FE8828 Programming Web Applications in Finance

Group Assignment

Dr. Yang Ye yy@runchee.com

Sep 24, 2019

1. Bank

```
Data frame 1: Account
| AcountNo | Name |

Data frame 2: Transaction
| TransactionNo | Date | AccountNo | TransactionType | Amount | Currency |

Data frame 3: Currency to SGD
| Currency | Conversion | Date |
```

TransactionType can be: Withdraw/Deposit/Spend Write follow functions and combine them to form a small program

1. Create 10 accounts with initial random deposit and credit in SGD.
2. Create 3 currencies: CNY, USD, SGD. Download their conversion rate between 2018-07-01 and 2018-09-30.
3. Generate random transaction data for 10 accounts during 2018-07-01 and 2018-09-30. Make it more realistic, deposit is 1-2 times per month, a random number of 3000-5000, any of three currencies. Spend/Withdraw can be any times [0, 60] and any amount, any currencies. Deposit is positive, Withdraw/Spend is negative. Constraint: You can't withdraw more than the deposit, can't spend more than credit + deposit.
4. Generate report for transaction as month-end statement in SGD.

```
{ Client Name }
{ Month }

# Transaction History

Date | TransactionType | Amount | Currency | Deposit Balance | Credit Balance |
     |                 |        |          |                 |                |
     |                 |        |          |                 |                |
     |                 |        |          |                 |                |
-------------------------------------------------------------------------------
                        Month-End Balance | Deposit         | Credit         |

# Summary

TransactionType | Amount |
...
```

Submission:

R Markdown document, containing:

1. describing design
2. Code and explaination of result
3. Example running result.

2. Option trading - dynamic hedging

Financial derivatives like Options are named so because their value is derived from its underlyings. Greeks are the sensitivity of the price of the derivaties to its underlying parameters. For example, a vanilla European option depends on S (Underlying price), stri$K$e, Vol(atility), r (interest rate), q (cost of carry), T (time).

Greeks:

- $\frac{dV}{dS}$: Delta
- $\frac{d^2V}{dS^2}$: Gamma
- $\frac{dV}{dVol}$: Vega
- $\frac{dV}{dT}$: Theta

To simplify, we set $r = q = 0$ (so Rho is zero).

The daily PnL of option can be approximated into, according to Talyor expansion $\Delta V = Delta * \Delta S + 0.5 * Gamma * \Delta S^2 + Vega * \Delta Vol + Theta * (dT) + SmallRemainder$

Option trading with dynamtic hedging uses two positions: one position in option and one position in underlying. The position on option is established at the beginning of the trade. The position in underlying is adjusted periodically (e.g. daily) to the opposite of the option delta so the strategy-delte is zero. The objective is to realize the difference between realized volatility and implied volatility.

For example, there is a 30-day ATM call option, K = S = 100, r = q = 0, vol = 0.3 constant till option expiry.

```
days <- 20.0
vega <- GBSGreeks("Vega", "c", S = 100, X = 100, Time = days / 250, r = 0, b = 0, sigma = 0.3)
delta <- GBSGreeks("Delta", "c", S = 100, X = 100, Time = days / 250, r = 0, b = 0, sigma = 0.3)
gamma <- GBSGreeks("Gamma", "c", S = 100, X = 100, Time = days / 250, r = 0, b = 0, sigma = 0.3)
theta <- GBSGreeks("Theta", "c", S = 100, X = 100, Time = days / 250, r = 0, b = 0, sigma = 0.3)
price <- GBSOption("c", S = 100, X = 100, Time = days / 250, r = 0, b = 0, sigma = 0.3)@price
```

If on day 1, S1 = 98, Volatility is unchanged.

```
S1 <- 98
price1 <- GBSOption("c", S = S1, X = 100, Time = (days - 1) / 250, r = 0, b = 0, sigma = 0.3)@price
approx <- delta * (S1 - 100) + 0.5 * gamma * (S1 - 100) ** 2 + vega * 0 + theta * 1/250
cat(paste0("price1 - price: ", price1 - price, ", ", approx, "\n"))
```

```
## price1 - price: -1.02175324971442, -1.02444665039741
```

```
cat(paste0("diff with approx.: ", price1 - price - approx, "\n"))
```

```
## diff with approx.: 0.00269340068298818
```

Analytically, we can prove the following:

[1] If realized volatility and implied volatility are the same, Gamma and Theta can neutralize each other, i.e. $0.5 * Gamma * \Delta S^2 + Theta * (dT) \approx 0$

[2] And overall Profit and Loss "overall PnL" for the strategy is $\approx Vega * (RealizedVol - ImpliedVol) * T$. The overall PnL needs to count for the cost of buying the option.

Your task is to use simulation to prove above two points with the above-mentioned 30-day ATM call option. Note: that simulation would show some acceptable level of difference.

```
S <- 100
K <- 100
sigma <- 0.5 # realized vol can be 0.3 for [1] or 0.5 for [2]
drift <- 0 # drift = r - q
```

```
N <- days
timestep <- 1 / 250

p1 <- (drift - 0.5 * sigma * sigma) * timestep
p2 <- sigma * sqrt(timestep)

# ss is the simulated price movement for N days
ss <- rep(S, N) * cumprod(rlnorm(N, mean = p1, sd = p2))

df <- data_frame(S = ss, days = 1:days)
opt <- mutate(df,
        price = GBSOption("c", S = S, X = 100, Time = days / 250, r = 0, b = 0, sigma = 0.3)@price,
        delta = GBSGreeks("Delta", "c", S = S, X = 100, Time = days / 250, r = 0, b = 0, sigma = 0.3),
        gamma = ...
        theta = ...
        # vega: vol is unchagned so we can skip
      )

...
```

---

3. Option Delta Hedging (Real-world)

Show case how delta hedging works as a trading strategy.

- Show how one trade works
- Backtest for available history - one-year history from the website.

Assumption:

- You hold 100 ATM call/put option which expires in 30 days (calendar days). You just need to do either Call or Put. After finishing the tasks below, duplicate the R Notebook and carry out 2nd-round analysis using an option of ~25% delta, take a guess for the strike level.
- You start to do delta hedging daily immediately till 2nd last day. You close stock position in the last day. Option expires on the last day as well (either ITM with positive PnL or ATM/OTM with zero PnL)
- Delta hedging: calculate the delta from option, negate it, that's the quantity what you need to hold over 1 day. Repeat for every trading day.
- Daily PnL: (option premium change) + (stock holding quantity * price change).
- You can get your favorite stocks here. There is one year of data.
  - <https://marketchameleon.com/Overview/*Stock Code*/DailyHistory/>
  - e.g.: https://marketchameleon.com/Overview/GS/DailyHistory/
- Daily IV30 is provided, IV 30 is implied volatility for 30 days. We flat-extrapolate it to be for < 30 days.
- As underlying is equity, dividend yield is applicable for B-S valuation
- US risk-free rate for 1M: 0.8% (annualized)

Analysis: For one trade

- Daily PnL v.s. Time to expiry: split into option and stock.
- Final PnL: accumulative of Daily PnL. split into option and stock
- Max Drawdown: accumulative of Daily PnL, max - min.

- Sharpe ratio: Sharpe ratio = (Mean of Daily PnL - Risk-free rate)/Standard deviation of Daily PnL

Analysis: For backtest

- Distribution of Final PnL
- Distribution of Max Drawdown
- Final PnL v.s. Option Expiry Date
- ...
- More analysis
- in R Markdown

Example flow of analysi:

- Create xts object from the data from website.
- One trade analysis
  - Pick a date range using xts object.
    - Get starting date and end date.

```
dates <- index(xts_obj)
start_date <- min(dates)
end_date <- max(dates)
start_price <- xts_obj[start_date, "Close"]
start_volatility <- xts_obj[start_date, "IV30"]
```

    - Create a df with date column

```
df <- tibble(date = dates)
df$Close <- coredata(xts_obj[, "Close"])
```

  - Daily Profit and Loss ("DoD PnL")
    - Option side:

```
X <- start_price
sigma = start_volatility
r <- 0.8 / 100
# Vary S and Time everyday
S <- Close
Time <- (end_date - date) / 365
GBSOption(TypeFlag, S, X, Time, r, b, sigma)@price

df_opt <- rowwise(df) %>%
          mutate(premium = GBSOption(TypeFlag = "...",
                                     S = Close,
                                     X = start_price,
                                     Time = (end_date - date) / 365,
                                     r = ..., # interest rate
                                     b = ..., # dividend yield
                                     sigma = start_volatility)@price) %>%
          ungroup %>%
          mutate(Option_DoD_PnL = ifelse(date == start_date,
          # On the 1st date, we count the cost of buying the option
                                         premium * (-1),
                                         premium - lag(premium)))
```

    - Hedging side:

```
rowwise() %>%
mutate(delta_hedge = GBSGreeks("delta", TypeFlag, S, X, Time, r, b, sigma) *
                     quantity * (-1)) %>%
ungroup() %>%
mutate(Hedging_DoD_Pnl = ifelse(date == start_date,
                                0,
```

```
                                        delta_hedge * (Close - lag(Close))))
```

  - Daily PnL (combined):

```
mutate(DoD_PnL = Option_DoD_PnL + Hedging_DoD_Pnl)
```

- Max Drawdown: accumulative of Daily PnL, max - min.

```
ungroup() %>%
mutate(PnL = cumsum(DoD_PnL)) %>%
{
  xs <- .$PnL
  max(cummax(xs) - cummin(xs))
}
```