PREZENTACJA PROJEKTU

Grzegorz Kupeć, Natalia Słomka, Remigiusz Sroka, Adam Wawrzyńczyk

FUNKCJE ANALIZUJĄCE GIEŁDĘ - PLIK GLOBAL W APLIKACJI SHINY, ZWROTY

```
library(shiny)
library(readr)
library(ggplot2)
library(DT)

zwroty <- function(data, start_date, end_date, T_)
{
  nr_row_start_date <- which(data$Data == start_date)
  nr_row_end_date <- which(data$Data == end_date)
  dt <- T_/length(data$Data[nr_row_start_date:nr_row_end_date]) #deltat
  zwrot <- rep(0, times = (length(data$Data[nr_row_start_date:nr_row_end_date]) - 1))
  for (i in 1:length(zwrot))
  {
    zwrot[i] = (data$Zamkniecie[nr_row_start_date + i] - data$Zamkniecie[nr_row_start_date + i - 1])
    /data$Zamkniecie[nr_row_start_date + i - 1]
  }
  dryf <- mean(zwrot)/dt
  zmiennosc <- sqrt(1/dt)*sd(zwrot)
  return(list(zwroty, dryf, zmiennosc))
}
```

FUNKCJE ANALIZUJĄCE GIEŁDĘ - PLIK GLOBAL W APLIKACJI SHINY - MACIERZ

```
macierzAkcjaSim <- function(S_0, mu, sigma, n_days, n_sim, T_){
  mean = mu*(T_/n_days) - (sigma^2*(T_/n_days))/2
  sd = sigma*sqrt(T_/n_days)
  baz = rnorm(n_days*n_sim, mean, sd)
  tr = matrix(exp(baz),n_days, n_sim)
  tr = rbind(rep(1, times = n_sim), tr)
  tr = S_0 * apply(tr,2,cumprod)
  t(tr)
}
```

FUNKCJE ANALIZUJĄCE GIEŁDĘ - PLIK GLOBAL W APLIKACJI SHINY, WYKRES

```
#Plots

get_simulation <- function(x, tim, sim) {
  data.frame(t = tim, s = sim[x, ])
}

plot_sim <- function(n_sim, tim, sim) {
  p <- ggplot() + theme_bw()
  dat <- lapply(1:n_sim, function(x) get_simulation(x, tim, sim))
  for (i in 1:n_sim) {
    p <- p + geom_line(
      data = dat[[i]],
      mapping = aes(x = t, y = s),
      col = "black",
      alpha = 0.25
    )
  }
  return(p)
}
```

FUNKCJE ANALIZUJĄCE GIEŁDĘ - PLIK GLOBAL W APLIKACJI SHINY, WYKRES

```
plot_data <- function(data_path, n_sim, start_date, end_date,
input_date) {
  df <- read.csv(data_path)
  dryf <- zwroty(df, start_date, end_date, 1)[[2]]
  zmiennosc <- zwroty(df, start_date, end_date, 1)[[3]]
  stock_value <-
    as.numeric(df[which(df$Data == input_date), "Zamkniecie"])
  simulations <-
    macierzAkcjaSim(stock_value, dryf, zmiennosc,
                    length(as.Date(df$Data[which(df$Data ==
input_date):which(df$Data == end_date)])) - 1,
                    n_sim, 1)
  time <-
    as.Date(df$Data[which(df$Data == input_date):which(df$Data
== end_date)])
  p <- plot_sim(n_sim, tim = time, sim = simulations)
  plot1 <-
    p + geom_line(
      data = df[which(df$Data == start_date):which(df$Data ==
end_date),],
      mapping = aes(x = as.Date(Data),
                    y = Zamkniecie),
      col = "red"
    ) + labs(title = "Symulacje przyszŁ,ych trajektorii
WIG20",
             x = "Czas", y = "WartoŁ›Ä‡")
  return(plot1)
}
```

FUNKCJE ANALIZUJĄCE GIEŁDĘ - PLIK GLOBAL W APLIKACJI SHINY, WYKRES

```
plot_quantiles <- function(data_path, n_sim, start_date,
end_date, input_date){
  df <- read.csv(data_path)
  dryf <- zwroty(df, start_date, end_date, 1)[[2]]
  zmiennosc <- zwroty(df, start_date, end_date, 1)[[3]]
  stock_value <-
    as.numeric(df[which(df$Data == input_date), "Zamkniecie"])
  simulations <-
    macierzAkcjaSim(stock_value, dryf, zmiennosc,
length(as.Date(df$Data[which(df$Data == input_date):which(df
$Data == end_date)])) - 1, n_sim, 1)
  time <-
    as.Date(df$Data[which(df$Data == input_date):which(df$Data
== end_date)])
  p <- plot_sim(n_sim, tim = time, sim = simulations)
  p <-
    p + geom_line(
      data = df[which(df$Data == input_date):which(df$Data ==
end_date),],
      mapping = aes(x = as.Date(Data),
                    y = Zamkniecie),
      col = "red",
      size = 0.5
    )
  dat <- lapply(1:n_sim, function(x) get_simulation(x, time,
simulations))
  dat_all <- do.call(rbind, dat)
  quant <- data.frame(time)
  for (i in 1:250) {
    quant[i, "q.0.1"] <-
      quantile(dat_all[which(dat_all$t == time[i]), "s"],
probs = 0.1, names = FALSE)
```

FUNKCJE ANALIZUJĄCE GIEŁDĘ - PLIK GLOBAL W APLIKACJI SHINY, WYKRES

```
probs = 0.1, names = FALSE)
    quant[i, "q.0.25"] <-
      quantile(dat_all[which(dat_all$t == time[i]), "s"],
probs = 0.25, names = FALSE)
    quant[i, "q.0.5"] <-
      quantile(dat_all[which(dat_all$t == time[i]), "s"],
probs = 0.5, names = FALSE)
    quant[i, "q.0.75"] <-
      quantile(dat_all[which(dat_all$t == time[i]), "s"],
probs = 0.75, names = FALSE)
    quant[i, "q.0.9"] <-
      quantile(dat_all[which(dat_all$t == time[i]), "s"],
probs = 0.9, names = FALSE)
  }

  p <-
    p + geom_line(quant,
                  mapping = aes(x = time, y = q.0.1, colour =
"blue"),
                  size = 1) +
    geom_line(quant,
              mapping = aes(x = time, y = q.0.25, colour =
"gold"),
              size = 1) +
    geom_line(
      quant,
      mapping = aes(x = time, y = q.0.5, colour =
"chartreuse3"),
      size = 1
    ) +
    geom_line(quant,
              mapping = aes(x = time, y = q.0.75, colour =
```

FUNKCJE ANALIZUJĄCE GIEŁDĘ - PLIK GLOBAL W APLIKACJI SHINY, WYKRES

```
    geom_line(quant,
              mapping = aes(x = time, y = q.0.75, colour =
"gold"),
              size = 1) +
    geom_line(quant,
              mapping = aes(x = time, y = q.0.9, colour =
"blue"),
              size = 1)

  p <-
    p + labs(title = "Kwantyle symulacji przyszĹ‚ych
trajektorii WIG20", x = "czas", y = "wartoĹ›Ä‡") +
    scale_colour_manual(
      name = 'PrzedziaĹ‚y kwantylowe',
      breaks = c('gold', 'chartreuse3', "blue"),
      values = c('gold', 'chartreuse3', "blue"),
      labels = c("25% - 75%", '50%', '10% - 90%')
    )
  return(p)
}
```

FUNKCJE ANALIZUJĄCE GIEŁDĘ - PLIK GLOBAL W APLIKACJI SHINY, WSKAŹNIKI GRECKIE

```
vanillaDelta <- function(spot, strike, r, sigma, t, T_,
payoffType){
  if(payoffType == 'call'){
    return(pnorm(d1(spot, strike, r, sigma, t, T_)))
  }
  else{
    return(vanillaDelta(spot, strike, r, sigma, t, T_,
'call') - 1)
  }
}

d1 <- function(spot, strike, r, sigma, t, T_){
  asd = log(spot / strike)  + (r + sigma * sigma / 2) * (T_ -
t)
  asd = asd / (sigma * sqrt(T_ - t))
  asd
}

d2 <- function(spot, strike, r, sigma, t, T_){
  return (d1(spot, strike, r, sigma, t, T_) - sigma *
sqrt(T_ - t))
}

Delta <- function(spot, strike, r, sigma, t, T_, payoffType){
  if(payoffType == 'call'){
    return(pnorm(d1(spot, strike, r, sigma, t, T_)))
  }
  if(payoffType == 'put'){
    return(vanillaDelta(spot, strike, r, sigma, t, T_,
'call') - 1)
  }
}
```

FUNKCJE ANALIZUJĄCE GIEŁDĘ - PLIK GLOBAL W APLIKACJI SHINY, WSKAŹNIKI GRECKIE

```
Gamma <- function(spot, strike, r, sigma, t, T_, payoffType) {
  return(dnorm(d2(spot, strike, r, sigma, t, T_)) / (spot *
sigma * sqrt(T_ - t)))
}


Theta <- function(spot, strike, r, sigma, t, T_, payoffType){
  d1 = d1(spot, strike, r, sigma, t,T_)
  d2 = d2(spot, strike, r, sigma, t,T_)
  if(payoffType == 'call'){
    return(-sigma * spot *pnorm(d1)/(2 * sqrt(T_ - t))-r *
strike* exp(-(T_-t)*r)*dnorm(d2))}
  if(payoffType == 'put'){
    return(-sigma * spot *pnorm(-d1)/(2 * sqrt(T_ - t))+r *
strike* exp(-(T_-t)*r)*dnorm(-d2))}
}

Vega <- function(spot, strike, r, sigma, t, T_, payoffType){
  d1 = d1(spot, strike, r, sigma, t,T_)
  return(spot * sqrt(T_ - t) * pnorm(d1))
}

Rho <- function(spot, strike, r, sigma, t, T_, payoffType){
  d1 = d1(spot, strike, r, sigma, t,T_)
  d2 = d2(spot, strike, r, sigma, t,T_)
  if(payoffType == 'call'){
    return(spot * (T_ - t) * sigma * exp(-(T_-t)*r) *
dnorm(d2))}
  if(payoffType == 'put'){
    return(-spot * (T_ - t) * sigma * exp(-(T_-t)*r) * dnorm(-
d2))}
}
```

FUNKCJE ANALIZUJĄCE GIEŁDĘ - PLIK GLOBAL W APLIKACJI SHINY, WSKAŹNIKI GRECKIE

```
#cena
vanillaPrice <- function(spot, strike, r, sigma, t, T_,
payoffType){
  discountFactor = exp(-r * (T_ - t))
  d1 = d1(spot, strike, r, sigma, t,T_)
  d2 = d2(spot, strike, r, sigma, t,T_)
  if(payoffType == 'call'){
    return(spot * pnorm(d1) - strike * discountFactor *
pnorm(d2))}
  if(payoffType == 'put'){
    return(- spot * pnorm(-d1) + strike * discountFactor *
pnorm(-d2))}
}

computeGreeks <- function(spot, strike, r, sigma, t, T_){
  greeks <- data.frame(1:6, 1:6)
  colnames(greeks) <- c("Call", "Put")
  rownames(greeks) <- c("vanilla Delta", "Delta", "Gamma",
"Theta", "Rho", "vanilla Price")
  greeks[, "Call"] <- c(vanillaDelta(spot, strike, r, sigma,
t, T_, "call"),
                        Delta(spot, strike, r, sigma, t,
T_, "call"),
                        Gamma(spot, strike, r, sigma, t,
T_, "call"),
                        Theta(spot, strike, r, sigma, t,
T_, "call"),
                        Rho(spot, strike, r, sigma, t,
T_, "call"),
                        vanillaPrice(spot, strike, r,
sigma, t, T_, "call")
                        )
```

FUNKCJE ANALIZUJĄCE GIEŁDĘ - PLIK GLOBAL W APLIKACJI SHINY, WSKAŹNIKI GRECKIE

```
  greeks[, "Put"] <- c(vanillaDelta(spot, strike, r, sigma, t,
T_, "put"),
                       Delta(spot, strike, r, sigma, t,
T_, "put"),
                       Gamma(spot, strike, r, sigma, t,
T_, "put"),
                       Theta(spot, strike, r, sigma, t,
T_, "put"),
                       Rho(spot, strike, r, sigma, t,
T_, "put"),
                       vanillaPrice(spot, strike, r,
sigma, t, T_, "put")
                      )
  return(greeks)
}
```

PLIK SERVER W APLIKACJI SHINY

```
server <- function(input, output) {
  output$contents <- renderTable({
    req(input$data)
    tryCatch({
      df <- read.csv(input$data$datapath)
    },
    error = function(e) {
      # return a safeError if a parsing error occurs
      stop(safeError(e))
    })

    if (input$disp == "head") {
      return(head(df))
    }
    else {
      return(df)
    }

  })
```

PLIK SERVER W APLIKACJI SHINY

```
output$greeks <- renderDT({
  req(input$data)
  tryCatch({
    df <- read.csv(input$data$datapath)
  })
  computeGreeks(
    spot = input$spot,
    strike = input$strike,
    r = input$interest_rate,
    sigma = zwroty(
      data = df,
      start_date = as.character(input$our_start_date),
      end_date = as.character(input$our_end_date),
      T_ = 1
    )[[3]],
    0,
    1
  )
})
```

PLIK SERVER W APLIKACJI SHINY

```
  output$plot <- renderPlot({
    #req(input$data)
    tryCatch({
      df <- read.csv(input$data$datapath)
    })
    plot_data(
      data_path = input$data$datapath,
      n_sim = input$nsim,
      start_date = as.character(input$our_start_date),
      end_date = as.character(input$our_end_date),
      input_date = as.character(input$our_input_date)
    )
  })

  output$plot_future <- renderPlot({
    req(input$data)
    tryCatch({
      df <- read.csv(input$data$datapath)
    })
    plot_quantiles(
      data_path = input$data$datapath,
      n_sim = input$nsim,
      start_date = as.character(input$our_start_date),
      end_date = as.character(input$our_end_date),
      input_date = as.character(input$our_input_date)
    )
  })
}
```

PLIK UI W APLIKACJI SHINY

```
fluidRow(titlePanel("PodglÄ…d danych"),
         tableOutput("contents")))

ui <- navbarPage(
  "Projekt - dane gieĹ‚dowe",
  tabPanel(
    "Wczytywanie danych",
    sidebarLayout(
      sidebarPanel(
        fileInput(
          "data",
          "WybÄ‚Ĺr pliku CSV",
          multiple = FALSE,
          accept = c("text/csv",
                     "text/comma-separated-values,text/plain",
                     ".csv")
        ),
        tags$hr(),
        tags$hr(),
        radioButtons(
          "disp",
          "WyĹ›wietl",
          choices = c(Head = "head",
                      All = "all"),
          selected = "head"
        ),
        #poniĹĽej inputy dla dat
```

PLIK UI W APLIKACJI SHINY

```
dateInput(
  "our_start_date",
  "ProszÄ™ podaÄ‡ datÄ™ poczÄ…tku wykresu",
  value = NULL,
  min = NULL,
  max = NULL,
  format = "yyyy-mm-dd",
  startview = "month",
  weekstart = 0,
  language = "pl",
  width = NULL
),

dateInput(
  "our_end_date",
  "ProszÄ™ podaÄ‡ datÄ™ poczÄ…tku symulacji",
  value = NULL,
  min = NULL,
  max = NULL,
  format = "yyyy-mm-dd",
  startview = "month",
  weekstart = 0,
  language = "pl",
  width = NULL
),
```

PLIK UI W APLIKACJI SHINY

```
            dateInput(
              "our_input_date",
              "ProszÄ™ podaÄ‡ datÄ™ koŁ„ca symulacji",
              value = NULL,
              min = NULL,
              max = NULL,
              format = "yyyy-mm-dd",
              startview = "month",
              weekstart = 0,
              language = "pl",
              width = NULL
            )

          ),
          mainPanel(tableOutput("contents"))
        )
      ),
      tabPanel("Wykres",
               sidebarLayout(
                 sidebarPanel(
                   numericInput(
                     "nsim",
                     "IloŁ›Ä‡ symulacji:",
                     min = 0,
                     value = 100
                   )
                 ),
                 mainPanel(plotOutput("plot"),
plotOutput("plot_future"))
               )),
```

PLIK UI W APLIKACJI SHINY

```
    tabPanel("WskaĹşniki greckie",
             sidebarLayout(
               sidebarPanel(
                 tags$hr(),
                 sliderInput(
                   "interest_rate",
                   "Stopa procentowa:",
                   min = 0,
                   max = 0.1,
                   value = 0.05
                 ),
                 numericInput(
                   "strike",
                   "WartoĹ›Ä‡ strike:",
                   value = 0
                 ),
                 numericInput(
                   "spot",
                   "WartoĹ›Ä‡ spot:",
                   value = 0
                 ),
                 tags$hr()
               ),
               mainPanel(DTOutput("greeks"))
             ))

  )
```

KONIEC