i in 1:ncol(new_book)) {
```

```
  new_book_prices_IV[,i] <- blackscholes(callput = 1,
                                         S0 = sim_prices_sp500_Copula,
                                         K = new_book$K[i],
                                         r = new_book$r[i],
                                         time = new_book$tau[i],
                                         vola = model_sim_IV[i,])$Price }

# Calculer la valeur totale de chaque portefeuille
new_book_values_IV <- f_Get_Book_values_VolSurf(S = sim_prices_sp500_Copula,
                                                book = new_book,
                                                vol = model_sim_IV,
                                                n_sim = n_sim)


# Pour comparer les P&L, nous devons estimer le prix de nos options au temps t avec
# la surface de volatilité paramétrique. On recalcule donc le prix de notre portefeuille initial

init_book_prices_IV <- matrix(NA, nrow = n_sim, ncol = ncol(new_book))
init_model_IV <- f_parametric_surface(m = init_book$S/init_book$K, tau = init_book$tau, alpha)


init_book_values_IV <- sum( blackscholes(callput = 1,
                                         S0 = init_book$S,
                                         K = init_book$K,
                                         r = init_book$r,
                                         time = init_book$tau,
                                         vola = init_model_IV)$Price )



# Calculer les P&L
pnl_IV <- new_book_values_IV  - init_book_values_IV

# Calculer la VaR95 et l'ES95
VaR95_IV <- f_VaR(pnl_IV, level)
ES95_IV <- f_ES(pnl_IV, level)
PnLstats["Surface de Volatilité"] <- c(VaR95_IV, ES95_IV)

PnLstats["Surface de Volatilité"]
```

```
##     Surface de Volatilité
## VaR             -113.9698
## ES              -130.4392
```

```
# Visualisation des résultats des P&L
f_Visualize_Portfolio_stats(data = pnl_IV,
                            VaR = VaR95_IV,
                            ES = ES95_IV)
```


P&L Simulés à Une Semaine du Portefeuille
Fréquence
400
300
200
100
0
−200
0
200
400
600
P&L
Moyenne P&L
95% Valeur à Risque
95% Perte Espérée