AF6305 Project

1 Common Library, Variables, and Functions

1.1 Library

```
library(tidyverse)
library(scales)
library(frenchdata)
library(RSQLite)
library(RPostgres)
library(dbplyr)
library(progress)
library(slider)
library(lmtest)
library(broom)
library(sandwich)
library(DescTools)
```

1.2 Variables

```
start_date <- ymd("1996-01-01")
end_date <- ymd("2020-12-31")

db <- dbConnect(
  SQLite(),
  "data/main.sqlite",
  extended_types = TRUE
)
```

1.3 Functions

```
sum_stats <- function(df, cols) {
  results <- data.frame()

  for (col in cols) {
    # remove NA for this column
    df_na <- df |> filter(!is.na(!!rlang::sym(col)))

    result <- df_na |>
      group_by(date) |> # for a cross-section
      summarise(
        mean = mean(!!rlang::sym(col)),
        sd = sd(!!rlang::sym(col)),
        skew = skewness(!!rlang::sym(col)),
        kurtosis = kurtosis(!!rlang::sym(col)),
        min = min(!!rlang::sym(col)),
```

```
        `5%` = quantile(!!rlang::sym(col), 0.05),
        `25%` = quantile(!!rlang::sym(col), 0.25),
        median = median(!!rlang::sym(col)),
        `75%` = quantile(!!rlang::sym(col), 0.75),
        `95%` = quantile(!!rlang::sym(col), 0.95),
        max = max(!!rlang::sym(col)),
        n = sum(!is.na(!!rlang::sym(col)))
      ) |>
      summarise(
        mean = mean(mean),
        sd = mean(sd),
        skew = mean(skew),
        kurtosis = mean(kurtosis),
        min = mean(min),
        `5%` = mean(`5%`),
        `25%` = mean(`25%`),
        median = mean(median),
        `75%` = mean(`75%`),
        `95%` = mean(`95%`),
        max = mean(max),
        n = floor(mean(n))
      )

    result$var <- col # Add the column name to the result
    result <- result |> select(var, everything()) # Move the column name to the front
    results <- rbind(results, result) # Append the result to the results data frame
  }

  return(results)
}
```

2 Data Download and Cleaning

2.1 WRDS Connection

```
wrds <- dbConnect(
  Postgres(),
  host = "wrds-pgdata.wharton.upenn.edu",
  dbname = "wrds",
  port = 9737,
  sslmode = "require",
  user = Sys.getenv("WRDS_USERNAME"),
  password = Sys.getenv("WRDS_PASSWORD")
)
```

2.2 CRSP Monthly

```
msf_db <- tbl(wrds, in_schema("crsp", "msf"))
msenames_db <- tbl(wrds, in_schema("crsp", "msenames"))
msedelist_db <- tbl(wrds, in_schema("crsp", "msedelist"))

# Takes about 2 minutes
```

```r
crsp_monthly <- msf_db |>
  filter(date >= start_date & date <= end_date) |>
  inner_join(
    msenames_db |>
      filter(shrcd %in% c(10, 11)) |> # US Stocks
      select(permno, exchcd, siccd, namedt, nameendt),
    by = c("permno")
  ) |>
  filter(date >= namedt & date <= nameendt) |>
  mutate(month = floor_date(date, "month")) |>
  left_join(
    msedelist_db |>
      select(permno, dlstdt, dlret, dlstcd) |>
      mutate(month = floor_date(dlstdt, "month")),
    by = c("permno", "month")
  ) |>
  select(
    permno, # Security identifier
    date, # Date of the observation
    month, # Month of the observation
    ret, # Return
    shrout, # Shares outstanding (in thousands)
    prc, # Price or negative bid/ask average on last trading day on the month
    exchcd, # Exchange code
    siccd, # Industry code
    dlret, # Delisting return
    dlstcd # Delisting code
  ) |>
  collect() |>
  mutate(
    month = ymd(month),
    shrout = shrout * 1000
  )

dbDisconnect(wrds)

# Calc Market Cap --------------------------------------------------------

crsp_monthly <- crsp_monthly |>
  mutate(
    mktcap = abs(shrout * abs(prc)) / 10^6,
    mktcap = na_if(mktcap, 0)
  )

# Calc Adjusted Return ---------------------------------------------------

crsp_monthly <- crsp_monthly |>
  mutate(
    ret_adj = case_when(
      is.na(dlstcd) ~ ret, # ret can be NA
      !is.na(dlret) ~ dlret,
      dlstcd %in% c(500, 520, 580, 584) |
        (dlstcd >= 551 & dlstcd <= 574) ~ -0.30,
```

```
      dlstcd == 100 ~ ret,
      TRUE ~ -1
    )
  ) |>
  select(-c(dlret, dlstcd))

# Calc excess return ----------------------------------------------------

factors_ff3_monthly <- tbl(db, "factors_ff3_monthly") |>
  select(month, rf) |>
  collect()

crsp_monthly <- crsp_monthly |>
  left_join(
    factors_ff3_monthly,
    by = "month"
  ) |>
  mutate(
    ret_excess = ret_adj - rf,
    ret_excess = pmax(ret_excess, -1)
  ) |>
  select(-ret_adj, -rf)
# na.omit() # TODO: better treatment?

# Write to SQLite -------------------------------------------------------

dbWriteTable(db, "crsp_monthly", crsp_monthly, overwrite = TRUE)

# Summary Stats ---------------------------------------------------------

crsp_monthly |>
  na.omit() |>
  sum_stats(c("ret", "ret_excess"))

# Trend of N
crsp_monthly |>
  group_by(month) |>
  summarize(n = n()) |>
  ggplot(aes(x = month, y = n)) +
  geom_line() +
  labs(
    title = "Number of observations",
    x = "Month",
    y = "Number of observations"
  )

# Total Market Cap
crsp_monthly |>
  group_by(month) |>
  summarize(mktcap = sum(mktcap, na.rm = TRUE)) |>
  ggplot(aes(x = month, y = mktcap)) +
  geom_line() +
  labs(
```

```
    x = "Month",
    y = "Total Market Cap (in million USD)"
  )
```

2.3 CRSP Daily

```
dsf_db <- tbl(wrds, in_schema("crsp", "dsf"))

factors_ff3_daily <- tbl(db, "factors_ff3_daily") |>
  collect()

permnos <- tbl(db, "crsp_monthly") |>
  distinct(permno) |>
  pull()

# Determine the number of chunks
chunk_size <- 200
num_chunks <- ceiling(length(permnos) / chunk_size)

# Progress bar using progress package
pb <- progress_bar$new(
  format = "[:bar] :percent eta: :eta",
  total = num_chunks
)

for (j in 1:num_chunks) {
  # Select the permnos for this chunk
  permno_chunk <- permnos[((j - 1) * chunk_size + 1):min(j * chunk_size, length(permnos))]

  # Process all permnos in the chunk at once
  crsp_daily_sub <- dsf_db |>
    filter(permno %in% permno_chunk &
      date >= start_date & date <= end_date) |>
    select(permno, date, ret) |>
    collect() |>
    drop_na()

  if (nrow(crsp_daily_sub) > 0) {
    crsp_daily_sub <- crsp_daily_sub |>
      mutate(month = floor_date(date, "month")) |>
      left_join(factors_ff3_daily |>
        select(date, rf), by = "date") |>
      mutate(
        ret_excess = ret - rf,
        ret_excess = pmax(ret_excess, -1)
      ) |>
      select(permno, date, month, ret, ret_excess)

    dbWriteTable(db,
      "crsp_daily",
      value = crsp_daily_sub,
      overwrite = ifelse(j == 1, TRUE, FALSE),
      append = ifelse(j != 1, TRUE, FALSE)
```

```
    )
  }

  pb$tick()
}

dbDisconnect(wrds)

# Summary Stats ------------------------------------------------------------
crsp_daily <- tbl(db, "crsp_daily") |>
  select(permno, date, month, ret_excess) |>
  collect() |>
  drop_na()

crsp_daily |>
  sum_stats(c("ret_excess"))

# Trend of N along time-series
crsp_daily |>
  group_by(date) |>
  summarise(n = n()) |>
  ggplot(aes(x = date, y = n)) +
  geom_line() +
  labs(x = "Month", y = "Number of observations")
```

2.4 COMPUSTAT

```
funda_db <- tbl(wrds, in_schema("comp", "funda"))

compustat <- funda_db |>
  filter(
    indfmt == "INDL" &
      datafmt == "STD" &
      consol == "C" &
      datadate >= start_date & datadate <= end_date
  ) |>
  select(
    gvkey, # Firm identifier
    datadate, # Date of the accounting data
    seq, # Stockholders' equity
    ceq, # Total common/ordinary equity
    at, # Total assets
    lt, # Total liabilities
    txditc, # Deferred taxes and investment tax credit
    txdb, # Deferred taxes
    itcb, # Investment tax credit
    pstkrv, # Preferred stock redemption value
    pstkl, # Preferred stock liquidating value
    pstk, # Preferred stock par value
    capx, # Capital investment
    oancf, # Operating cash flow
    sale, # Revenue
    cogs, # Costs of goods sold
```

```
    xint, # Interest expense
    xsga # Selling, general, and administrative expenses
  ) |>
  collect()

compustat <- compustat |>
  mutate(
    be = coalesce(seq, ceq + pstk, at - lt) +
      coalesce(txditc, txdb + itcb, 0) -
      coalesce(pstkrv, pstkl, pstk, 0),
    be = if_else(be <= 0, as.numeric(NA), be),
    op = (sale - coalesce(cogs, 0) -
      coalesce(xsga, 0) - coalesce(xint, 0)) / be,
  )

compustat <- compustat |>
  mutate(year = year(datadate)) |>
  group_by(gvkey, year) |>
  filter(datadate == max(datadate)) |>
  ungroup()

compustat <- compustat |>
  left_join(
    compustat |>
      select(gvkey, year, at_lag = at) |>
      mutate(year = year + 1),
    by = c("gvkey", "year")
  ) |>
  mutate(
    inv = at / at_lag - 1,
    inv = if_else(at_lag <= 0, as.numeric(NA), inv)
  )

db <- dbConnect(
  SQLite(),
  "data/main.sqlite",
  extended_types = TRUE
)

dbWriteTable(
  db,
  "compustat",
  value = compustat,
  overwrite = TRUE
)

# Merge with CRSP ---------------------------------------------------------

ccmxpf_linktable_db <- tbl(
  wrds,
  in_schema("crsp", "ccmxpf_linktable")
)
```

```r
ccmxpf_linktable <- ccmxpf_linktable_db |>
  filter(linktype %in% c("LU", "LC") &
    linkprim %in% c("P", "C") &
    usedflag == 1) |>
  select(permno = lpermno, gvkey, linkdt, linkenddt) |>
  collect() |>
  mutate(linkenddt = replace_na(linkenddt, today()))

crsp_monthly <- tbl(db, "crsp_monthly") |>
  collect()

ccm_links <- crsp_monthly |>
  inner_join(ccmxpf_linktable,
    by = "permno", relationship = "many-to-many"
  ) |>
  filter(!is.na(gvkey) &
    (date >= linkdt & date <= linkenddt)) |>
  select(permno, gvkey, date)

crsp_monthly <- crsp_monthly |>
  left_join(ccm_links, by = c("permno", "date"))

dbWriteTable(
  db,
  "crsp_monthly",
  value = crsp_monthly,
  overwrite = TRUE
)


# EDA ----------------------------------------------------------------------

compustat <- tbl(db, "compustat") |>
  collect()

crsp_monthly <- tbl(db, "crsp_monthly") |>
  collect()

crsp_monthly |>
  mutate(exchange = case_when(
    exchcd %in% c(1, 31) ~ "NYSE",
    exchcd %in% c(2, 32) ~ "AMEX",
    exchcd %in% c(3, 33) ~ "NASDAQ",
    .default = "Other"
  )) |>
  group_by(permno, year = year(month)) |>
  filter(date == max(date)) |>
  ungroup() |>
  left_join(compustat, by = c("gvkey", "year")) |>
  group_by(exchange, year) |>
  summarize(
    share = n_distinct(permno[!is.na(be)]) / n_distinct(permno),
    .groups = "drop"
```

```
  ) |>
  ggplot(aes(
    x = year,
    y = share,
    color = exchange,
    linetype = exchange
  )) +
  geom_line() +
  labs(
    x = NULL, y = NULL, color = NULL, linetype = NULL,
    title = "Share of securities with book equity values by exchange"
  ) +
  scale_y_continuous(labels = percent) +
  coord_cartesian(ylim = c(0, 1))
```

2.5 Fama-French Factors

```
# FF [Monthly] ------------------------------------------------------------
factors_ff3_monthly_raw <- download_french_data("Fama/French 3 Factors")

factors_ff3_monthly <- factors_ff3_monthly_raw$subsets$data[[1]] |>
  mutate(
    month = floor_date(ymd(str_c(date, "01")), "month"),
    across(c(RF, `Mkt-RF`, SMB, HML), ~ as.numeric(.) / 100),
    .keep = "none"
  ) |>
  rename_with(str_to_lower) |>
  rename(mkt_excess = `mkt-rf`) |>
  filter(month >= start_date & month <= end_date)

dbWriteTable(db, "factors_ff3_monthly", factors_ff3_monthly, overwrite = TRUE)

# FF [Daily] --------------------------------------------------------------
factors_ff3_daily_raw <- download_french_data("Fama/French 3 Factors [Daily]")

factors_ff3_daily <- factors_ff3_daily_raw$subsets$data[[1]] |>
  mutate(
    date = ymd(date),
    across(c(RF, `Mkt-RF`, SMB, HML), ~ as.numeric(.) / 100),
    .keep = "none"
  ) |>
  rename_with(str_to_lower) |>
  rename(mkt_excess = `mkt-rf`) |>
  filter(date >= start_date & date <= end_date)

dbWriteTable(db, "factors_ff3_daily", factors_ff3_daily, overwrite = TRUE)
```

3 Calculate Risk Loadings

```
# Load data ---------------------------------------------------------------
crsp_daily <- tbl(db, "crsp_daily") |>
  select(permno, date, month, ret_excess) |>
```

```
  collect() |>
  drop_na()

factors_ff3_daily <- tbl(db, "factors_ff3_daily") |>
  select(date, mkt_excess, smb, hml) |>
  collect()

# Join the two
crsp_daily <- crsp_daily |>
  left_join(factors_ff3_daily, by = c("date")) |>
  select(permno, date, month, ret_excess, mkt_excess, smb, hml)

# Nest by permno
crsp_daily_nested <- crsp_daily |>
  nest(rets = c(date, month, ret_excess, mkt_excess, smb, hml))

# Functions ------------------------------------------------------------

# Data is a time-series of returns of a single estimation period
estimate_capm <- function(data, min_obs = 1) {
  if (nrow(data) < min_obs) {
    return(list(alpha = NA, beta_mkt = NA, beta_smb = NA, beta_hml = NA, res_std = NA))
  } else {
    fit <- lm(ret_excess ~ mkt_excess + smb + hml, data = data)
    c <- coefficients(fit)
    return(list(alpha = c[1], beta_mkt = c[2], beta_smb = c[3], beta_hml = c[4], res_std = sd(residuals
  }
}

# Data is a stock's time-series returns for which to calculate betas
roll_capm_estimation <- function(data, months, min_obs) {
  data <- data |>
    arrange(month)

  betas <- slide_period_dfr(
    .x = data,
    .i = data$month, # index
    .period = "month", # aggregation is applied to each month (monthly betas)
    .f = ~ estimate_capm(., min_obs),
    .before = months - 1,
    .complete = FALSE # ignore incomplete periods
  )

  betas$month <- unique(data$month)

  return(betas)
}

# Sanity Check ---------------------------------------------------------

examples <- tribble(
  ~permno, ~company,
  14593, "Apple",
```

```
  10107, "Microsoft",
  93436, "Tesla",
  17778, "Berkshire Hathaway"
)

examples_beta <- crsp_daily_nested |>
  inner_join(examples, by = "permno") |>
  mutate(betas = map(
    rets,
    ~ roll_capm_estimation(., months = 1, min_obs = 17)
  )) |>
  unnest(betas) |>
  unnest(c(alpha, beta_mkt, beta_smb, beta_hml, res_std)) |>
  drop_na()

examples_beta

examples_beta |>
  ggplot(aes(
    x = month,
    y = beta_mkt,
    color = company,
    linetype = company
  )) +
  geom_line() +
  labs(
    x = NULL, y = NULL, color = NULL, linetype = NULL,
  )

# Run ---------------------------------------------------------------------

plan(multisession, workers = 8)

specs <- tibble(
  periods = c(1, 3, 6, 12, 24),
  min_obs = c(17, 51, 102, 204, 408)
)

for (i in 1:nrow(specs)) {
  with_progress({
    p <- progressor(steps = nrow(crsp_daily_nested))
    print(paste0("Running ", specs$periods[i], " month betas"))

    betas <- crsp_daily_nested |>
      mutate(betas = future_map(
        rets,
        ~ {
          p()
          roll_capm_estimation(., months = specs$periods[i], min_obs = specs$min_obs[i])
        }
      )) |>
      unnest(betas) |>
      unnest(c(alpha, beta_mkt, beta_smb, beta_hml, res_std)) |>
```

```
      drop_na() |>
      select(-rets) |>
      relocate(permno, month)
  })

  dbWriteTable(db, paste0("betas_ff3_", specs$periods[i], "m"), betas, overwrite = TRUE)
}

# Calc Summary Stats for BETA_MKT ----------------------------------------

# Join beta_mkt from different specs to a single data frame
for (i in 1:nrow(specs)) {
  if (i == 1) {
    betas_all <- tbl(db, paste0("betas_ff3_", specs$periods[i], "m")) |>
      select(permno, month, beta_mkt) |>
      collect() |>
      rename_with(~ paste0(specs$periods[i], "m"), beta_mkt)

    next
  }

  betas_all <- betas_all |>
    left_join(
      tbl(db, paste0("betas_ff3_", specs$periods[i], "m")) |> select(permno, month, beta_mkt) |> collec
      by = c("permno", "month")
    ) |>
    rename_with(~ paste0(specs$periods[i], "m"), beta_mkt)
}

# Summary Stats
source("r/utils.R")

beta_stats <- sum_stats(betas_all |> rename(date = month), c("1m", "3m", "6m", "12m", "24m"))

# Correlation Matrix
beta_cor_mat_p <- betas_all |>
  drop_na() |>
  select(-permno, -month) |>
  cor(method = "p")

beta_cor_mat_s <- betas_all |>
  drop_na() |>
  select(-permno, -month) |>
  cor(method = "s")

# Merge the two, place Spearman in the upper triangle and Pearson in the lower
beta_cor_mat <- beta_cor_mat_p
beta_cor_mat[upper.tri(beta_cor_mat, diag = TRUE)] <- beta_cor_mat_s[upper.tri(beta_cor_mat_s, diag = TR

# Persistence Matrix
lags <- c(1, 3, 6, 12, 24, 36, 48, 60, 120)
persistences <- tibble()
```

```r
for (i in 1:length(lags)) {
  betas_lag <- betas_all |>
    mutate(month = month %m+% months(lags[i])) |>
    rename(`1m_lag` = `1m`, `3m_lag` = `3m`, `6m_lag` = `6m`, `12m_lag` = `12m`, `24m_lag` = `24m`)

  tmp <- betas_all |>
    left_join(betas_lag, by = c("permno", "month"))

  tmp <- tmp |>
    group_by(month) |>
    drop_na() |>
    summarise(
      cor_1m = cor(`1m`, `1m_lag`),
      cor_3m = cor(`3m`, `3m_lag`),
      cor_6m = cor(`6m`, `6m_lag`),
      cor_12m = cor(`12m`, `12m_lag`),
      cor_24m = cor(`24m`, `24m_lag`)
    ) |>
    summarise(across(starts_with("cor"), mean))

  tmp$lag <- lags[i]

  persistences <- bind_rows(persistences, tmp)
}

persistences <- persistences |> relocate(lag)

# Replace with NA for upper triangular due to autocorrelation
persistences[2:6][upper.tri(persistences[2:6], diag = FALSE)] <- NA

# Stargazer ----------------------------------------------------------------

beta_stats %>%
  mutate_if(is.numeric, round, 2) %>%
  mutate(var = substr(var, 1, nchar(var) - 1)) %>%
  stargazer(
    type = "latex",
    summary = FALSE,
    align = TRUE,
    header = FALSE,
    title = "Beta Summary Statistics",
    rownames = FALSE,
    covariate.labels = c("Months", "$\\mu$", "$\\sigma$"),
    label = "tab:beta_summary_stats"
  ) %>%
  as.character() %>%
  cat(file = "report/tabs/beta_summary_stats.tex", sep = "\n")

diag(beta_cor_mat) <- NA
beta_cor_mat %>%
  round(2) %>%
  stargazer(
    type = "latex",
```

```
    summary = FALSE,
    align = TRUE,
    header = FALSE,
    title = "Beta Correlation Matrix",
    label = "tab:beta_cor_mat"
  ) %>%
  as.character() %>%
  cat(file = "report/tabs/beta_cor_mat.tex", sep = "\n")

persistences %>%
  mutate_if(is.numeric, round, 2) %>%
  # convert to string, NA is empty
  mutate(across(starts_with("cor"), as.character)) %>%
  stargazer(
    type = "latex",
    summary = FALSE,
    align = TRUE,
    header = FALSE,
    title = "Beta Persistence",
    label = "tab:beta_persistence",
    rownames = FALSE,
    covariate.labels = c("Lag", "$\\beta^{1M}$", "$\\beta^{3M}$", "$\\beta^{6M}$", "$\\beta^{12M}$", "$
  ) %>%
  as.character() %>%
  cat(file = "report/tabs/beta_persistence.tex", sep = "\n")
```

4 Link Size, BM, and MOM

```
# Load Data ---------------------------------------------------------------

crsp_monthly <- tbl(db, "crsp_monthly") |>
  select(permno, month, mktcap, ret, gvkey) |>
  collect()

compustat <- tbl(db, "compustat") |>
  select(gvkey, year, be) |>
  collect()

# Annual Size -------------------------------------------------------------

# `Size` is `mktcap` at June of year t
# `ME` (market equity) is `mktcap` at December of year t-1
factors <- crsp_monthly |>
  group_by(permno, year = year(month)) |>
  mutate(size = if_else(month(month) == 6, mktcap, NA_real_)) |> # size at June (t)
  mutate(me = if_else(month(month) == 12, mktcap, NA_real_)) |> # me at December (t)
  summarise(
    size = last(size, na_rm = TRUE),
    me = last(me, na_rm = TRUE),
    gvkey = last(gvkey, na_rm = TRUE)
  )
```

```r
# Shift `me` to t-1
factors <- factors |>
  left_join(
    factors |> select(permno, year, me_lagged = me) |> mutate(year = year + 1),
    by = c("permno", "year")
  ) |>
  select(-me) |>
  rename(me = me_lagged)

# Annual BM ------------------------------------------------------------------

factors <- factors |>
  left_join(
    compustat |> mutate(year = year + 1),
    by = c("gvkey", "year")
  ) |>
  mutate(bm = be / me) |>
  select(-be, -me, -gvkey)

# Monthly MOM ----------------------------------------------------------------

cumret <- function(data, min_obs = 5) {
  if (nrow(data) < min_obs) {
    return(NA_real_)
  } else {
    return(prod(1 + data$ret) - 1)
  }
}

roll_mom_estimation <- function(data, months, min_obs) {
  slide_period_dbl(
    .x = data,
    .i = data$month, # index
    .period = "month", # aggregation is applied to each month
    .f = ~ cumret(., min_obs),
    .before = months - 1,
    .complete = FALSE # ignore incomplete periods
  )
}

# Sanity Check
examples <- tribble(
  ~permno, ~company,
  14593, "Apple",
  10107, "Microsoft",
  93436, "Tesla",
  17778, "Berkshire Hathaway"
)

t <- crsp_monthly %>%
  inner_join(examples, by = "permno") %>%
  group_by(permno) %>%
  arrange(month) %>%
```

```
  group_modify(~ mutate(., mom = roll_mom_estimation(., months = 6, min_obs = 5))) %>%
  ungroup()

# Calculate MOM
mom_monthly <- crsp_monthly %>%
  group_by(permno) %>%
  arrange(month) %>%
  group_modify(~ mutate(., mom = roll_mom_estimation(., months = 6, min_obs = 5))) %>%
  ungroup()

# Shift `mom` to t-1
mom_monthly <- mom_monthly |>
  left_join(
    mom_monthly |>
      group_by(permno) |>
      mutate(month = month %m+% months(1)) |>
      ungroup() |>
      select(permno, month, mom_lagged = mom),
    by = c("permno", "month")
  ) |>
  select(-mom) |>
  rename(mom = mom_lagged)

# Merge all factors ------------------------------------------------------

all_factors <- mom_monthly |>
  mutate(year = year(month)) |>
  left_join(
    factors,
    by = c("permno", "year")
  ) |>
  select(permno, month, mom, size, bm)

# Shift size by 5 months
all_factors <- all_factors |>
  left_join(
    all_factors |>
      group_by(permno) |>
      mutate(month = month %m+% months(5)) |>
      ungroup() |>
      select(permno, month, size_lagged = size),
    by = c("permno", "month")
  ) |>
  select(-size) |>
  rename(size = size_lagged)

# Save -------------------------------------------------------------------

dbWriteTable(
  db,
  "factors_size_bm_mom",
  all_factors,
  overwrite = TRUE
```

```
)
```

5 Portfolio Sort

```
# Load Data ------------------------------------------------------------------

crsp_monthly <- tbl(db, "crsp_monthly") |>
  select(permno, month, mktcap, ret_excess, exchcd, prc) |>
  collect()

factors <- tbl(db, "factors_size_bm_mom") |>
  select(permno, month, size, bm, mom) |>
  collect()

factors_ff3_monthly <- tbl(db, "factors_ff3_monthly") |>
  select(month, mkt_excess) |>
  collect()

data <- crsp_monthly |>
  inner_join(factors, by = c("permno", "month")) |>
  drop_na()

# Add lag variables
data <- data |>
  left_join(
    data |> select(permno, month, size_lag = size, bm_lag = bm, mom_lag = mom, mktcap_lag = mktcap) |> r
    by = c("permno", "month")
  )

# Functions ------------------------------------------------------------------

assign_portfolio <- function(
    data,
    sort_variable,
    n_portfolios,
    sort_data = data,  # New argument with default value as 'data'
    probs = seq(0, 1, length.out = n_portfolios + 1)  # New argument with default probability sequence
) {
  # Compute breakpoints using 'sort_data'
  breakpoints <- sort_data |>
    pull({{ sort_variable }}) |>
    quantile(
      probs = probs,  # Use the provided 'prob' argument
      na.rm = TRUE,
      names = FALSE
    )

  # Assign portfolios
  assigned_portfolios <- data |>
    mutate(portfolio = findInterval(
      pick(everything()) |> pull({{ sort_variable }}),
      breakpoints,
```

```
      all.inside = TRUE
    )) |>
    pull(portfolio)

  return(assigned_portfolios)
}

single_sort <- function(
    data, # panel data
    sort_variable,
    n_portfolios,
    sort_data = data,
    probs = seq(0, 1, length.out = n_portfolios + 1)
){
  portfolios <- data |>
    drop_na() |>
    group_by(month) |>
    mutate(
      portfolio = assign_portfolio(
        data = pick(everything()),
        sort_variable = {{ sort_variable }},
        n_portfolios = n_portfolios,
        sort_data = pick(everything()) %>% filter(exchcd %in% c(1, 31)),
        probs = probs
      ) %>%
        as.factor()
    ) |>
    group_by(portfolio, month) |> # cross-sectional
    summarize(
      ew_ret = mean(ret_excess),
      vw_ret = weighted.mean(ret_excess, mktcap_lag),
      .groups = "drop"
    ) |>
    group_by(portfolio) |> # time-series average
    summarize(
      avg_ew_ret = mean(ew_ret),
      ew_tstat = {
        lm_model <- lm(ew_ret ~ 1)
        summary_model <- summary(lm_model, vcov = NeweyWest(lm_model))
        summary_model$coefficients["(Intercept)", "t value"]
      },
      avg_vw_ret = mean(vw_ret),
      vw_tstat = {
        lm_model <- lm(vw_ret ~ 1)
        summary_model <- summary(lm_model, vcov = NeweyWest(lm_model))
        summary_model$coefficients["(Intercept)", "t value"]
      },
      .groups = "drop"
    )

  return (portfolios)
}
```

```r
indepedent_double_sort <- function(
    data, # panel data
    sort_variable_1,
    sort_variable_2,
    n_portfolios_1,
    n_portfolios_2 = n_portfolios_1,
    sort_data = data
){
  portfolios <- data |>
    drop_na() |>
    group_by(month) |>
    mutate(
      portfolio_1 = assign_portfolio(
        data = pick(everything()),
        sort_variable = {{ sort_variable_1 }},
        n_portfolios = n_portfolios_1,
        sort_data = pick(everything()) %>% filter(exchcd %in% c(1, 31))
      ) %>%
        as.factor(),
      portfolio_2 = assign_portfolio(
        data = pick(everything()),
        sort_variable = {{ sort_variable_2 }},
        n_portfolios = n_portfolios_2,
        sort_data = pick(everything()) %>% filter(exchcd %in% c(1, 31))
      ) %>%
        as.factor()
    ) |>
    group_by(portfolio_1, portfolio_2, month) |> # cross-sectional
    summarize(
      ew_ret = mean(ret_excess),
      vw_ret = weighted.mean(ret_excess, mktcap_lag),
      .groups = "drop"
    ) |>
    group_by(portfolio_1, portfolio_2) |> # time-series average
    summarize(
      avg_ew_ret = mean(ew_ret),
      ew_tstat = {
        lm_model <- lm(ew_ret ~ 1)
        summary_model <- summary(lm_model, vcov = NeweyWest(lm_model))
        summary_model$coefficients["(Intercept)", "t value"]
      },
      avg_vw_ret = mean(vw_ret),
      vw_tstat = {
        lm_model <- lm(vw_ret ~ 1)
        summary_model <- summary(lm_model, vcov = NeweyWest(lm_model))
        summary_model$coefficients["(Intercept)", "t value"]
      },
      .groups = "drop"
    )

  return (portfolios)
}
```

```r
dependent_double_sort <- function(
    data, # panel data
    sort_variable_1,
    sort_variable_2,
    n_portfolios_1,
    n_portfolios_2 = n_portfolios_1,
    sort_data = data
){
  portfolios <- data |>
    drop_na() |>
    group_by(month) |>
    mutate(
      portfolio_1 = assign_portfolio(
        data = pick(everything()),
        sort_variable = {{ sort_variable_1 }},
        n_portfolios = n_portfolios_1,
        sort_data = pick(everything()) %>% filter(exchcd %in% c(1, 31))
      ) %>%
        as.factor()
    ) |>
    group_by(portfolio_1) |>
    mutate(
      portfolio_2 = assign_portfolio(
        data = pick(everything()),
        sort_variable = {{ sort_variable_2 }},
        n_portfolios = n_portfolios_2,
        sort_data = pick(everything()) %>% filter(exchcd %in% c(1, 31))
      ) %>%
        as.factor()
    ) |>
    ungroup() |>
    group_by(portfolio_1, portfolio_2, month) |> # cross-sectional
    summarize(
      ew_ret = mean(ret_excess),
      vw_ret = weighted.mean(ret_excess, mktcap_lag),
      .groups = "drop"
    ) |>
    group_by(portfolio_1, portfolio_2) |> # time-series average
    summarize(
      avg_ew_ret = mean(ew_ret),
      ew_tstat = {
        lm_model <- lm(ew_ret ~ 1)
        summary_model <- summary(lm_model, vcov = NeweyWest(lm_model))
        summary_model$coefficients["(Intercept)", "t value"]
      },
      avg_vw_ret = mean(vw_ret),
      vw_tstat = {
        lm_model <- lm(vw_ret ~ 1)
        summary_model <- summary(lm_model, vcov = NeweyWest(lm_model))
        summary_model$coefficients["(Intercept)", "t value"]
      },
      .groups = "drop"
    )
```

```
  return (portfolios)
}

# Single Sort
single_sort(data, size_lag, 10)
single_sort(data, bm_lag, 10)

mom_data <- data |>
  left_join(
    data |>
      filter(exchcd %in% c(1, 31)) |>
      group_by(month) |>
      summarize(
        size_10th = quantile(size_lag, probs = 0.1, na.rm = TRUE),
        .groups = "drop"
      ), # NYSE breakpoints
    by = "month"
  ) |>
  filter(size >= size_10th) |>
  filter(prc > 5) |>
  select(-size_10th)
single_sort(mom_data, mom_lag, 5)

# Double Sort
indepedent_double_sort(data, size_lag, bm_lag, n_portfolios_1 =5, n_portfolios_2 = 5)
dependent_double_sort(data, size_lag, bm_lag, n_portfolios_1 =5, n_portfolios_2 = 5)
```

6 Fama-MacBeth Regression

```
crsp_monthly <- tbl(db, "crsp_monthly") |>
  select(permno, month, ret_excess) |>
  collect()

factors <- tbl(db, "factors_size_bm_mom") |>
  select(permno, month, size, bm, mom) |>
  collect()

betas <- tbl(db, "betas_ff3_1m") |>
  select(permno, month, ivol = res_std) |>
  collect()

data <- crsp_monthly |>
  inner_join(factors, by = c("permno", "month")) |>
  inner_join(betas, by = c("permno", "month")) |>
  inner_join(crsp_monthly |> select(permno, month, ret_lead = ret_excess) |> mutate(month = month %m-% m
  drop_na() |>
  # winsorize at 0.5% each month
  group_by(month) |>
  mutate(
    size = Winsorize(size, probs = c(0.005, 0.995)),
    bm = Winsorize(bm, probs = c(0.005, 0.995)),
    mom = Winsorize(mom, probs = c(0.005, 0.995)),
```

```
    ivol = Winsorize(ivol, probs = c(0.005, 0.995))
  ) |>
  ungroup() |>
  nest(cross_section = c(permno, ret_lead, size, bm, mom, ret_excess, ivol))

# Define the function
fm <- function(data, independent_vars) {
  # Create a formula string for the regression model
  formula_str <- paste('ret_lead', "~", paste(independent_vars, collapse = " + "))

  data %>%
    mutate(estimates = map(
      cross_section,
      ~ tidy(lm(as.formula(formula_str), data = .x))
    )) %>%
    unnest(estimates) %>%
    select(month, factor = term, estimate) %>%
    nest(time_series = c(month, estimate)) %>%
    mutate(
      model = map(time_series, ~ lm(estimate ~ 1, .)),
      result = map(model, tidy)
    ) %>%
    mutate(newey_west_se = map_dbl(model, ~ sqrt(NeweyWest(.)))) %>%
    unnest(result) %>%
    mutate(t_stat = estimate / newey_west_se) %>%
    select(factor, estimate, t_stat)
}

fm(data, c("bm", "size"))
```