FE621 - Homework 1

James G. Tenreiro

February 15, 2026

Part 1. Data gathering component

1.

Equity history is pulled with getSymbols and normalized into a simple OHLCV table. Option snapshots are pulled from Yahoo's options endpoint by selecting the next three standard monthlies (third Fridays) and flattening each chain into contract rows with the quote timestamp and underlying from the returned JSON. Mid is (bid+ask)/2 only when both quotes are positive and volume is positive; otherwise last is used when available. The risk free rate is taken from external effective fed funds data (FRED DFF) on the snapshot date and converted to a continuously compounded annual rate.

```
# SECTION 1: RUN ONCE
options(stringsAsFactors=FALSE)
pkgs<-c("httr","jsonlite","quantmod","readr","writexl","zoo")
need<-pkgs[!pkgs%in%rownames(installed.packages())]
if(length(need)) install.packages(need,repos="https://cloud.r-project.org")
invisible(lapply(pkgs,library,character.only=TRUE))

is_third_friday<-function(d){lt<-as.POSIXlt(d); lt$wday==5&&lt$mday>=15&&lt$mday<=21}
next_three_monthlies<-function(exp_dates,as_of){x<-sort(unique(as.Date(exp_dates)));
 ↪  x<-x[x>=as_of]; x<-x[vapply(x,is_third_friday,logical(1))]; head(x,3)}
mid<-function(bid,ask,volume,last){if(!is.na(bid)&&!is.na(ask)&&bid>0&&ask>0&&!is.na
 ↪  (volume)&&volume>0) return((bid+ask)/2); if(!is.na(last)&&last>0) return(last);
 ↪  NA_real_}

eqdl<-function(symbol,from,to){
  xt<-quantmod::getSymbols(symbol,src="yahoo",from=from,to=to,auto.assign=FALSE)
  data.frame(time=as.POSIXct(zoo::index(xt)),
             open=as.numeric(quantmod::Op(xt)),
             high=as.numeric(quantmod::Hi(xt)),
             low=as.numeric(quantmod::Lo(xt)),
             close=as.numeric(quantmod::Cl(xt)),
             volume=as.numeric(quantmod::Vo(xt)),
             adjusted=as.numeric(quantmod::Ad(xt)),
             symbol=symbol,stringsAsFactors=FALSE)
}

opt_schema<-function(){
  data.frame(symbol=character(),option_type=character(),contract=character(),expiration=
   ↪  as.Date(character()),
             strike=double(),bid=double(),ask=double(),last=double(),volume=double(),
              ↪  open_interest=double(),
```

```r
            in_the_money=logical(),underlying_price=double(),snapshot_time=as.POSIXct(character()),mid=double(),
            stringsAsFactors=FALSE)
}

yahoo_auth<-local({
  h<-httr::handle("https://yahoo.com"); cr<-NULL
  function(refresh=FALSE){
    if(!is.null(cr)&&!refresh) return(list(handle=h,crumb=cr))
    httr::GET("https://fc.yahoo.com",httr::user_agent("Mozilla/5.0"),handle=h)
    r<-httr::GET("https://query1.finance.yahoo.com/v1/test/getcrumb",httr::user_agent("Mozilla/5.0"),handle=h)
    if(httr::status_code(r)!=200) stop("crumb HTTP ",httr::status_code(r))
    cr<<-httr::content(r,"text",encoding="UTF-8")
    list(handle=h,crumb=cr)
  }
})

yopt_url<-function(symbol,epoch=NULL,crumb=NULL){
  s<-URLencode(symbol,reserved=TRUE)
  base<-paste0("https://query2.finance.yahoo.com/v7/finance/options/",s)
  if(is.null(epoch)) paste0(base,"?crumb=",crumb) else paste0(base,"?date=",as.integer(epoch),"&crumb=",crumb)
}

getj<-function(symbol,epoch=NULL){
  a<-yahoo_auth(); u<-yopt_url(symbol,epoch,a$crumb)
  r<-httr::GET(u,httr::user_agent("Mozilla/5.0"),handle=a$handle,
               httr::add_headers("Accept"="application/json,text/plain,*/*","Accept-Language"="en-US,en;q=0.9"))
  if(httr::status_code(r)==401){
    a<-yahoo_auth(refresh=TRUE); u<-yopt_url(symbol,epoch,a$crumb)
    r<-httr::GET(u,httr::user_agent("Mozilla/5.0"),handle=a$handle,
                 httr::add_headers("Accept"="application/json,text/plain,*/*","Accept-Language"="en-US,en;q=0.9"))
  }
  if(httr::status_code(r)!=200) stop("HTTP ",httr::status_code(r)," for ",symbol)
  txt<-trimws(httr::content(r,"text",encoding="UTF-8"))
  if(nchar(txt)==0) stop("Empty response for ",symbol)
  if(substr(txt,1,1)!="{") stop("Non-JSON response for ",symbol,": ",substr(txt,1,min(200,nchar(txt))))
  jsonlite::fromJSON(txt,simplifyVector=FALSE)
}

available_expirations<-function(symbol){
  js<-getj(symbol); res<-js$optionChain$result
  if(is.null(res)||length(res)==0) return(as.Date(character(0)))
  e<-res[[1]]$expirationDates
  if(is.null(e)||length(e)==0) return(as.Date(character(0)))
  as.Date(as.POSIXct(unlist(e),origin="1970-01-01",tz="UTC"))
}

pick1<-function(x){if(is.null(x)||length(x)==0) NA else x[[1]]}

contracts_to_df<-function(side,symbol,type,S,t){
```

```
  if(is.null(side)||length(side)==0) return(NULL)
  rows<-lapply(side,function(cn){
    data.frame(
      symbol=symbol,
      option_type=type,
      contract=as.character(pick1(cn$contractSymbol)),
      expiration=as.Date(as.POSIXct(as.numeric(pick1(cn$expiration)),origin="1970-01-01"
       ↪ ,tz="UTC")),
      strike=as.numeric(pick1(cn$strike)),
      bid=as.numeric(pick1(cn$bid)),
      ask=as.numeric(pick1(cn$ask)),
      last=as.numeric(pick1(cn$lastPrice)),
      volume=as.numeric(pick1(cn$volume)),
      open_interest=as.numeric(pick1(cn$openInterest)),
      in_the_money=as.logical(pick1(cn$inTheMoney)),
      underlying_price=as.numeric(S),
      snapshot_time=as.POSIXct(t,origin="1970-01-01",tz="UTC"),
      stringsAsFactors=FALSE
    )
  })
  out<-do.call(rbind,rows)
  out$mid<-mapply(mid,out$bid,out$ask,out$volume,out$last)
  out
}

parse_chain<-function(js,symbol){
  res<-js$optionChain$result[[1]]; q<-res$quote; opt<-res$options[[1]]
  t<-as.POSIXct(q$regularMarketTime,origin="1970-01-01",tz="UTC")
  S<-suppressWarnings(as.numeric(q$regularMarketPrice))
  calls<-contracts_to_df(opt$calls,symbol,"call",S,t)
  puts <-contracts_to_df(opt$puts ,symbol,"put" ,S,t)
  out<-rbind(calls,puts)
  if(is.null(out)||nrow(out)==0) return(opt_schema())
  out[order(out$symbol,out$expiration,out$option_type,out$strike),]
}

download_options_3m<-function(symbol,as_of){
  ex<-next_three_monthlies(available_expirations(symbol),as_of)
  if(length(ex)==0) return(opt_schema())
  epochs<-as.integer(as.POSIXct(ex,tz="UTC"))
  pieces<-lapply(epochs,function(ep){
    Sys.sleep(0.4)
    tryCatch(parse_chain(getj(symbol,ep),symbol),
             error=function(e){message("OPTIONS ERROR ",symbol," epoch=",ep,":
              ↪ ",conditionMessage(e)); opt_schema()})
  })
  out<-do.call(rbind,pieces)
  if(is.null(out)||nrow(out)==0) opt_schema() else out
}

download_dataset<-function(dataset_name,symbols=c("TSLA","SPY","^VIX"),as_of=Sys.Date(),
↪ equity_from=as_of-10,equity_to=as_of){
  dir.create(dataset_name,showWarnings=FALSE,recursive=TRUE)
  snaps<-list(); eqs<-list(); opts<-list()
  for(sym in symbols){
```

```
    ed<-eqdl(sym,equity_from,equity_to); S0<-tail(ed$close[!is.na(ed$close)],1)
    od<-tryCatch(download_options_3m(sym,as_of),
                 error=function(e){message("SYMBOL ERROR ",sym,": ",conditionMessage(e));
                  ↪  opt_schema()})
    snaps[[sym]]<-data.frame(symbol=sym,downloaded_at=as.character(Sys.time()),
                             underlying_price_at_download=as.numeric(S0),
                              ↪  stringsAsFactors=FALSE)
    eqs[[sym]]<-ed; opts[[sym]]<-od
  }
  snap<-do.call(rbind,snaps); eq<-do.call(rbind,eqs); opt<-do.call(rbind,opts)
  if(is.null(opt)||nrow(opt)==0) opt<-opt_schema()

  readr::write_csv(snap,file.path(dataset_name,"snapshot.csv"))
  readr::write_csv(eq,file.path(dataset_name,"equity.csv"))
  readr::write_csv(opt,file.path(dataset_name,"options.csv"))
  writexl::write_xlsx(list(snapshot=snap,equity=eq,options=opt),
                      file.path(dataset_name,paste0(dataset_name,".xlsx")))
  invisible(list(snapshot=snap,equity=eq,options=opt))
}
```

2.

The folders `DATA_2026-02-05` and `DATA_2026-02-06` were created by temporarily removing the leading comment character and running `download_dataset(...)` in the R console on two consecutive trading days, February 5, 2026 and February 6, 2026. Knitting reads the saved CSV files and does not re-download.

```
# SECTION 2: RUN ON THURSDAY 2026-02-05
# download_dataset(dataset_name="DATA_2026-02-05",as_of=as.Date("2026-02-05"))

# SECTION 3: RUN ON FRIDAY 2026-02-06
# download_dataset(dataset_name="DATA_2026-02-06",as_of=as.Date("2026-02-06"))
```

```
read_day<-function(dir){
  snap<-read.csv(file.path(dir,"snapshot.csv"),stringsAsFactors=FALSE)
  eq<-read.csv(file.path(dir,"equity.csv"),stringsAsFactors=FALSE)
  opt<-read.csv(file.path(dir,"options.csv"),stringsAsFactors=FALSE)
  list(snapshot=snap,equity=eq,options=opt)
}

data1_dir<-"DATA_2026-02-05"
data2_dir<-"DATA_2026-02-06"

D1<-read_day(data1_dir)
D2<-read_day(data2_dir)

parse_utc<-function(x){
  as.POSIXct(x,tz="UTC",tryFormats=c("%Y-%m-%dT%H:%M:%SZ","%Y-%m-%d %H:%M:%S"))
}

to_et<-function(t_utc){
  as.POSIXct(format(t_utc,tz="America/New_York",usetz=TRUE),tz="America/New_York")
}

D1$options$snapshot_time_utc<-parse_utc(D1$options$snapshot_time)
D2$options$snapshot_time_utc<-parse_utc(D2$options$snapshot_time)
```

```
D1$options$snapshot_time_et<-to_et(D1$options$snapshot_time_utc)
D2$options$snapshot_time_et<-to_et(D2$options$snapshot_time_utc)

D1$options$expiration_date<-as_date_safe(D1$options$expiration)
D2$options$expiration_date<-as_date_safe(D2$options$expiration)

D1$snapshot_date_et<-as.Date(min(D1$options$snapshot_time_et))
D2$snapshot_date_et<-as.Date(min(D2$options$snapshot_time_et))

exp_D1<-sort(unique(D1$options$expiration_date))
exp_D2<-sort(unique(D2$options$expiration_date))

kbl(data.frame(DATA="DATA1",monthly_expirations=as.character(next_three_monthlies
    (exp_D1,D1$snapshot_date_et))),digits=0)
```

DATA	monthly_expirations
DATA1	2026-02-20
DATA1	2026-03-20
DATA1	2026-04-17

```
kbl(data.frame(DATA="DATA2",monthly_expirations=as.character(next_three_monthlies
    (exp_D2,D2$snapshot_date_et))),digits=0)
```

DATA	monthly_expirations
DATA2	2026-02-20
DATA2	2026-03-20
DATA2	2026-04-17

```
kbl(D1$snapshot[,c("symbol","downloaded_at","underlying_price_at_download")],digits=6)
```

symbol	downloaded_at	underlying_price_at_download
TSLA	2026-02-05 14:54:56.287823	406.01
SPY	2026-02-05 14:55:02.516582	686.19
^VIX	2026-02-05 14:55:02.83657	18.64

```
kbl(D2$snapshot[,c("symbol","downloaded_at","underlying_price_at_download")],digits=6)
```

symbol	downloaded_at	underlying_price_at_download
TSLA	2026-02-06 13:23:56.857472	397.21
SPY	2026-02-06 13:24:02.303604	677.62
^VIX	2026-02-06 13:24:02.509882	21.77

3.

The symbols downloaded for this assignment are TSLA, SPY, and ^VIX. TSLA represents Tesla, Inc., an individual publicly traded equity listed on NASDAQ, and its options are standard equity options that grant the right to buy or sell 100 shares at a specified strike price on or before a stated expiration date. SPY represents the SPDR S&P 500 ETF Trust, which is an exchange traded fund, meaning it is a single investment containing a diversified basket of assets and trades on an exchange like a stock. SPY is designed to track the S&P 500 Index, which consists of 500 large U.S. companies weighted by free float market capitalization, and it provides broad market exposure through a single tradable security rather than requiring investment in all 500 underlying stocks individually. Tesla currently comprises approximately 2 percent of the SPY portfolio, reflecting its weight within the index. Options on both TSLA and SPY have specified expiration dates, including standard monthly expirations that typically occur on the third Friday of each month, as well as additional weekly expirations. The symbol ^VIX represents the CBOE Volatility Index, where the caret indicates that it is an index value rather than a directly tradable security. The VIX measures the market's expectation of 30 day forward looking volatility implied by S&P 500 index options and is commonly interpreted as a gauge of market uncertainty. Its all time intraday high of 89.53 was recorded on October 24, 2008 during the global financial crisis, illustrating the extreme volatility that can occur during periods of severe market stress. Although the index itself cannot be directly purchased, it serves as a benchmark for comparing implied volatility levels observed in SPY and TSLA options.

4.

```
day_count_basis <- 365

add_market_mid <- function(opt){
  opt$market_mid <- mapply(function(b,a,v,l) mid(b,a,v,l),
                           opt$bid, opt$ask, opt$volume, opt$last)
  opt
}

add_ttm <- function(opt, snap_date_et){
  opt$ttm_years <- as.numeric(as_date_safe(opt$expiration_date) - as.Date(snap_date_et))
 ↪  / day_count_basis
  opt
}

D1$options <- add_market_mid(D1$options)
D2$options <- add_market_mid(D2$options)

D1$options <- add_ttm(D1$options, D1$snapshot_date_et)
D2$options <- add_ttm(D2$options, D2$snapshot_date_et)

# Effective fed funds (FRED DFF) converted to continuous compounding.
# Values recorded externally for the two snapshot dates to avoid re-downloading during
 ↪  knit.
r_DATA1 <- 0.0364
r_DATA2 <- 0.0364

kbl(data.frame(
  DATA = c("DATA1","DATA2"),
  snapshot_date_et = c(as.character(D1$snapshot_date_et),
   ↪  as.character(D2$snapshot_date_et)),
  r_cont = c(r_DATA1, r_DATA2)
), digits = 6)
```

DATA	snapshot_date_et	r_cont
DATA1	2026-02-05	0.0364
DATA2	2026-02-06	0.0364

```
ttm_rng1 <- do.call(rbind, lapply(split(D1$options$ttm_years, D1$options$symbol),
                                  function(x) c(min = min(x, na.rm = TRUE), max = max(x,
                                   ↪  na.rm = TRUE))))
ttm_rng1 <- data.frame(symbol = rownames(ttm_rng1), min_ttm = ttm_rng1[,1], max_ttm =
 ↪  ttm_rng1[,2], row.names = NULL)
kbl(ttm_rng1, 6)
```

symbol	min_ttm	max_ttm
SPY	0.041096	0.194521
TSLA	0.041096	0.194521

```
ttm_rng2 <- do.call(rbind, lapply(split(D2$options$ttm_years, D2$options$symbol),
```

```
                                  function(x) c(min = min(x, na.rm = TRUE), max = max(x,
                                   ↪  na.rm = TRUE))))
ttm_rng2 <- data.frame(symbol = rownames(ttm_rng2), min_ttm = ttm_rng2[,1], max_ttm =
 ↪  ttm_rng2[,2], row.names = NULL)
kbl(ttm_rng2, 6)
```

symbol	min_ttm	max_ttm
SPY	0.038356	0.191781
TSLA	0.038356	0.191781

Part 2. Analysis of the data

5.

```
BS_price <- function(S, K, T, r, sigma, type = c("call","put")){
  type <- match.arg(type)
  if(is.na(S) || is.na(K) || is.na(T) || is.na(r) || is.na(sigma)) return(NA_real_)
  if(T <= 0) return(if(type == "call") max(S - K, 0) else max(K - S, 0))
  if(sigma <= 0) return(exp(-r * T) * (if(type == "call") max(S - K, 0) else max(K - S,
   ↪  0)))

  srt <- sigma * sqrt(T)
  d1 <- (log(S / K) + (r + 0.5 * sigma * sigma) * T) / srt
  d2 <- d1 - srt

  if(type == "call"){
    S * pnorm(d1) - K * exp(-r * T) * pnorm(d2)
  } else {
    K * exp(-r * T) * pnorm(-d2) - S * pnorm(-d1)
  }
}

BS_vega <- function(S, K, T, r, sigma){
  if(is.na(S) || is.na(K) || is.na(T) || is.na(r) || is.na(sigma)) return(NA_real_)
  if(T <= 0 || sigma <= 0) return(0)
  srt <- sigma * sqrt(T)
  d1 <- (log(S / K) + (r + 0.5 * sigma * sigma) * T) / srt
  S * dnorm(d1) * sqrt(T)
}
```

6.

```
iv_bisect <- function(S, K, T, r, market, type,
                      tol = 1e-6, max_iter = 200, lo = 1e-6, hi = 5){
  if(is.na(market) || market <= 0 || T <= 0) return(NA_real_)

  f <- function(sig) BS_price(S, K, T, r, sig, type) - market
  flo <- f(lo)
  fhi <- f(hi)

  if(!is.finite(flo) || !is.finite(fhi) || flo * fhi > 0) return(NA_real_)

  a <- lo
  b <- hi
  for(i in 1:max_iter){
```

```
    m <- 0.5 * (a + b)
    fm <- f(m)
    if(!is.finite(fm)) return(NA_real_)
    if(abs(fm) < tol) return(m)
    if(flo * fm <= 0){
      b <- m
      fhi <- fm
    } else {
      a <- m
      flo <- fm
    }
  }
  0.5 * (a + b)
}

moneyness <- function(S, K) S / K

atm_and_nearATM_iv <- function(opt, r, window = c(0.95, 1.05)){
  opt <- opt[opt$symbol %in% c("TSLA","SPY") & !is.na(opt$market_mid), ]
  syms <- sort(unique(opt$symbol))
  out <- list()

  for(sym in syms){
    sub <- opt[opt$symbol == sym, ]
    exps <- sort(unique(as_date_safe(sub$expiration_date)))
    S0 <- as.numeric(sub$underlying_price[1])

    for(e in exps){
      e <- as_date_safe(e)
      se <- sub[as_date_safe(sub$expiration_date) == e, ]
      T <- se$ttm_years[1]

      K_atm <- se$strike[which.min(abs(se$strike - S0))]

      c_row <- se[se$option_type == "call" & se$strike == K_atm, ][1, ]
      p_row <- se[se$option_type == "put"  & se$strike == K_atm, ][1, ]

      iv_c <- iv_bisect(S0, K_atm, T, r, c_row$market_mid, "call")
      iv_p <- iv_bisect(S0, K_atm, T, r, p_row$market_mid, "put")

      mn <- moneyness(S0, se$strike)
      near <- se[mn >= window[1] & mn <= window[2], ]

      ivs <- mapply(function(K, mp, tp) iv_bisect(S0, K, T, r, mp, tp),
                    near$strike, near$market_mid, near$option_type)

      out[[length(out) + 1]] <- data.frame(
        symbol = sym,
        expiration_date = as_date_safe(e),
        S = S0,
        K_ATM = K_atm,
        iv_ATM_call = iv_c,
        iv_ATM_put = iv_p,
        iv_avg_nearATM = mean(ivs, na.rm = TRUE),
        stringsAsFactors = FALSE
```

```
      )
    }
  }
  do.call(rbind, out)
}

iv_summary_D1 <- atm_and_nearATM_iv(D1$options, r_DATA1)
kbl(iv_summary_D1, digits = 6)
```

symbol	expiration_date	S	K_ATM	iv_ATM_call	iv_ATM_put	iv_avg_nearATM
SPY	2026-02-20	678.6800	679	0.208715	0.156643	0.189318
SPY	2026-03-20	678.6800	679	0.188237	0.165709	0.170666
SPY	2026-04-17	678.6800	679	0.176939	0.167450	0.170842
TSLA	2026-02-20	394.8899	395	0.573964	0.398446	0.474003
TSLA	2026-03-20	394.8899	395	0.526585	0.422993	0.471408
TSLA	2026-04-17	394.8899	395	0.519880	0.435272	0.476819

Near ATM implied volatility is higher for TSLA than SPY at all three maturities in DATA1. The near ATM average is less sensitive to isolated strikes with stale quotes.

7.

```
iv_newton <- function(S, K, T, r, market, type,
                      tol = 1e-6, max_iter = 50, sigma0 = 0.2){
  if(is.na(market) || market <= 0 || T <= 0) return(NA_real_)

  sig <- sigma0
  for(i in 1:max_iter){
    price <- BS_price(S, K, T, r, sig, type)
    diff <- price - market
    if(!is.finite(diff)) return(NA_real_)
    if(abs(diff) < tol) return(sig)

    v <- BS_vega(S, K, T, r, sig)
    if(!is.finite(v) || v < 1e-10) return(NA_real_)

    sig_new <- sig - diff / v
    if(!is.finite(sig_new) || sig_new <= 0 || sig_new > 5) return(NA_real_)
    sig <- sig_new
  }
  sig
}

time_compare <- function(opt, r, n = 300){
  sub <- opt[opt$symbol %in% c("TSLA","SPY") & !is.na(opt$market_mid), ]
  sub <- sub[seq_len(min(n, nrow(sub))), ]

  t1 <- system.time({
    invisible(mapply(function(S,K,T,mp,tp) iv_bisect(S,K,T,r,mp,tp),
                     sub$underlying_price, sub$strike, sub$ttm_years,
                     sub$market_mid, sub$option_type))
  })[["elapsed"]]

  t2 <- system.time({
    invisible(mapply(function(S,K,T,mp,tp) iv_newton(S,K,T,r,mp,tp),
                     sub$underlying_price, sub$strike, sub$ttm_years,
```

```
                     sub$market_mid, sub$option_type))
  })[["elapsed"]]

  data.frame(
    n_used = nrow(sub),
    bisection_seconds = t1,
    newton_seconds = t2,
    stringsAsFactors = FALSE
  )
}
```

```
kbl(time_compare(D1$options, r_DATA1, n = 300), digits = 6)
```

n_used	bisection_seconds	newton_seconds
300	0.06	0.03

Newton fails when Vega is effectively zero or when the update exits the admissible volatility range. Bisection fails when the pricing function does not change sign on the bracket.

8.

```
iv_table_20 <- function(opt, r){
  opt <- opt[opt$symbol %in% c("TSLA","SPY") & !is.na(opt$market_mid), ]
  out <- list()

  for(sym in sort(unique(opt$symbol))){
    sub <- opt[opt$symbol == sym, ]
    S0 <- as.numeric(sub$underlying_price[1])

    for(e in sort(unique(as_date_safe(sub$expiration_date)))){
      e <- as_date_safe(e)
      for(tp in c("call","put")){
        se <- sub[as_date_safe(sub$expiration_date) == e & sub$option_type == tp, ]
        if(nrow(se) == 0) next

        se <- se[order(abs(se$strike - S0)), ]
        se <- se[seq_len(min(20, nrow(se))), ]

        se$expiration_date <- as_date_safe(se$expiration_date)

        se$iv_bisect <- mapply(function(K,T,mp) iv_bisect(S0, K, T, r, mp, tp),
                               se$strike, se$ttm_years, se$market_mid)

        se$iv_newton <- mapply(function(K,T,mp) iv_newton(S0, K, T, r, mp, tp),
                               se$strike, se$ttm_years, se$market_mid)

        out[[length(out) + 1]] <- se[,
↪   c("symbol","expiration_date","option_type","strike",
                                        "market_mid","iv_bisect","iv_newton")]
      }
    }
  }

  tab <- do.call(rbind, out)
  tab$expiration_date <- as_date_safe(tab$expiration_date)
  tab[order(tab$symbol, tab$expiration_date, tab$option_type, tab$strike), ]
```

```
}

iv20_D1 <- iv_table_20(D1$options, r_DATA1)

kbl(head(iv20_D1,20),digits=6)
```

symbol	expiration_date	option_type	strike	market_mid	iv_bisect	iv_newton
SPY	2026-02-20	call	669	18.850	0.235223	0.235223
SPY	2026-02-20	call	670	18.090	0.232424	0.232424
SPY	2026-02-20	call	671	17.320	0.229258	0.229258
SPY	2026-02-20	call	672	16.590	0.226671	0.226672
SPY	2026-02-20	call	673	15.865	0.223986	0.223986
SPY	2026-02-20	call	674	15.170	0.221661	0.221661
SPY	2026-02-20	call	675	14.485	0.219313	0.219313
SPY	2026-02-20	call	676	13.795	0.216657	0.216657
SPY	2026-02-20	call	677	13.010	0.212040	0.212040
SPY	2026-02-20	call	678	12.455	0.211413	0.211413
SPY	2026-02-20	call	679	11.800	0.208715	0.208715
SPY	2026-02-20	call	680	11.105	0.205045	0.205045
SPY	2026-02-20	call	681	10.450	0.201849	0.201849
SPY	2026-02-20	call	682	9.865	0.199665	0.199665
SPY	2026-02-20	call	683	9.225	0.196200	0.196200
SPY	2026-02-20	call	684	8.665	0.193913	0.193913
SPY	2026-02-20	call	685	8.075	0.190776	0.190776
SPY	2026-02-20	call	686	7.525	0.188067	0.188067
SPY	2026-02-20	call	687	7.015	0.185786	0.185786
SPY	2026-02-20	call	688	6.495	0.182983	0.182983

```
iv_means <- aggregate(iv_bisect ~ symbol + expiration_date + option_type,
                      data = iv20_D1,
                      FUN = function(x) mean(x, na.rm = TRUE))
iv_means$expiration_date <- as_date_safe(iv_means$expiration_date)
kbl(iv_means, digits = 6)
```

symbol	expiration_date	option_type	iv_bisect
SPY	2026-02-20	call	0.209082
TSLA	2026-02-20	call	0.571964
SPY	2026-03-20	call	0.189324
TSLA	2026-03-20	call	0.542761
SPY	2026-04-17	call	0.180018
TSLA	2026-04-17	call	0.529372
SPY	2026-02-20	put	0.157312
TSLA	2026-02-20	put	0.375104
SPY	2026-03-20	put	0.166292
TSLA	2026-03-20	put	0.424188
SPY	2026-04-17	put	0.168302
TSLA	2026-04-17	put	0.439402

```
vix_level_D1 <- tail(D1$equity$close[D1$equity$symbol == "^VIX"], 1)
kbl(data.frame(VIX_close_level = vix_level_D1), digits = 6)
```

VIX_close_level
18.64

9.

```r
parity_compare <- function(opt, r){
  sub <- opt[opt$symbol %in% c("TSLA","SPY") & !is.na(opt$market_mid), ]

  calls <- sub[sub$option_type == "call", ]
  puts  <- sub[sub$option_type == "put",  ]

  calls$key <- paste(calls$symbol, as.character(as_date_safe(calls$expiration_date)),
↪  calls$strike)
  puts$key  <- paste(puts$symbol,  as.character(as_date_safe(puts$expiration_date)),
↪  puts$strike)

  m <- merge(calls, puts, by = "key", suffixes = c("_C","_P"))

  S <- m$underlying_price_C
  K <- m$strike_C
  T <- m$ttm_years_C
  C <- m$market_mid_C
  P <- m$market_mid_P

  P_from_C <- C - S + K * exp(-r * T)
  C_from_P <- P + S - K * exp(-r * T)

  data.frame(
    symbol = m$symbol_C,
    expiration_date = as_date_safe(m$expiration_date_C),
    strike = K,
    T = T,
    moneyness = S / K,
    spread_P = m$ask_P - m$bid_P,
    volume_P = m$volume_P,
    abs_err_put = abs(P_from_C - P),
    stringsAsFactors = FALSE
  )
}

pc_D1 <- parity_compare(D1$options, r_DATA1)

pc_q <- function(x) as.numeric(quantile(x, c(0, 0.5, 0.9, 0.99, 1), na.rm = TRUE))

kbl(data.frame(
  stat = c("min","p50","p90","p99","max"),
  abs_err_put = pc_q(pc_D1$abs_err_put)
), digits = 6)
```

stat	abs_err_put
min	0.015011
p50	3.603898
p90	33.158736
p99	56.836209
max	109.634392

```r
top10 <- pc_D1[order(pc_D1$abs_err_put, decreasing = TRUE), ][1:10, ]
kbl(top10, digits = 4)
```

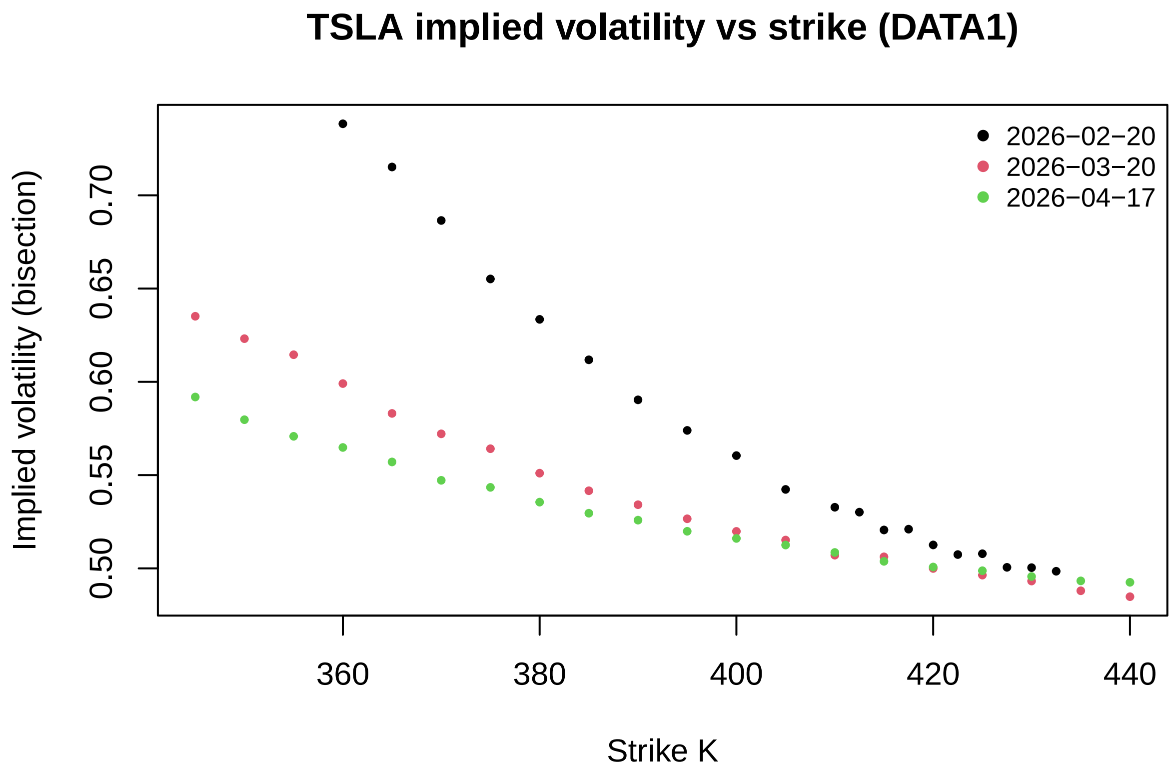
symbol	expiration_date	strike	T	moneyness	spread_P	volume_P	abs_err_put
SPY	2026-03-20	840	0.1178	0.8080	2.81	40	109.6344
SPY	2026-03-20	295	0.1178	2.3009	0.01	17	104.3773
TSLA	2026-02-20	695	0.0411	0.5682	2.25	NA	96.1412
TSLA	2026-04-17	25	0.1945	15.7908	0.23	NA	93.6136
TSLA	2026-02-20	870	0.0411	0.4539	2.95	NA	92.5196
SPY	2026-03-20	830	0.1178	0.8178	3.90	10	84.1066
TSLA	2026-04-17	40	0.1945	9.8692	0.03	130	82.4378
TSLA	2026-03-20	90	0.1178	4.3877	0.04	3	75.2942
SPY	2026-03-20	255	0.1178	2.6618	0.01	4	69.7312
TSLA	2026-02-20	920	0.0411	0.4292	2.00	NA	55.7149

Both directions of parity (put-from-call and call-from-put) were computed; the table reports the larger deviation for compactness. Parity residuals concentrate in contracts with poor microstructure: low volume, wide spreads, and deep moneyness. American early exercise premia also shifts observed mids away from European parity.

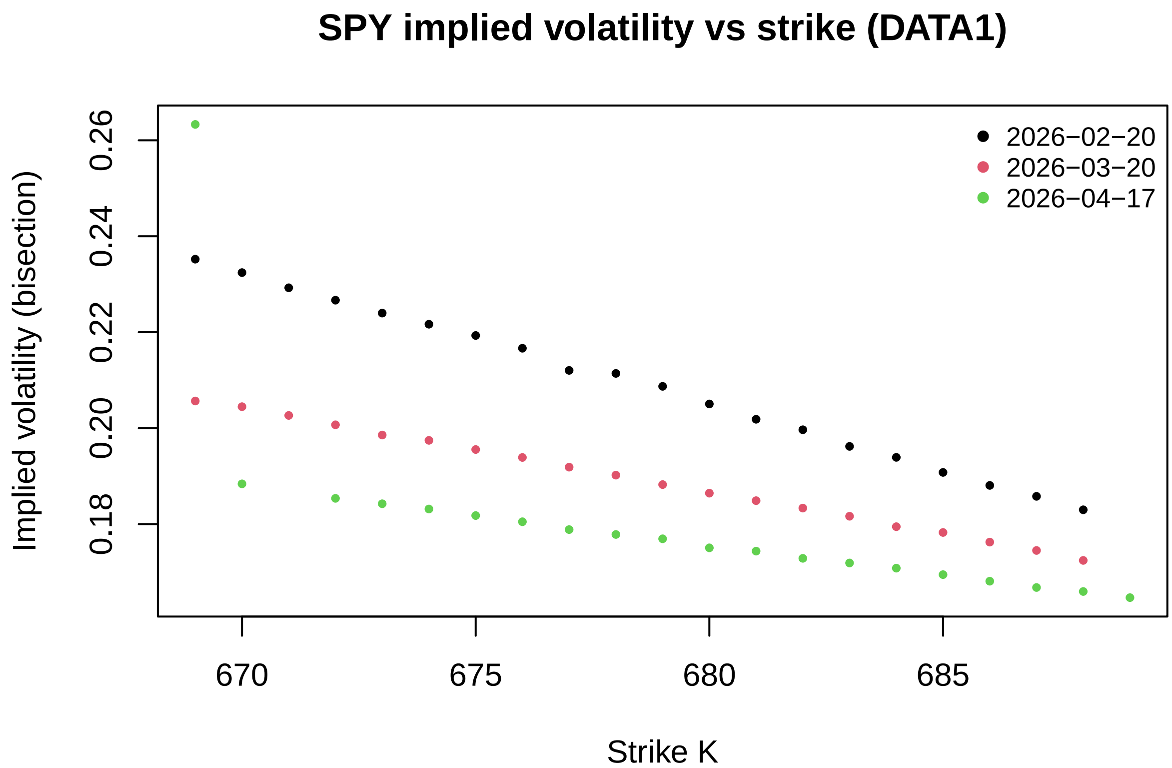
10.

```
plot_smile <- function(tab, sym){
  sub <- tab[tab$symbol == sym & tab$option_type == "call", ]
  exps <- sort(unique(as_date_safe(sub$expiration_date)))
  if(length(exps) == 0) return(invisible(NULL))

  ylim <- range(sub$iv_bisect, na.rm = TRUE)
  xlim <- range(sub$strike, na.rm = TRUE)

  plot(NA, xlim = xlim, ylim = ylim,
       xlab = "Strike K",
       ylab = "Implied volatility (bisection)",
       main = paste(sym, "implied volatility vs strike (DATA1)"))

  cols <- seq_along(exps)
  for(i in seq_along(exps)){
    e <- exps[i]
    s <- sub[as_date_safe(sub$expiration_date) == e, ]
    points(s$strike, s$iv_bisect, pch = 16, cex = 0.6, col = cols[i])
  }
  legend("topright", legend = as.character(exps), pch = 16, col = cols, bty = "n", cex =
   ↪  0.8)
}

plot_closest_smile <- function(tab, sym){
  sub <- tab[tab$symbol == sym & tab$option_type == "call", ]
  exps <- sort(unique(as_date_safe(sub$expiration_date)))
  if(length(exps) == 0) return(invisible(NULL))

  e0 <- exps[1]
  s <- sub[as_date_safe(sub$expiration_date) == e0, ]
  plot(s$strike, s$iv_bisect,
       pch = 16, cex = 0.7,
       xlab = "Strike K",
       ylab = "Implied volatility (bisection)",
       main = paste(sym, "closest maturity IV vs strike (DATA1)"))
}
```

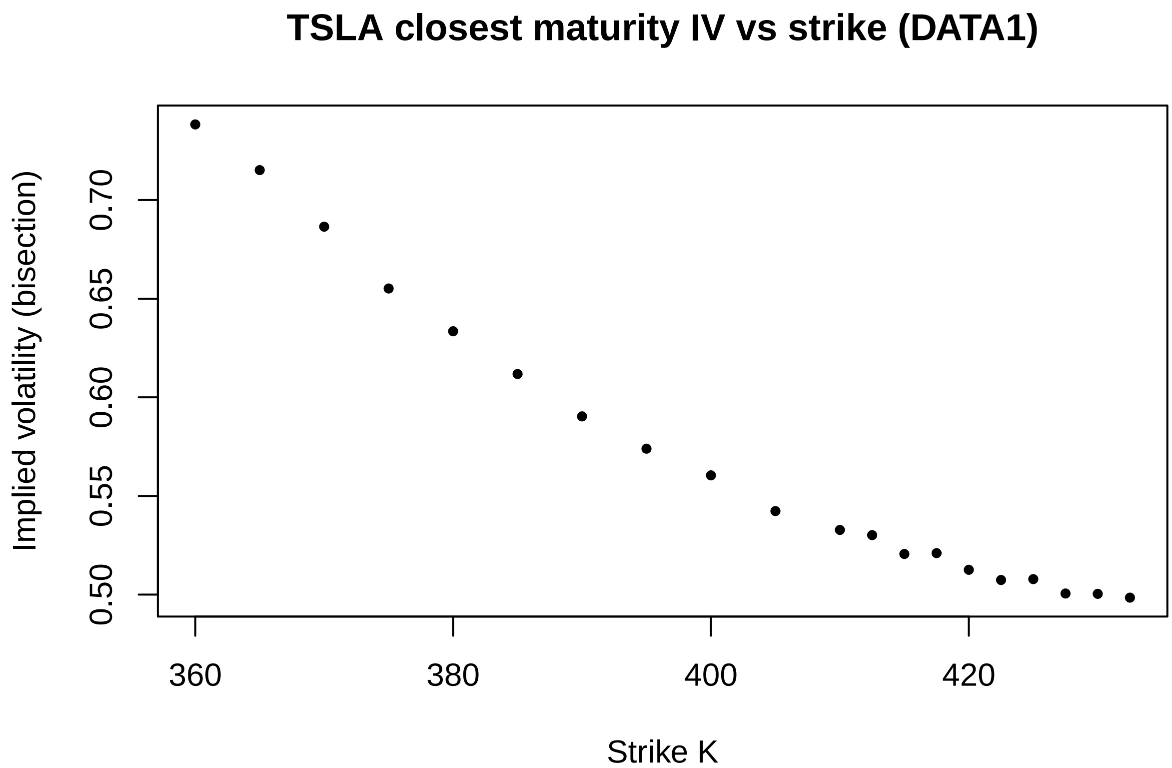
```
plot_smile(iv20_D1, "TSLA")
```



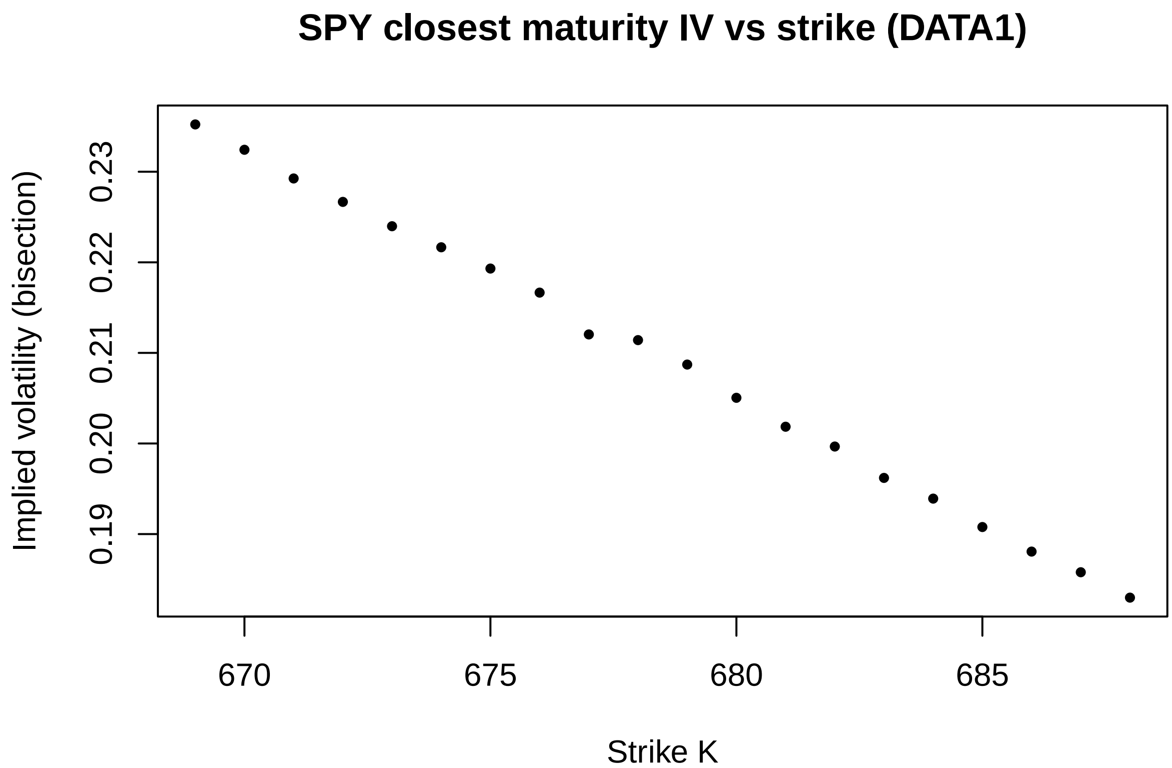

```
plot_smile(iv20_D1, "SPY")
```




```
plot_closest_smile(iv20_D1, "TSLA")
```




```
plot_closest_smile(iv20_D1, "SPY")
```



11.

```
greeks_call<-function(S,K,T,r,sigma){
  if(is.na(S)||is.na(K)||is.na(T)||is.na(r)||is.na(sigma)||T<=0||sigma<=0){
    return(c(Delta=NA_real_,Gamma=NA_real_,Vega=NA_real_))
  }
  srt<-sigma*sqrt(T)
  d1<-(log(S/K)+(r+0.5*sigma*sigma)*T)/srt
  Delta<-pnorm(d1)
  Gamma<-dnorm(d1)/(S*sigma*sqrt(T))
  Vega<-S*dnorm(d1)*sqrt(T)
  c(Delta=Delta,Gamma=Gamma,Vega=Vega)
}

greeks_fd<-function(S,K,T,r,sigma,hS=1e-3,hV=1e-3){
  C0<-BS_price(S,K,T,r,sigma,"call")
  CSp<-BS_price(S*(1+hS),K,T,r,sigma,"call")
  CSm<-BS_price(S*(1-hS),K,T,r,sigma,"call")
  Delta<-(CSp-CSm)/(2*S*hS)
  Gamma<-(CSp-2*C0+CSm)/((S*hS)^2)
  CVp<-BS_price(S,K,T,r,sigma*(1+hV),"call")
  CVm<-BS_price(S,K,T,r,sigma*(1-hV),"call")
  Vega<-(CVp-CVm)/(2*sigma*hV)
  c(Delta=Delta,Gamma=Gamma,Vega=Vega)
}
```

```r
greek_table<-function(opt,r){
  opt<-opt[opt$symbol%in%c("TSLA","SPY")&opt$option_type=="call"&!is.na(opt$market_mid),]
  out<-list()
  for(sym in sort(unique(opt$symbol))){
    sub<-opt[opt$symbol==sym,]
    S0<-as.numeric(sub$underlying_price[1])
    for(e in sort(unique(as_date_safe(sub$expiration_date)))){
      e<-as_date_safe(e)
      se<-sub[as_date_safe(sub$expiration_date)==e,]
      K<-se$strike[which.min(abs(se$strike-S0))]
      row<-se[se$strike==K,][1,]
      T<-row$ttm_years
      iv<-iv_bisect(S0,K,T,r,row$market_mid,"call")
      ga<-greeks_call(S0,K,T,r,iv)
      gn<-greeks_fd(S0,K,T,r,iv)
      out[[length(out)+1]]<-data.frame(
        symbol=sym,
        expiration_date=as_date_safe(e),
        K=K,
        T=T,
        iv=iv,
        Delta_analytic=as.numeric(ga["Delta"]),
        Gamma_analytic=as.numeric(ga["Gamma"]),
        Vega_analytic=as.numeric(ga["Vega"]),
        Delta_fd=as.numeric(gn["Delta"]),
        Gamma_fd=as.numeric(gn["Gamma"]),
        Vega_fd=as.numeric(gn["Vega"]),
        stringsAsFactors=FALSE
      )
    }
  }
  ans<-do.call(rbind,out)
  rownames(ans)<-NULL
  ans
}

gt<-greek_table(D1$options,r_DATA1)

kbl(gt[,c("symbol","expiration_date","K","T","iv","Delta_analytic","Gamma_analytic",
  "Vega_analytic")],digits=6)
```

symbol	expiration_date	K	T	iv	Delta_analytic	Gamma_analytic	Vega_analytic
SPY	2026-02-20	679	0.041096	0.208715	0.518093	0.013879	54.83115
SPY	2026-03-20	679	0.117808	0.188237	0.536405	0.009060	92.54430
SPY	2026-04-17	679	0.194521	0.176939	0.549228	0.007475	118.50443
TSLA	2026-02-20	395	0.041096	0.573964	0.527361	0.008662	31.86121
TSLA	2026-03-20	395	0.117808	0.526585	0.544808	0.005554	53.73076
TSLA	2026-04-17	395	0.194521	0.519880	0.557372	0.004360	68.76171

```r
kbl(gt[,c("symbol","expiration_date","Delta_fd","Gamma_fd","Vega_fd")],digits=6)
```

symbol	expiration_date	Delta_fd	Gamma_fd	Vega_fd
SPY	2026-02-20	0.518090	0.013878	54.83115
SPY	2026-03-20	0.536402	0.009060	92.54430
SPY	2026-04-17	0.549226	0.007475	118.50443
TSLA	2026-02-20	0.527360	0.008662	31.86121
TSLA	2026-03-20	0.544807	0.005554	53.73076
TSLA	2026-04-17	0.557372	0.004360	68.76171

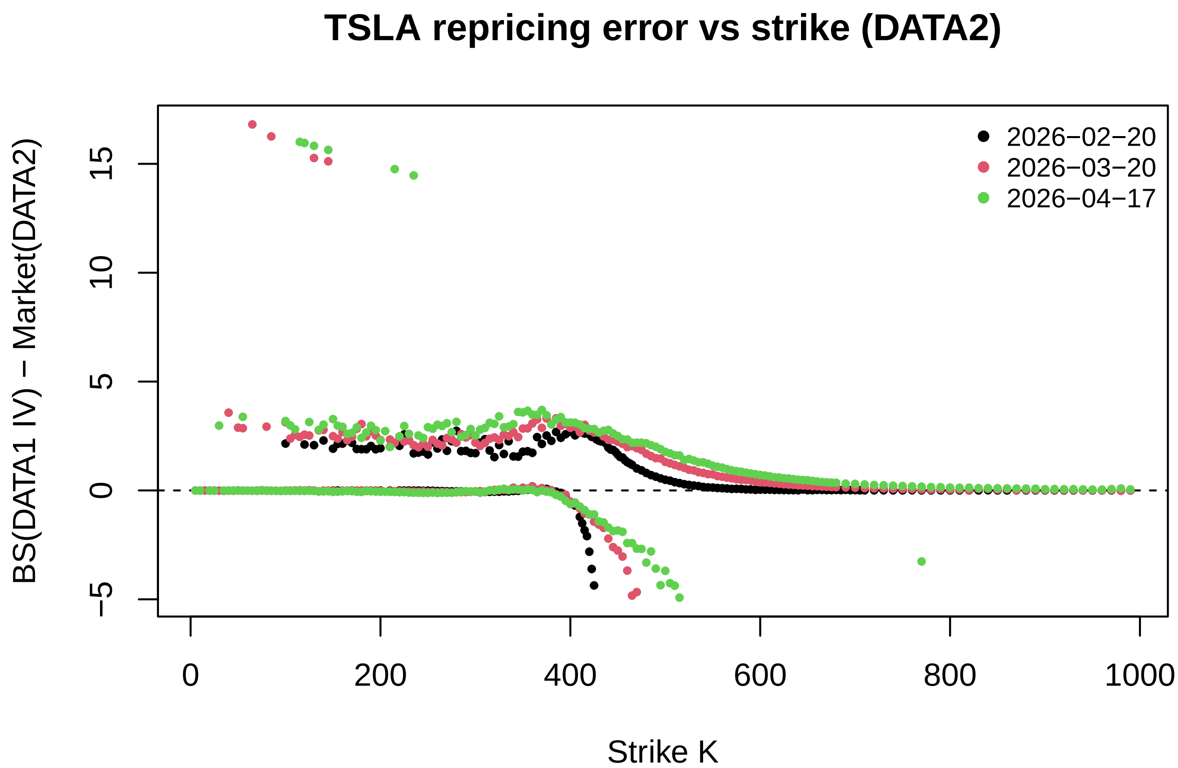
12.

```
iv_surface_D1 <- function(opt, r){
  sub <- opt[opt$symbol %in% c("TSLA","SPY") & !is.na(opt$market_mid), ]
  sub$key <- paste(sub$symbol, as.character(as_date_safe(sub$expiration_date)),
                   sub$option_type, sub$strike)

  sub$iv_bisect <- mapply(function(S,K,T,mp,tp) iv_bisect(S, K, T, r, mp, tp),
                          sub$underlying_price, sub$strike, sub$ttm_years,
                          sub$market_mid, sub$option_type)

  sub[, c("key","iv_bisect")]
}

surf_D1 <- iv_surface_D1(D1$options, r_DATA1)

D2o <- D2$options
D2o$expiration_date <- as_date_safe(D2o$expiration)
D2o$market_mid <- mapply(function(b,a,v,l) mid(b,a,v,l),
                         D2o$bid, D2o$ask, D2o$volume, D2o$last)
D2o$ttm_years <- as.numeric(D2o$expiration_date - D2$snapshot_date_et) / day_count_basis
D2o$key <- paste(D2o$symbol, as.character(as_date_safe(D2o$expiration_date)),
                 D2o$option_type, D2o$strike)

D2m <- merge(D2o, surf_D1[, c("key","iv_bisect")], by = "key", all.x = TRUE, sort =
↪  FALSE)
names(D2m)[names(D2m) == "iv_bisect"] <- "iv_from_DATA1"

D2m$bs_price <- mapply(function(S,K,T,vol,tp){
  if(is.na(vol) || is.na(S) || is.na(K) || is.na(T) || T <= 0) return(NA_real_)
  BS_price(S, K, T, r_DATA2, vol, tp)
}, D2m$underlying_price, D2m$strike, D2m$ttm_years, D2m$iv_from_DATA1, D2m$option_type)

D2m$error <- D2m$bs_price - D2m$market_mid

err_summary <- do.call(rbind, lapply(
  split(D2m, interaction(D2m$symbol, D2m$expiration_date, D2m$option_type, drop = TRUE)),
  function(g){
    data.frame(
      symbol = g$symbol[1],
      expiration_date = as_date_safe(g$expiration_date[1]),
      option_type = g$option_type[1],
      mean_error = mean(g$error, na.rm = TRUE),
      rmse = sqrt(mean(g$error * g$error, na.rm = TRUE)),
      stringsAsFactors = FALSE
    )
  }
```

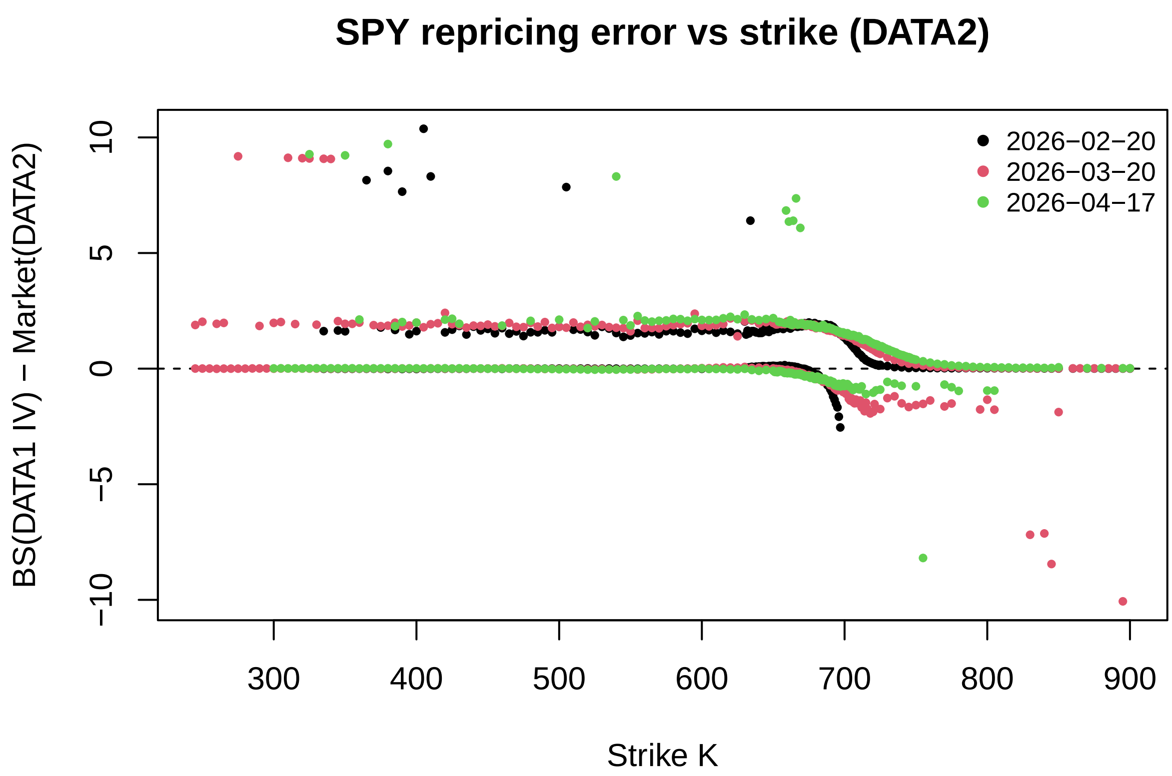
```
))

rownames(err_summary) <- NULL
err_summary <- err_summary[order(err_summary$symbol, err_summary$expiration_date,
 ↪  err_summary$option_type), ]
kbl(err_summary, 6)
```

symbol	expiration_date	option_type	mean_error	rmse
SPY	2026-02-20	call	1.485872	2.132300
SPY	2026-02-20	put	-0.115795	0.435429
SPY	2026-03-20	call	1.654093	2.266575
SPY	2026-03-20	put	-0.610817	1.472303
SPY	2026-04-17	call	1.671981	2.378921
SPY	2026-04-17	put	-0.326929	0.799150
TSLA	2026-02-20	call	1.087222	1.490691
TSLA	2026-02-20	put	-0.320688	0.919263
TSLA	2026-03-20	call	1.874053	3.173297
TSLA	2026-03-20	put	-0.366618	1.029411
TSLA	2026-04-17	call	2.300871	3.736596
TSLA	2026-04-17	put	-0.623282	1.346845

```
plot_error <- function(df, sym){
  sub <- df[df$symbol == sym & !is.na(df$error), ]
  exps <- sort(unique(as_date_safe(sub$expiration_date)))
  if(length(exps) == 0) return(invisible(NULL))

  ylim <- range(sub$error, na.rm = TRUE)
  xlim <- range(sub$strike, na.rm = TRUE)

  plot(NA, xlim = xlim, ylim = ylim,
       xlab = "Strike K",
       ylab = "BS(DATA1 IV) - Market(DATA2)",
       main = paste(sym, "repricing error vs strike (DATA2)"))
  abline(h = 0, lty = 2)

  cols <- seq_along(exps)
  for(i in seq_along(exps)){
    e <- exps[i]
    s <- sub[as_date_safe(sub$expiration_date) == e, ]
    points(s$strike, s$error, pch = 16, cex = 0.6, col = cols[i])
  }
  legend("topright", legend = as.character(exps), pch = 16, col = cols, bty = "n", cex =
   ↪  0.8)
}

plot_error(D2m, "TSLA")
```




```
plot_error(D2m, "SPY")
```



Holding volatility fixed at DATA1 implied volatilities isolates day over day surface movement in the residual error.

Part 3. Numerical integration of real valued functions. AMM arbitrage fee revenue

(a)

```
swap_amounts <- function(s, x = 1000, y = 1000, gamma = 0.003){
  k <- x * y
  P <- y / x
  upper <- P / (1 - gamma)
  lower <- P * (1 - gamma)

  if(s > upper){
    x1 <- sqrt(k / (s * (1 - gamma)))
    y1 <- k / x1
    dx <- x - x1
    dy <- (y1 - y) / (1 - gamma)
    fee_usdc <- gamma * dy
    c(dx = dx, dy = dy, fee_usdc = fee_usdc)
  } else if(s < lower){
    x1 <- sqrt(k * (1 - gamma) / s)
    y1 <- k / x1
    dy <- y - y1
    dx <- (x1 - x) / (1 - gamma)
```

```
    fee_usdc <- gamma * dx * s
    c(dx = dx, dy = dy, fee_usdc = fee_usdc)
  } else {
    c(dx = 0, dy = 0, fee_usdc = 0)
  }
}

examples <- data.frame(St1 = c(0.95, 0.99, 1.00, 1.01, 1.05))
tmp <- t(sapply(examples$St1, function(s) swap_amounts(s, gamma = 0.003)))
ans <- cbind(examples, tmp)
rownames(ans) <- NULL
kbl(ans, digits = 6)
```

St1	dx	dy	fee_usdc
0.95	24.511764	23.855249	0.069859
0.99	3.539745	3.516715	0.010513
1.00	0.000000	0.000000	0.000000
1.01	3.466887	3.489417	0.010468
1.05	22.632775	23.226559	0.069680

(b)

For the numerical approximation, sigma = 0.2 and gamma = 0.003 as specified by the course staff.

```
lognorm_pdf <- function(s, mu, sd){
  ifelse(s > 0, dnorm((log(s) - mu) / sd) / (s * sd), 0)
}

E_fee_trapz <- function(sigma, gamma, x = 1000, y = 1000, dt = 1/365, smax = 5, n =
 ↪  2000){
  P <- y / x
  upper <- P / (1 - gamma)
  lower <- P * (1 - gamma)

  mu <- -0.5 * sigma * sigma * dt
  sd <- sigma * sqrt(dt)

  trapz <- function(a, b, integrand){
    grid <- seq(a, b, length.out = n + 1)
    h <- (b - a) / n
    vals <- sapply(grid, integrand)
    (h/2) * (vals[1] + vals[n+1] + 2 * sum(vals[2:n]))
  }

  I1 <- trapz(upper, smax, function(s){
    sa <- swap_amounts(s, x = x, y = y, gamma = gamma)
    sa["fee_usdc"] * lognorm_pdf(s, mu, sd)
  })

  I2 <- trapz(1e-8, lower, function(s){
    sa <- swap_amounts(s, x = x, y = y, gamma = gamma)
    sa["fee_usdc"] * lognorm_pdf(s, mu, sd)
  })

  as.numeric(I1 + I2)
}
```

```r
kbl(data.frame(example_E_fee = E_fee_trapz(0.2, 0.003)), digits = 8)
```

example_E_fee
0.00850269

(c)

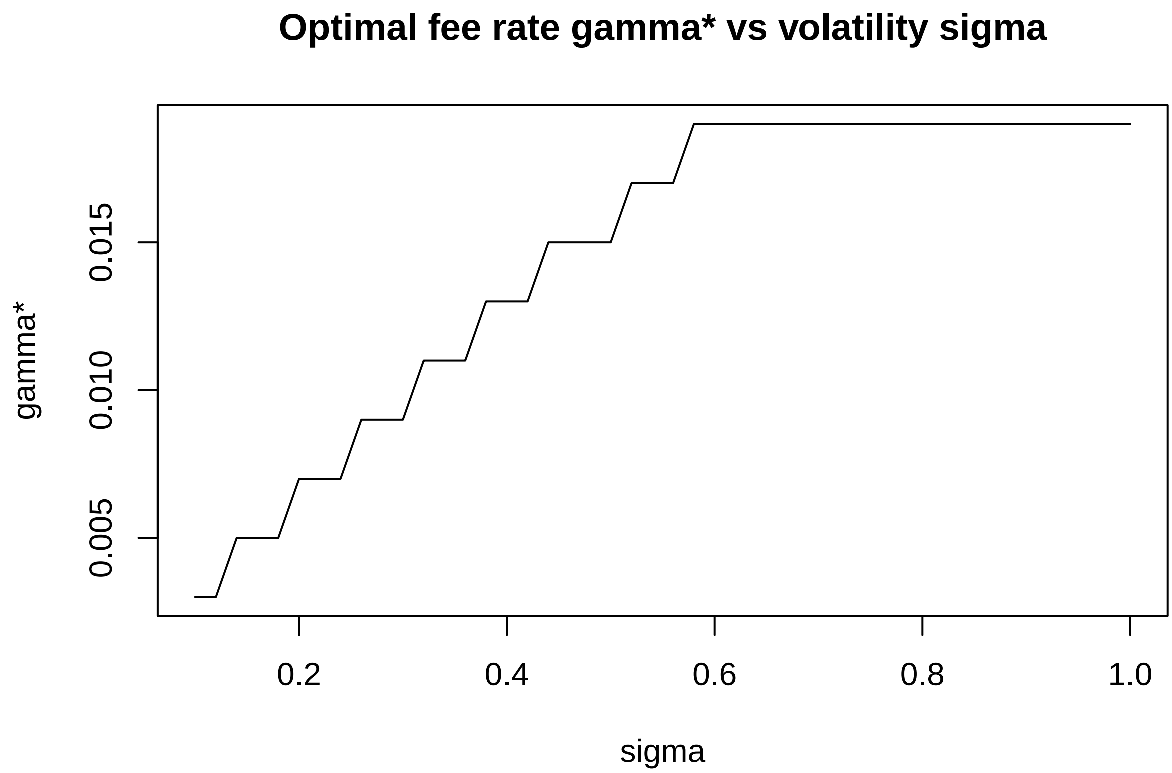
```r
sigmas <- c(0.2, 0.6, 1.0)
gammas <- c(0.001, 0.003, 0.01)

tab <- expand.grid(sigma = sigmas, gamma = gammas)
tab$E_fee <- mapply(E_fee_trapz, tab$sigma, tab$gamma)
kbl(tab, digits = 10)
```

sigma	gamma	E_fee
0.2	0.001	0.003678504
0.6	0.001	0.011921131
1.0	0.001	0.020059375
0.2	0.003	0.008502686
0.6	0.003	0.032976609
1.0	0.003	0.057379739
0.2	0.010	0.009388617
0.6	0.010	0.081061380
1.0	0.010	0.160676894

```r
best <- aggregate(E_fee ~ sigma, data = tab, FUN = max)
best$gamma_star <- mapply(function(s){
  tmp <- tab[tab$sigma == s, ]
  tmp$gamma[which.max(tmp$E_fee)]
}, best$sigma)
kbl(best, digits = 10)
```

sigma	E_fee	gamma_star
0.2	0.009388617	0.01
0.6	0.081061380	0.01
1.0	0.160676894	0.01

```r
sigma_grid <- seq(0.1, 1.0, by = 0.02)
gamma_choices <- seq(0.001, 0.02, by = 0.002)

gamma_star <- sapply(sigma_grid, function(s){
  vals <- sapply(gamma_choices, function(g) E_fee_trapz(s, g, n = 1500))
  gamma_choices[which.max(vals)]
})

plot(sigma_grid, gamma_star, type = "l", xlab = "sigma", ylab = "gamma*",
     main = "Optimal fee rate gamma* vs volatility sigma")
```



Increasing n beyond 2000 produces negligible change in expected fee estimates, indicating numerical stability at the chosen grid resolution.

Part 4. Bonus. Numerical integration check

1.

```
I1_true <- 9/4
I2_true <- (exp(3) - 1) * (exp(1) - 1)
kbl(data.frame(I1_true = I1_true, I2_true = I2_true), digits = 10)
```

I1_true	I2_true
2.25	32.79433

2.

```
f1 <- function(x, y) x * y
f2 <- function(x, y) exp(x + y)

double_trapz <- function(f, x0 = 0, x1 = 1, y0 = 0, y1 = 3, dx, dy){
  xs <- seq(x0, x1, by = dx); if(tail(xs, 1) != x1) xs <- c(xs, x1)
  ys <- seq(y0, y1, by = dy); if(tail(ys, 1) != y1) ys <- c(ys, y1)

  n <- length(xs) - 1
  m <- length(ys) - 1

  total <- 0
```

```
  for(i in 1:n){
    for(j in 1:m){
      xi <- xs[i];   xi1 <- xs[i+1]
      yj <- ys[j];   yj1 <- ys[j+1]
      xm <- 0.5 * (xi + xi1)
      ym <- 0.5 * (yj + yj1)
      hx <- xi1 - xi
      hy <- yj1 - yj
      total <- total + (hx * hy / 16) * (
        f(xi, yj) + f(xi, yj1) + f(xi1, yj) + f(xi1, yj1) +
          2 * (f(xm, yj) + f(xm, yj1) + f(xi, ym) + f(xi1, ym)) +
          4 * f(xm, ym)
      )
    }
  }
  total
}

pairs <- rbind(
  c(dx = 0.25, dy = 0.75),
  c(dx = 0.20, dy = 0.60),
  c(dx = 0.10, dy = 0.30),
  c(dx = 0.05, dy = 0.15)
)

res <- data.frame()
for(i in 1:nrow(pairs)){
  dx <- pairs[i, "dx"]
  dy <- pairs[i, "dy"]
  I1_hat <- double_trapz(f1, dx = dx, dy = dy)
  I2_hat <- double_trapz(f2, dx = dx, dy = dy)
  res <- rbind(res, data.frame(
    dx = dx, dy = dy,
    I1_hat = I1_hat, I1_err = I1_hat - I1_true,
    I2_hat = I2_hat, I2_err = I2_hat - I2_true
  ))
}
rownames(res) <- NULL
kbl(res, digits = 12)
```

dx	dy	I1_hat	I1_err	I2_hat	I2_err
0.25	0.75	2.25	0	33.22093	0.42659979
0.20	0.60	2.25	0	33.06745	0.27311802
0.10	0.30	2.25	0	32.86264	0.06831100
0.05	0.15	2.25	0	32.81141	0.01707972

As the grid is refined, approximation error decreases. Higher curvature in exp(x+y) slows convergence relative to x*y.

-James G. Tenreiro