Relative Performance Evaluation and Competitive Aggressiveness

Setup

Load required packages

```
rm(list = ls())

lapply(c("tidyverse", "lubridate", "broom", "RPostgres"), library, character.only = TRUE)

summarize <- dplyr::summarize
select <- dplyr::select
rename <- dplyr::rename

wrds <- dbConnect(Postgres(),
                  host = 'wrds-pgdata.wharton.upenn.edu',
                  user = '...', # add WRDS username
                  password = '...', # add WRDS password
                  port = 9737,
                  dbname = 'wrds',
                  sslmode ='require')
```

```
##### FUNCTIONS #####
### WINSORIZE FUNCTION ###
winsorize <- function(object, lower_percentile, upper_percentile, exclude = NULL){

  if (is.data.frame(object) == TRUE){

    reference_columns <- names(object)

    numeric_columns <- sapply(object, is.numeric)
    exclude_columns <- if(!is.null(exclude)) exclude %>% as.vector() %>% as.character() else vec
tor() %>% as.character()

    numeric_subset <- object[ , numeric_columns]
    nonnumeric_subset <- cbind(object[ , !numeric_columns], object[ , exclude_columns])

    winsorized_variables <- numeric_subset %>% select(- one_of(exclude_columns))

    columns <- ncol(winsorized_variables)
    rows <- nrow(winsorized_variables)

    for (i in 1:columns){
      L <- (lower_percentile / 100)
      U <- (upper_percentile / 100)
      percentile <- as.matrix(quantile(winsorized_variables[[i]], c(L, U) , na.rm = T))
      min <- percentile[1,1]
      max <- percentile[2,1]

      for (j in 1:rows){
        winsorized_variables[j, i] <- if (is.na(winsorized_variables[j, i]) | is.infinite(winsor
ized_variables[j, i])) NA else
          if (winsorized_variables[j, i] < min) min else if (winsorized_variables[j, i] > max) m
ax else winsorized_variables[j, i]
      }
    }

    final_output <- cbind(nonnumeric_subset, winsorized_variables)
    final_output <- final_output[, c(reference_columns, setdiff(names(final_output), reference_c
olumns))]

    cat("\n", "In total,", columns, "variables were winsorized:", colnames(winsorized_variable
s))

    return(final_output)
  }

  else if (is.vector(object) == TRUE){
    L <- (lower_percentile / 100)
    U <- (upper_percentile / 100)
    percentile <- as.matrix(quantile(object, c(L, U) , na.rm = T))
    min <- percentile[1,1]
    max <- percentile[2,1]
    for (i in 1:length(object)){
```

```
      object[i] <- if (is.na(object[i]) | is.infinite(object[i])) NA else
        if (object[i] < min) min else if (object[i] > max) max else object[i]
    }
    return(object)
  }
}

### STD FUNCTION ###
std.function <- function(x) {
  x <- as.data.frame(x)
  c(FYEAR = tail(x$FYEAR, 1), SD_RET = sd(x$RET, na.rm = T), SD_ROA = sd(x$ROA, na.rm = T))
}

### FAMA AND FRENCH INDUSTRIES ###
sic_48 <- function (X, Y){
  m = nrow(IDs)
  for (i in 1:m)
    if (X[i] >= 100 & X[i] <= 799) {
      Y[i] <- 1
    } else if (X[i] >= 2048 & X[i] <= 2048) {
      Y[i] <- 1
    } else if (X[i] >= 2000 & X[i] <= 2046) {
      Y[i] <- 2
    } else if (X[i] >= 2050 & X[i] <= 2063) {
      Y[i] <- 2
    } else if (X[i] >= 2070 & X[i] <= 2079) {
      Y[i] <- 2
    } else if (X[i] >= 2090 & X[i] <= 2095) {
      Y[i] <- 2
    } else if (X[i] >= 2098 & X[i] <= 2099) {
      Y[i] <- 2
    } else if (X[i] >= 2064 & X[i] <= 2068) {
      Y[i] <- 3
    } else if (X[i] >= 2086 & X[i] <= 2087) {
      Y[i] <- 3
    } else if (X[i] >= 2096 & X[i] <= 2097) {
      Y[i] <- 3
    } else if (X[i] >= 2080 & X[i] <= 2085) {
      Y[i] <- 4
    } else if (X[i] >= 2100 & X[i] <= 2199) {
      Y[i] <- 5
    } else if (X[i] >= 0900 & X[i] <= 0999) {
      Y[i] <- 6
    } else if (X[i] >= 3650 & X[i] <= 3652) {
      Y[i] <- 6
    } else if (X[i] >= 3732 & X[i] <= 3732) {
      Y[i] <- 6
    } else if (X[i] >= 3930 & X[i] <= 3949) {
      Y[i] <- 6
    } else if (X[i] >= 7800 & X[i] <= 7841) {
      Y[i] <- 7
    } else if (X[i] >= 7900 & X[i] <= 7999) {
```

```
    Y[i] <- 7
  } else if (X[i] >= 2700 & X[i] <= 2749) {
    Y[i] <- 8
  } else if (X[i] >= 2770 & X[i] <= 2799) {
    Y[i] <- 8
  } else if (X[i] >= 2047 & X[i] <= 2047) {
    Y[i] <- 9
  } else if (X[i] >= 2391 & X[i] <= 2392) {
    Y[i] <- 9
  } else if (X[i] >= 2510 & X[i] <= 2519) {
    Y[i] <- 9
  } else if (X[i] >= 2590 & X[i] <= 2599) {
    Y[i] <- 9
  } else if (X[i] >= 2840 & X[i] <= 2844) {
    Y[i] <- 9
  } else if (X[i] >= 3160 & X[i] <= 3199) {
    Y[i] <- 9
  } else if (X[i] >= 3229 & X[i] <= 3231) {
    Y[i] <- 9
  } else if (X[i] >= 3260 & X[i] <= 3260) {
    Y[i] <- 9
  } else if (X[i] >= 3262 & X[i] <= 3263) {
    Y[i] <- 9
  } else if (X[i] >= 3269 & X[i] <= 3269) {
    Y[i] <- 9
  } else if (X[i] >= 3630 & X[i] <= 3639) {
    Y[i] <- 9
  } else if (X[i] >= 3750 & X[i] <= 3751) {
    Y[i] <- 9
  } else if (X[i] >= 3800 & X[i] <= 3800) {
    Y[i] <- 9
  } else if (X[i] >= 3860 & X[i] <= 3879) {
    Y[i] <- 9
  } else if (X[i] >= 3910 & X[i] <= 3919) {
    Y[i] <- 9
  } else if (X[i] >= 3960 & X[i] <= 3961) {
    Y[i] <- 9
  } else if (X[i] >= 3630 & X[i] <= 3639) {
    Y[i] <- 9
  } else if (X[i] >= 3991 & X[i] <= 3991) {
    Y[i] <- 9
  } else if (X[i] >= 3995 & X[i] <= 3995) {
    Y[i] <- 9
  } else if (X[i] >= 2300 & X[i] <= 2390) {
    Y[i] <- 10
  } else if (X[i] >= 3020 & X[i] <= 3021) {
    Y[i] <- 10
  } else if (X[i] >= 3100 & X[i] <= 3111) {
    Y[i] <- 10
  } else if (X[i] >= 3130 & X[i] <= 3159) {
    Y[i] <- 10
  } else if (X[i] >= 8000 & X[i] <= 8099) {
```

```
    Y[i] <- 11
  } else if (X[i] >= 3693 & X[i] <= 3693) {
    Y[i] <- 12
  } else if (X[i] >= 3840 & X[i] <= 3851) {
    Y[i] <- 12
  } else if (X[i] >= 2830 & X[i] <= 2836) {
    Y[i] <- 13
  } else if (X[i] >= 2800 & X[i] <= 2829) {
    Y[i] <- 14
  } else if (X[i] >= 2850 & X[i] <= 2899) {
    Y[i] <- 14
  } else if (X[i] >= 3000 & X[i] <= 3000) {
    Y[i] <- 15
  } else if (X[i] >= 3050 & X[i] <= 3099) {
    Y[i] <- 15
  } else if (X[i] >= 2200 & X[i] <= 2295) {
    Y[i] <- 16
  } else if (X[i] >= 2297 & X[i] <= 2299) {
    Y[i] <- 16
  } else if (X[i] >= 2393 & X[i] <= 2395) {
    Y[i] <- 16
  } else if (X[i] >= 2397 & X[i] <= 2399) {
    Y[i] <- 16
  } else if (X[i] >= 0800 & X[i] <= 0899) {
    Y[i] <- 17
  } else if (X[i] >= 2400 & X[i] <= 2439) {
    Y[i] <- 17
  } else if (X[i] >= 2450 & X[i] <= 2459) {
    Y[i] <- 17
  } else if (X[i] >= 2490 & X[i] <= 2499) {
    Y[i] <- 17
  } else if (X[i] >= 2950 & X[i] <= 2952) {
    Y[i] <- 17
  } else if (X[i] >= 3200 & X[i] <= 3219) {
    Y[i] <- 17
  } else if (X[i] >= 3240 & X[i] <= 3259) {
    Y[i] <- 17
  } else if (X[i] >= 3261 & X[i] <= 3261) {
    Y[i] <- 17
  } else if (X[i] >= 3264 & X[i] <= 3264) {
    Y[i] <- 17
  } else if (X[i] >= 3270 & X[i] <= 3299) {
    Y[i] <- 17
  } else if (X[i] >= 3420 & X[i] <= 3442) {
    Y[i] <- 17
  } else if (X[i] >= 3446 & X[i] <= 3452) {
    Y[i] <- 17
  } else if (X[i] >= 3490 & X[i] <= 3499) {
    Y[i] <- 17
  } else if (X[i] >= 3996 & X[i] <= 3996) {
    Y[i] <- 17
  } else if (X[i] >= 5000 & X[i] <= 5199) {
```

```
    Y[i] <- 18
  } else if (X[i] >= 5200 & X[i] <= 5736) {
    Y[i] <- 19
  } else if (X[i] >= 5900 & X[i] <= 5999) {
    Y[i] <- 19
  } else if (X[i] >= 5800 & X[i] <= 5813) {
    Y[i] <- 20
  } else if (X[i] >= 5890 & X[i] <= 5890) {
    Y[i] <- 20
  } else if (X[i] >= 7000 & X[i] <= 7019) {
    Y[i] <- 20
  } else if (X[i] >= 7040 & X[i] <= 7049) {
    Y[i] <- 20
  } else if (X[i] >= 7213 & X[i] <= 7213) {
    Y[i] <- 20
  } else if (X[i] >= 6000 & X[i] <= 6199) {
    Y[i] <- 21
  } else if (X[i] >= 6300 & X[i] <= 6411) {
    Y[i] <- 22
  } else if (X[i] >= 6500 & X[i] <= 6553) {
    Y[i] <- 23
  } else if (X[i] >= 6200 & X[i] <= 6299) {
    Y[i] <- 24
  } else if (X[i] >= 6700 & X[i] <= 6799) {
    Y[i] <- 24
  } else if (X[i] >= 1500 & X[i] <= 1549) {
    Y[i] <- 25
  } else if (X[i] >= 1600 & X[i] <= 1799) {
    Y[i] <- 25
  } else if (X[i] >= 3300 & X[i] <= 3369) {
    Y[i] <- 26
  } else if (X[i] >= 3390 & X[i] <= 3399) {
    Y[i] <- 26
  } else if (X[i] >= 3400 & X[i] <= 3400) {
    Y[i] <- 27
  } else if (X[i] >= 3443 & X[i] <= 3444) {
    Y[i] <- 27
  } else if (X[i] >= 3460 & X[i] <= 3479) {
    Y[i] <- 27
  } else if (X[i] >= 3510 & X[i] <= 3536) {
    Y[i] <- 28
  } else if (X[i] >= 3540 & X[i] <= 3569) {
    Y[i] <- 28
  } else if (X[i] >= 3580 & X[i] <= 3599) {
    Y[i] <- 28
  } else if (X[i] >= 3600 & X[i] <= 3621) {
    Y[i] <- 29
  } else if (X[i] >= 3623 & X[i] <= 3629) {
    Y[i] <- 29
  } else if (X[i] >= 3640 & X[i] <= 3646) {
    Y[i] <- 29
  } else if (X[i] >= 3648 & X[i] <= 3649) {
```

```
    Y[i] <- 29
  } else if (X[i] >= 3660 & X[i] <= 3660) {
    Y[i] <- 29
  } else if (X[i] >= 3690 & X[i] <= 3692) {
    Y[i] <- 29
  } else if (X[i] >= 3699 & X[i] <= 3699) {
    Y[i] <- 29
  } else if (X[i] >= 3900 & X[i] <= 3990) {
    Y[i] <- 30
  } else if (X[i] >= 3999 & X[i] <= 3999) {
    Y[i] <- 30
  } else if (X[i] >= 9900 & X[i] <= 9999) {
    Y[i] <- 30
  } else if (X[i] >= 2296 & X[i] <= 2296) {
    Y[i] <- 31
  } else if (X[i] >= 2396 & X[i] <= 2396) {
    Y[i] <- 31
  } else if (X[i] >= 3010 & X[i] <= 3011) {
    Y[i] <- 31
  } else if (X[i] >= 3537 & X[i] <= 3537) {
    Y[i] <- 31
  } else if (X[i] >= 3647 & X[i] <= 3647) {
    Y[i] <- 31
  } else if (X[i] >= 3694 & X[i] <= 3694) {
    Y[i] <- 31
  } else if (X[i] >= 3700 & X[i] <= 3716) {
    Y[i] <- 31
  } else if (X[i] >= 3790 & X[i] <= 3792) {
    Y[i] <- 31
  } else if (X[i] >= 3799 & X[i] <= 3799) {
    Y[i] <- 31
  } else if (X[i] >= 3720 & X[i] <= 3729) {
    Y[i] <- 32
  } else if (X[i] >= 3730 & X[i] <= 3731) {
    Y[i] <- 33
  } else if (X[i] >= 3740 & X[i] <= 3743) {
    Y[i] <- 33
  } else if (X[i] >= 3480 & X[i] <= 3489) {
    Y[i] <- 34
  } else if (X[i] >= 3760 & X[i] <= 3769) {
    Y[i] <- 34
  } else if (X[i] >= 3795 & X[i] <= 3795) {
    Y[i] <- 34
  } else if (X[i] >= 1040 & X[i] <= 1049) {
    Y[i] <- 35
  } else if (X[i] >= 1000 & X[i] <= 1039) {
    Y[i] <- 36
  } else if (X[i] >= 1060 & X[i] <= 1099) {
    Y[i] <- 36
  } else if (X[i] >= 1400 & X[i] <= 1499) {
    Y[i] <- 36
  } else if (X[i] >= 1200 & X[i] <= 1299) {
```

```
    Y[i] <- 37
  } else if (X[i] >= 1310 & X[i] <= 1389) {
    Y[i] <- 38
  } else if (X[i] >= 2900 & X[i] <= 2911) {
    Y[i] <- 38
  } else if (X[i] >= 2990 & X[i] <= 2999) {
    Y[i] <- 38
  } else if (X[i] >= 4900 & X[i] <= 4999) {
    Y[i] <- 39
  } else if (X[i] >= 4800 & X[i] <= 4899) {
    Y[i] <- 40
  } else if (X[i] >= 7020 & X[i] <= 7021) {
    Y[i] <- 41
  } else if (X[i] >= 7030 & X[i] <= 7039) {
    Y[i] <- 41
  } else if (X[i] >= 7200 & X[i] <= 7212) {
    Y[i] <- 41
  } else if (X[i] >= 7215 & X[i] <= 7299) {
    Y[i] <- 41
  } else if (X[i] >= 7395 & X[i] <= 7395) {
    Y[i] <- 41
  } else if (X[i] >= 7500 & X[i] <= 7500) {
    Y[i] <- 41
  } else if (X[i] >= 7520 & X[i] <= 7549) {
    Y[i] <- 41
  } else if (X[i] >= 7600 & X[i] <= 7699) {
    Y[i] <- 41
  } else if (X[i] >= 8100 & X[i] <= 8199) {
    Y[i] <- 41
  } else if (X[i] >= 8200 & X[i] <= 8499) {
    Y[i] <- 41
  } else if (X[i] >= 8600 & X[i] <= 8699) {
    Y[i] <- 41
  } else if (X[i] >= 8800 & X[i] <= 8899) {
    Y[i] <- 41
  } else if (X[i] >= 2750 & X[i] <= 2759) {
    Y[i] <- 42
  } else if (X[i] >= 3993 & X[i] <= 3993) {
    Y[i] <- 42
  } else if (X[i] >= 7300 & X[i] <= 7372) {
    Y[i] <- 42
  } else if (X[i] >= 7374 & X[i] <= 7394) {
    Y[i] <- 42
  } else if (X[i] >= 7397 & X[i] <= 7397) {
    Y[i] <- 42
  } else if (X[i] >= 7399 & X[i] <= 7399) {
    Y[i] <- 42
  } else if (X[i] >= 7510 & X[i] <= 7519) {
    Y[i] <- 42
  } else if (X[i] >= 8700 & X[i] <= 8748) {
    Y[i] <- 42
  } else if (X[i] >= 8900 & X[i] <= 8999) {
```

```
      Y[i] <- 42
    } else if (X[i] >= 3570 & X[i] <= 3579) {
      Y[i] <- 43
    } else if (X[i] >= 3680 & X[i] <= 3689) {
      Y[i] <- 43
    } else if (X[i] >= 3695 & X[i] <= 3695) {
      Y[i] <- 43
    } else if (X[i] >= 7373 & X[i] <= 7373) {
      Y[i] <- 43
    } else if (X[i] >= 3622 & X[i] <= 3622) {
      Y[i] <- 44
    } else if (X[i] >= 3661 & X[i] <= 3679) {
      Y[i] <- 44
    } else if (X[i] >= 3810 & X[i] <= 3810) {
      Y[i] <- 44
    } else if (X[i] >= 3812 & X[i] <= 3812) {
      Y[i] <- 44
    } else if (X[i] >= 3811 & X[i] <= 3811) {
      Y[i] <- 45
    } else if (X[i] >= 3820 & X[i] <= 3830) {
      Y[i] <- 45
    } else if (X[i] >= 2520 & X[i] <= 2549) {
      Y[i] <- 46
    } else if (X[i] >= 2600 & X[i] <= 2639) {
      Y[i] <- 46
    } else if (X[i] >= 2670 & X[i] <= 2699) {
      Y[i] <- 46
    } else if (X[i] >= 2760 & X[i] <= 2761) {
      Y[i] <- 46
    } else if (X[i] >= 3950 & X[i] <= 3955) {
      Y[i] <- 46
    } else if (X[i] >= 2440 & X[i] <= 2449) {
      Y[i] <- 47
    } else if (X[i] >= 2640 & X[i] <= 2659) {
      Y[i] <- 47
    } else if (X[i] >= 3210 & X[i] <= 3221) {
      Y[i] <- 47
    } else if (X[i] >= 3410 & X[i] <= 3412) {
      Y[i] <- 47
    } else if (X[i] >= 4000 & X[i] <= 4299) {
      Y[i] <- 48
    } else if (X[i] >= 4400 & X[i] <= 4499) {
      Y[i] <- 48
    } else if (X[i] >= 4500 & X[i] <= 4799) {
      Y[i] <- 48
    } else {
      Y[i] <- 0
    }
  IDs$FF_INDUSTRY_48 <<- Y
}

sic_12 <- function (X, Y){
```

```
m = nrow(IDs)
for (i in 1:m)
  if (X[i] >= 100 & X[i] <= 999) {
    Y[i] <- 1
  } else if (X[i] >= 2000 & X[i] <= 2399) {
    Y[i] <- 1
  } else if (X[i] >= 2700 & X[i] <= 2749) {
    Y[i] <- 1
  } else if (X[i] >= 2770 & X[i] <= 2799) {
    Y[i] <- 1
  } else if (X[i] >= 3100 & X[i] <= 3199) {
    Y[i] <- 1
  } else if (X[i] >= 3940 & X[i] <= 3989) {
    Y[i] <- 1
  } else if (X[i] >= 2500 & X[i] <= 2519) {
    Y[i] <- 2
  } else if (X[i] >= 2590 & X[i] <= 2599) {
    Y[i] <- 2
  } else if (X[i] >= 3630 & X[i] <= 3659) {
    Y[i] <- 2
  } else if (X[i] >= 3710 & X[i] <= 3711) {
    Y[i] <- 2
  } else if (X[i] >= 3714 & X[i] <= 3714) {
    Y[i] <- 2
  } else if (X[i] >= 3716 & X[i] <= 3716) {
    Y[i] <- 2
  } else if (X[i] >= 3750 & X[i] <= 3751) {
    Y[i] <- 2
  } else if (X[i] >= 3792 & X[i] <= 3792) {
    Y[i] <- 2
  } else if (X[i] >= 3900 & X[i] <= 3939) {
    Y[i] <- 2
  } else if (X[i] >= 3990 & X[i] <= 3999) {
    Y[i] <- 2
  } else if (X[i] >= 2520 & X[i] <= 2589) {
    Y[i] <- 3
  } else if (X[i] >= 2600 & X[i] <= 2699) {
    Y[i] <- 3
  } else if (X[i] >= 2750 & X[i] <= 2769) {
    Y[i] <- 3
  } else if (X[i] >= 3000 & X[i] <= 3099) {
    Y[i] <- 3
  } else if (X[i] >= 3200 & X[i] <= 3569) {
    Y[i] <- 3
  } else if (X[i] >= 3580 & X[i] <= 3629) {
    Y[i] <- 3
  } else if (X[i] >= 3700 & X[i] <= 3709) {
    Y[i] <- 3
  } else if (X[i] >= 3712 & X[i] <= 3713) {
    Y[i] <- 3
  } else if (X[i] >= 3715 & X[i] <= 3715) {
    Y[i] <- 3
```

```
  } else if (X[i] >= 3717 & X[i] <= 3749) {
    Y[i] <- 3
  } else if (X[i] >= 3752 & X[i] <= 3791) {
    Y[i] <- 3
  } else if (X[i] >= 3793 & X[i] <= 3799) {
    Y[i] <- 3
  } else if (X[i] >= 3830 & X[i] <= 3839) {
    Y[i] <- 3
  } else if (X[i] >= 3860 & X[i] <= 3899) {
    Y[i] <- 3
  } else if (X[i] >= 1200 & X[i] <= 1399) {
    Y[i] <- 4
  } else if (X[i] >= 2900 & X[i] <= 2999) {
    Y[i] <- 4
  } else if (X[i] >= 2800 & X[i] <= 2829) {
    Y[i] <- 5
  } else if (X[i] >= 2840 & X[i] <= 2899) {
    Y[i] <- 5
  } else if (X[i] >= 3570 & X[i] <= 3579) {
    Y[i] <- 6
  } else if (X[i] >= 3660 & X[i] <= 3692) {
    Y[i] <- 6
  } else if (X[i] >= 3694 & X[i] <= 3699) {
    Y[i] <- 6
  } else if (X[i] >= 3810 & X[i] <= 3829) {
    Y[i] <- 6
  } else if (X[i] >= 7370 & X[i] <= 7379) {
    Y[i] <- 6
  } else if (X[i] >= 4800 & X[i] <= 4899) {
    Y[i] <- 7
  } else if (X[i] >= 4900 & X[i] <= 4949) {
    Y[i] <- 8
  } else if (X[i] >= 5000 & X[i] <= 5999) {
    Y[i] <- 9
  } else if (X[i] >= 7200 & X[i] <= 7299) {
    Y[i] <- 9
  } else if (X[i] >= 7600 & X[i] <= 7699) {
    Y[i] <- 9
  } else if (X[i] >= 2830 & X[i] <= 2839) {
    Y[i] <- 10
  } else if (X[i] >= 3693 & X[i] <= 3693) {
    Y[i] <- 10
  } else if (X[i] >= 3840 & X[i] <= 3859) {
    Y[i] <- 10
  } else if (X[i] >= 8000 & X[i] <= 8099) {
    Y[i] <- 10
  } else if (X[i] >= 6000 & X[i] <= 6999) {
    Y[i] <- 11
  } else {
    Y[i] <- 12
  }
IDs$FF_INDUSTRY_12 <<- Y
```

```
}

### RANK FUNDAMENTALS ###
ranked_fundamentals <- function(){

  output = matrix(nrow = length(unique(Peers$CIK_FYEAR)),
                  ncol = length(names(Funda)) - 2 + 1) %>% as_tibble()

  colnames(output) = c("CIK", "FYEAR", "PG_OVERLAP_RANK", paste0(colnames(Funda %>% select(-c("G
VKEY", "FYEAR", "LAG_ROA", "LAG_RET"))), "_RANK"))

  count = 0

  for(i in unique(Peers$CIK_FYEAR)){

    count = count + 1

    df = Peers %>%
      filter(CIK_FYEAR == i)

    output$CIK[count] = df$CIK[1]
    output$FYEAR[count] = df$FYEAR[1]
    output$PG_OVERLAP_RANK[count] = rank(c(df$PG_OVERLAP.x[1], df$PG_OVERLAP.y))[1] / max(rank(c
(df$PG_OVERLAP.x[1], df$PG_OVERLAP.y)))
    output$HP.COMPETITION_RANK[count] = rank(c(df$HP.COMPETITION.x[1], df$HP.COMPETITION.y))[1]
 / max(rank(c(df$HP.COMPETITION.x[1], df$HP.COMPETITION.y)))
    output$HP.SIMILARITY_RANK[count] = rank(c(df$HP.SIMILARITY.x[1], df$HP.SIMILARITY.y))[1] / m
ax(rank(c(df$HP.SIMILARITY.x[1], df$HP.SIMILARITY.y)))
    output$SALES_RANK[count] = rank(c(df$SALES.x[1], df$SALES.y))[1] / max(rank(c(df$SALES.x[1],
df$SALES.y)))
    output$MVE_RANK[count] = rank(c(df$MVE.x[1], df$MVE.y))[1] / max(rank(c(df$MVE.x[1], df$MVE.
y)))
    output$BTM_RANK[count] = rank(c(df$BTM.x[1], df$BTM.y))[1] / max(rank(c(df$BTM.x[1], df$BTM.
y)))
    output$LEVERAGE_RANK[count] = rank(c(df$LEVERAGE.x[1], df$LEVERAGE.y))[1] / max(rank(c(df$LE
VERAGE.x[1], df$LEVERAGE.y)))
    output$GROWTH_RANK[count] = rank(c(df$GROWTH.x[1], df$GROWTH.y))[1] / max(rank(c(df$GROWTH.x
[1], df$GROWTH.y)))
    output$PPE_RANK[count] = rank(c(df$PPE.x[1], df$PPE.y))[1] / max(rank(c(df$PPE.x[1], df$PPE.
y)))
    output$CASH_RANK[count] = rank(c(df$CASH.x[1], df$CASH.y))[1] / max(rank(c(df$CASH.x[1], df
$CASH.y)))
    output$SD_ROA_RANK[count] = rank(c(df$SD_ROA.x[1], df$SD_ROA.y))[1] / max(rank(c(df$SD_ROA.x
[1], df$SD_ROA.y)))
    output$ROA_RANK[count] = rank(c(df$SALES.x[1], df$SALES.y))[1] / max(rank(c(df$SALES.x[1], d
f$SALES.y)))
    output$SD_RET_RANK[count] = rank(c(df$SD_RET.x[1], df$SD_RET.y))[1] / max(rank(c(df$SD_RET.x
[1], df$SD_RET.y)))
    output$RET_RANK[count] = rank(c(df$RET.x[1], df$RET.y))[1] / max(rank(c(df$RET.x[1], df$RET.
y)))

    cat(i, "done \n")
```

```
  }

  return(output)

}

### SYNCHRONICITY FUNCTION ###
Peer_synchro <- function(){

  # PARAMETERS:
  options(warn = -1)

  output = Peers2 %>%
    select(GVKEY, FYEAR, FYEAR) %>%
    unique() %>%
    mutate(ACTUAL_IN = NA,
           ACTUAL_OUT = NA)

  in_sample.r2 = function(){

    ret = left_join(ret_firm %>%
                      filter(DATE >= id$DATE1[1] & DATE <= (id$DATE[1])),
                    ret_peers, by = "DATE")

    ret = ret %>%
      mutate_if(is.numeric, winsorize, 1, 99)

    # COMPUTE:
    f = ret[, 3]
    p = ret[, -c(1:3)] %>% data.frame
    p = if(ncol(p) > 1) rowMeans(p, na.rm = T) else as_vector(p)

    cor(f, p)^2 }

  out_sample.r2 = function(){

    ret = left_join(ret_firm %>%
                      filter(DATE >= id$DATE[1] & DATE <= (id$DATE2[1])),
                    ret_peers, by = "DATE")

    ret = ret %>%
      mutate_if(is.numeric, winsorize, 1, 99)

    # COMPUTE:
    f = ret[, 3]
    p = ret[, -c(1:3)] %>% data.frame
    p = if(ncol(p) > 1) rowMeans(p, na.rm = T) else as_vector(p)

    cor(f, p)^2 }

  # INPUT:
  for (i in 1:nrow(output)){
```

```
    start_time = Sys.time()

    # GET DATA:
    id = Peers2 %>%
      filter(GVKEY == output$GVKEY[i] & FYEAR == output$FYEAR[i])

    ret_firm = CRSP %>%
      select(GVKEY, DATE, FIRM_RET = RET) %>%
      filter(GVKEY == output$GVKEY[i])

    ret_peers = CRSP %>%
      filter(GVKEY %in% unique(id$PEER_GVKEY)) %>%
      select(PEER_GVKEY = GVKEY, DATE, PEER_RET = RET) %>%
      unique() %>%
      spread(PEER_GVKEY, PEER_RET)

    # FILL:
    optimized_value = try(in_sample.r2())

    if(is(optimized_value, "try-error")) {
      output$ACTUAL_IN[i] <- NA } else {
        output$ACTUAL_IN[i] <- optimized_value
      }

    optimized_value2 = try(out_sample.r2())

    if(is(optimized_value2, "try-error")) {
      output$ACTUAL_OUT[i] <- NA } else {
        output$ACTUAL_OUT[i] <- optimized_value2

      }

    # MESSAGE:
    ETA = Sys.time() + (Sys.time() - start_time) * (nrow(output) - i)

    cat("Done:", as.character(output$GVKEY[i]), paste0("FY", output$FYEAR[i]), paste0("- ", roun
d(i/nrow(output) * 100, 2), "% - ETA: ", ETA), "\n")

  }

  # return output #
  options(warn = 0)

  return(output)
}

##### IMPORT DATA: IDENTIFIERS #####
IDs <- left_join(x = dbSendQuery(wrds, "select gvkey, tic, datadate, fyear, cik, cusip
                   from compa.funda") %>% dbFetch,
                 y = dbSendQuery(wrds, "select gvkey, datadate, sich
                   from compa.co_industry") %>% dbFetch,
```

```
                  by = c("gvkey", "datadate")); n <- nrow(IDs)

IDs <- IDs[!duplicated(IDs), ] %>%
  filter(!is.na(fyear) & !is.na(gvkey)) %>%
  mutate(GVKEY_YEAR = paste(gvkey, fyear, sep = "_"))

IDs <- IDs %>%
  mutate(na_count = rowSums(is.na(IDs)),
         zero_count = rowSums(IDs == 0)) %>%
  arrange(na_count, zero_count) %>% distinct(GVKEY_YEAR, .keep_all = TRUE) %>%
  select(-c(GVKEY_YEAR, na_count, zero_count)); cat("Observations dropped:", n - nrow(IDs)); rm
(n)

IDs <- IDs %>%
  select(GVKEY = gvkey,
         CUSIP9 = cusip,
         CIK = cik,
         TICKER = tic,
         FYEAR = fyear,
         DATE = datadate,
         SIC4 = sich) %>%
  mutate(GVKEY = as.numeric(GVKEY),
         CIK = as.numeric(CIK),
         CUSIP8 = substr(CUSIP9, 1, 8),
         DATE = format(base::as.Date(DATE, "%Y%m%d"), format = "%Y-%m"),
         SIC2 = substr(SIC4, 1, 2)) %>%
  na.omit()

IDs <- IDs %>%
  filter(!is.na(SIC4))

sic_48(IDs$SIC4, IDs$FF48_INDUSTRY)
sic_12(IDs$SIC4, IDs$FF12_INDUSTRY)

# GICS DATA #
GICS <- dbSendQuery(wrds, "select gvkey, gind
                   from co_hgic") %>%
  dbFetch %>%
  transmute(GVKEY = as.numeric(gvkey),
            GICS6 = gind %>% as.numeric,
            GICS4 = substr(GICS6, 1, 4),
            GICS2 = substr(GICS6, 1, 2)) %>%
  unique() %>%
  distinct(GVKEY, .keep_all = TRUE)

##### IMPORT DATA: AGGRESSIVENESS - RAVENPACK #####
setwd("...") # set working directory

Ravenpack <- bind_rows(fread("RavenPack 2005.csv"),
                       fread("RavenPack 2006.csv"),
                       fread("RavenPack 2007.csv"),
                       fread("RavenPack 2008.csv"),
```

```
                      fread("RavenPack 2009.csv"),
                      fread("RavenPack 2010.csv"),
                      fread("RavenPack 2011.csv"),
                      fread("RavenPack 2012.csv"),
                      fread("RavenPack 2013.csv"),
                      fread("RavenPack 2014.csv"),
                      fread("RavenPack 2015.csv"),
                      fread("RavenPack 2016.csv"),
                      fread("RavenPack 2017.csv"),
                      fread("RavenPack 2018.csv"),
                      fread("RavenPack 2019.csv"))

Ravenpack2 <- Ravenpack %>%
  filter(ENS == 100) %>%
  transmute(RP_ENTITY_ID = RP_ENTITY_ID,
            ISIN = ISIN,
            DATE = as.Date(TIMESTAMP_UTC, "%Y-%m-%d"),
            GROUP = GROUP,
            TYPE = TYPE,
            ACTION = 0,
            NEW_PRODUCTS = as.numeric(GROUP == "products-services" & TYPE == "product-release"),
            PRICING = as.numeric(GROUP == "products-services" & TYPE == "product-price"),
            MARKETING = as.numeric(GROUP == "marketing" & TYPE == "campaign-ad" |
                                     GROUP == "marketing" & TYPE == "conference" |
                                     GROUP == "corporate-responsibility" & TYPE == "donation" |
                                     GROUP == "corporate-responsibility" & TYPE == "sponsorship"
),
            ACQUISITIONS = as.numeric(GROUP == "acquisitions-mergers" & TYPE == "acquisition" |
                                        GROUP == "acquisitions-mergers" & TYPE == "merger" |
                                        GROUP == "acquisitions-mergers" & TYPE == "stake" |
                                        GROUP == "acquisitions-mergers" & TYPE == "unit-acquisit
ion"),
            JOINT_VENTURE = as.numeric(GROUP == "partnerships" & TYPE == "joint-venture"),
            ALLIANCE = as.numeric(GROUP == "partnerships" & TYPE == "partnership"),
            MARKET_EXPANSION = as.numeric(GROUP == "products-services" & TYPE == "market-entry"
)) %>%
  mutate(ACTION = ifelse(NEW_PRODUCTS == 1, "NEW_PRODUCTS",
                         ifelse(PRICING == 1, "PRICING",
                                ifelse(MARKETING == 1, "MARKETING",
                                       ifelse(ACQUISITIONS == 1, "ACQUISITIONS",
                                              ifelse(JOINT_VENTURE == 1, "JOINT_VENTURE",
                                                     ifelse(ALLIANCE == 1, "ALLIANCE",
                                                            ifelse(MARKET_EXPANSION == 1, "MARKE
T_EXPANSION", "NO_ACTION")))))))) %>%
  filter(ACTION != "NO_ACTION") %>%
  na.omit() %>%
  unique() %>%
  na.omit() %>%
  filter(ISIN != "") %>%
  left_join(.,
            left_join(fread("Identifiers ISIN.csv") %>% select(ID = companyid,
                                                               ISIN = isin),
```

```
                      fread("Identifiers GVKEY.csv") %>% select(ID = companyid,
                                                                GVKEY = gvkey),
                      by = "ID") %>%
              select(-ID) %>%
              unique() %>%
              na.omit(),
            by = "ISIN") %>%
  na.omit() %>%
  select(GVKEY, DATE, ACTION, NEW_PRODUCTS, PRICING, MARKETING, ACQUISITIONS, JOINT_VENTURE, ALL
IANCE, MARKET_EXPANSION)

# write_csv(Ravenpack, "Ravenpack All (Cleaned).csv")

Ravenpack <- fread("Ravenpack All (Cleaned).csv") %>%
  mutate(YEAR = substr(DATE, 1, 4)) %>%
  group_by(GVKEY, YEAR, ACTION) %>%
  summarize(ACTIONS = n()) %>%
  group_by(GVKEY, YEAR) %>%
  mutate(ACTION_VOLUME = sum(ACTIONS, na.rm = T),
         ACTION_COMPLEXITY = (ACTIONS / sum(ACTIONS, na.rm = T))^2) %>%
  group_by(GVKEY, YEAR) %>%
  summarize(ACTION_VOLUME = mean(ACTION_VOLUME, na.rm = T),
            ACTION_COMPLEXITY = 1 - sum(ACTION_COMPLEXITY)) %>%
  unique()

Ravenpack2 <- fread("Ravenpack All (Cleaned).csv") %>%
  mutate(YEAR = substr(DATE, 1, 4)) %>%
  group_by(GVKEY, YEAR, ACTION) %>%
  summarize(ACTIONS = n()) %>%
  spread(., ACTION, ACTIONS)

colnames(Ravenpack2) <- c("GVKEY", "FYEAR", paste0("ACTION_", colnames(Ravenpack2)[3:9]))
Ravenpack2[is.na(Ravenpack2)] <- 0

##### IMPORT DATA: FUNDA #####
Funda <- left_join(x = dbSendQuery(wrds, "select gvkey, datadate, fyear, sale, ib, at, prcc_f, d
vpsp_f, csho, ceq, ajex, ppent, che, dltt, tlcf
                   from compa.funda") %>% dbFetch,
                   y = dbSendQuery(wrds, "select datadate, gvkey, sich
                   from compa.co_industry") %>% dbFetch,
                   by = c("gvkey", "datadate")); n <- nrow(Funda)

Funda <- Funda[!duplicated(Funda), ] %>%
  filter(!is.na(fyear) & !is.na(gvkey)) %>%
  mutate(GVKEY_YEAR = paste(gvkey, fyear, sep = "_"))

Funda <- Funda %>%
  mutate(na_count = rowSums(is.na(Funda)),
         zero_count = rowSums(Funda == 0)) %>%
  arrange(na_count, zero_count) %>% distinct(GVKEY_YEAR, .keep_all = TRUE) %>%
  select(-c(GVKEY_YEAR, na_count, zero_count)); cat("Observations dropped:", n - nrow(Funda)); r
m(n)
```

```
Funda <- Funda %>%
  transmute(GVKEY = as.numeric(gvkey),
            FYEAR = fyear,
            SIC4 = sich,
            SALES = sale,
            INCOME = ib,
            ASSETS = winsorize(at, 1, 99),
            PRICE = prcc_f,
            DIVIDEND = dvpsp_f,
            SHARES = csho,
            EQUITY = ceq,
            ADJUST = ajex,
            DEBT = dltt,
            PPE = ppent,
            CASH = che)

Lag_Funda <- Funda %>%
  mutate(FYEAR = FYEAR + 1,
         LAG_SALES = SALES,
         LAG_ASSETS = ASSETS,
         LAG_PRICE = PRICE,
         LAG_ADJUST = ADJUST) %>%
  select(GVKEY, FYEAR, LAG_SALES, LAG_ASSETS, LAG_PRICE, LAG_ADJUST)

Funda <- Funda %>% left_join(Lag_Funda, by = c("GVKEY", "FYEAR")); rm(Lag_Funda)

Funda <- Funda %>%
  mutate(GROWTH = (SALES - LAG_SALES) / LAG_SALES,

         ROA = INCOME / (ASSETS / 2 + LAG_ASSETS / 2),

         PRICE_ADJUST = PRICE / ADJUST,
         LAG_PRICE_ADJUST = LAG_PRICE / LAG_ADJUST,
         DIVIDEND_ADJUST = DIVIDEND / ADJUST,

         PPE = PPE / ASSETS,
         BTM = ASSETS / (PRICE * SHARES + ASSETS - EQUITY),
         MVE = PRICE * SHARES,
         LEVERAGE = DEBT / ASSETS,
         CASH = CASH / ASSETS,

         RET = (PRICE_ADJUST + DIVIDEND_ADJUST - LAG_PRICE_ADJUST) / LAG_PRICE_ADJUST)

Lag_Funda <- Funda %>%
  mutate(FYEAR = FYEAR + 1,
         LAG_ROA = ROA,
         LAG_RET = RET) %>%
  select(GVKEY, FYEAR, LAG_ROA, LAG_RET)

Funda <- Funda %>% left_join(Lag_Funda, by = c("GVKEY", "FYEAR")); rm(Lag_Funda)
```

```
Std_Perf <- Funda %>%
  select(GVKEY, FYEAR, RET, ROA) %>%
  arrange(GVKEY, FYEAR) %>%
  group_by(GVKEY) %>%
  do(data.frame(rollapplyr(.[-1], 5, by = 1, std.function, by.column = FALSE))) %>%
  ungroup %>%
  select(GVKEY, FYEAR, SD_RET, SD_ROA)

Funda <- Funda %>%
  left_join(Std_Perf,
            by = c("GVKEY", "FYEAR")); rm(Std_Perf)

Funda <- Funda %>%
  select(GVKEY, FYEAR, SALES, MVE, BTM, LEVERAGE, GROWTH, PPE, CASH, SD_ROA, ROA, LAG_ROA, SD_RE
T, RET, LAG_RET) %>%
  na.omit()

##### IMPORT DATA: HHI-INDEX #####
HHI <- Funda %>%
  select(GVKEY, FYEAR, SALES) %>%
  left_join(IDs %>%
              select(GVKEY, FYEAR, FF_INDUSTRY_48, FF_INDUSTRY_12),
            by = c("GVKEY", "FYEAR"))

HHI_FF48 <- HHI %>%
  left_join(HHI %>%
              group_by(FF_INDUSTRY_48, FYEAR) %>%
              summarize(INDUSTRY_SALES = sum(SALES, na.rm = T)),
            by = c("FF_INDUSTRY_48", "FYEAR")) %>%
  mutate(MARKET_SHARE = SALES / INDUSTRY_SALES * 100,
         MARKET_SHARE_SQUARED = MARKET_SHARE * MARKET_SHARE) %>%
  group_by(FF_INDUSTRY_48, FYEAR) %>%
  summarize(HHI_FF48 = sum(MARKET_SHARE_SQUARED, na.rm = T))

HHI_FF12 <- HHI %>%
  left_join(HHI %>%
              group_by(FF_INDUSTRY_12, FYEAR) %>%
              summarize(INDUSTRY_SALES = sum(SALES, na.rm = T)),
            by = c("FF_INDUSTRY_12", "FYEAR")) %>%
  mutate(MARKET_SHARE = SALES / INDUSTRY_SALES * 100,
         MARKET_SHARE_SQUARED = MARKET_SHARE * MARKET_SHARE) %>%
  group_by(FF_INDUSTRY_12, FYEAR) %>%
  summarize(HHI_FF12 = sum(MARKET_SHARE_SQUARED, na.rm = T)); rm(HHI)

##### IMPORT DATA: HOBERG AND PHILLIPS #####
setwd("...") # set working directory

Hoberg_Phillips <- read.delim("tnic3_data.txt") %>% # https://hobergphillips.tuck.dartmouth.edu/
industryclass.htm
  group_by(gvkey1, year) %>%
  summarize(HP.COMPETITION = n(),
            HP.SIMILARITY = median(score, na.rm = T)) %>%
```

```
  rename(GVKEY = gvkey1,
         FYEAR = year) %>%
  mutate(HP.COMPETITION = HP.COMPETITION %>% replace_na(., 0),
         HP.SIMILARITY = HP.SIMILARITY %>% replace_na(., 0)) %>%
  ungroup()

##### IMPORT DATA: RPE - RPE PLANS #####
Firms <- dbSendQuery(wrds, "select cik, ticker, companyname, cusip, fiscalyear
                    from iss_incentive_lab.companyFY") %>%
  dbFetch %>%
  filter(fiscalyear >= 2006)

Grants <- left_join(x = dbSendQuery(wrds, "select cik, fiscalyear, grantid, equitytarget, equity
threshold, equitymax, nonequitytarget, nonequitythreshold, nonequitymax
                    from iss_incentive_lab.gpbagrant") %>%
                       dbFetch %>%
                       unique(),
                     y = dbSendQuery(wrds, "select grantid, relid, periodid, metrictype, relative
benchmark, relativebenchmarkother
                    from iss_incentive_lab.gpbarel") %>%
                       dbFetch %>%
                       filter(!is.na(metrictype) & !is.na(relativebenchmark)) %>%
                       mutate(metrictype = fct_recode(as.factor(metrictype),
                                                      "Price" = "Stock Price",
                                                      "Price" = "Market-related",
                                                      "Accounting" = "Balance Sheet-related",
                                                      "Accounting" = "Earnings/Profit-related",
                                                      "Accounting" = "Financial/Investment return
rati",
                                                      "Accounting" = "Liquidity/Solvency-related"
,
                                                      "Accounting" = "Revenue-related",
                                                      "Other" = "Activity-related",
                                                      "Other" = "Cash Flow",
                                                      "Other" = "CSR",
                                                      "Other" = "Non-Financial",
                                                      "Other" = "Social",
                                                      "Other" = "NA"),
                              relativebenchmark = fct_recode(as.factor(relativebenchmark),
                                                             "Index" = "Other",
                                                             "Index" = "Other Index",
                                                             "SP500" = "S&P500",
                                                             "SP500" = "S&P 500",
                                                             "Peer" = "Peer Group")) %>%
                       distinct(paste0(grantid, metrictype, relativebenchmark), .keep_all = TRUE)
%>%
                       mutate(type = paste0(metrictype, relativebenchmark)) %>%
                       mutate(PRICE_PEER = as.numeric(type == "PricePeer"),
                              PRICE_INDEX = as.numeric(type == "PriceIndex"),
                              PRICE_SP500 = as.numeric(type == "PriceSP500"),
                              ACCOUNTING_PEER = as.numeric(type == "AccountingPeer"),
                              ACCOUNTING_INDEX = as.numeric(type == "AccountingIndex"),
```

```
                            ACCOUNTING_SP500 = as.numeric(type == "AccountingSP500"),
                            OTHER_PEER = as.numeric(type == "OtherPeer"),
                            OTHER_INDEX = as.numeric(type == "OtherIndex"),
                            OTHER_SP500 = as.numeric(type == "OtherSP500")) %>%
                        mutate(RPE = as.numeric(PRICE_PEER + ACCOUNTING_PEER + OTHER_PEER + PRICE_
INDEX + ACCOUNTING_INDEX + OTHER_INDEX + PRICE_SP500 + ACCOUNTING_SP500 + OTHER_SP500 >= 1),
                            RPE_SELF_SELECTED = as.numeric(PRICE_PEER + ACCOUNTING_PEER + OTHER
_PEER >= 1),
                            RPE_INDEX = as.numeric(PRICE_INDEX + ACCOUNTING_INDEX + OTHER_INDEX
+ PRICE_SP500 + ACCOUNTING_SP500 + OTHER_SP500 >= 1),
                            RPE_SP500 = as.numeric(PRICE_SP500 + ACCOUNTING_SP500 + OTHER_SP500
>= 1),
                            RPE_PRICE = as.numeric(PRICE_PEER + PRICE_INDEX + PRICE_SP500 >= 1
),
                            RPE_ACCOUNTING = as.numeric(ACCOUNTING_PEER + ACCOUNTING_INDEX + AC
COUNTING_SP500 >= 1)) %>%
                        group_by(grantid) %>%
                        summarize(RPE = max(RPE, na.rm = T),
                                  RPE_SELF_SELECTED = max(RPE_SELF_SELECTED, na.rm = T),
                                  RPE_INDEX = max(RPE_INDEX, na.rm = T),
                                  RPE_SP500 = max(RPE_SP500, na.rm = T),
                                  RPE_PRICE = max(RPE_PRICE, na.rm = T),
                                  RPE_ACCOUNTING = max(RPE_ACCOUNTING, na.rm = T)),
                      by = "grantid") %>%
  filter(fiscalyear >= 2006) %>%
  mutate(TARGET_UNITS = replace_na(equitytarget, 0), # equitymax, equitythreshold, equitytarget
         TARGET_DOLLAR = replace_na(nonequitytarget, 0)) %>% # nonequitymax, nonequitythreshold,
nonequitytarget
  filter(RPE == 1) %>%
  group_by(cik, fiscalyear) %>%
  summarize(RPE = max(replace_na(RPE, 0)),
            RPE_SELF_SELECTED = max(replace_na(RPE_SELF_SELECTED, 0)),
            RPE_INDEX = max(replace_na(RPE_INDEX, 0)),
            RPE_SP500 = max(replace_na(RPE_SP500, 0)),
            RPE_PRICE = max(replace_na(RPE_PRICE, 0)),
            RPE_ACCOUNTING = max(replace_na(RPE_ACCOUNTING, 0)),
            TARGET_UNITS = max(TARGET_UNITS, na.rm = T),
            TARGET_DOLLAR = max(TARGET_DOLLAR, na.rm = T)) %>%
  transmute(cik, fiscalyear,
            RPE, RPE_SELF_SELECTED, RPE_INDEX, RPE_SP500, RPE_PRICE, RPE_ACCOUNTING,
            RPE_EQUITY = as.numeric(TARGET_UNITS != 0),
            RPE_CASH = as.numeric(TARGET_DOLLAR != 0)) %>%
  na.omit()

RPE <- Firms %>%
  left_join(Grants,
            by = c("cik", "fiscalyear")) %>%
  transmute(CIK = cik %>% as.numeric(),
            FYEAR = fiscalyear %>% as.numeric(),
            RPE = RPE %>% replace_na(., 0),
            RPE_SELF_SELECTED = RPE_SELF_SELECTED %>% replace_na(., 0),
            RPE_INDEX = RPE_INDEX %>% replace_na(., 0),
```

```
            RPE_SP500 = RPE_SP500 %>% replace_na(., 0),
            RPE_PRICE = RPE_PRICE %>% replace_na(., 0),
            RPE_ACCOUNTING = RPE_ACCOUNTING %>% replace_na(., 0),
            RPE_EQUITY = RPE_EQUITY %>% replace_na(., 0),
            RPE_CASH = RPE_CASH %>% replace_na(., 0)) %>%
  na.omit(); rm(Firms, Grants)

##### IMPORT DATA: RPE - PEER SYNCHRONICITY #####
CRSP <- dbSendQuery(wrds, "select date, cusip, ret
                   from crsp.msf") %>%
  dbFetch %>%
  rename(DATE = date,
         CUSIP8 = cusip,
         RET = ret)

CCM <- dbSendQuery(wrds2, "select gvkey, tic, lpermno, cusip
                   from crsp_a_ccm.ccm_lookup") %>%
  dbFetch %>%
  transmute(GVKEY = gvkey %>% as.numeric,
            CUSIP8 = substr(cusip, 1, 8)) %>%
  na.omit() %>%
  unique()

CRSP <- CRSP %>%
  left_join(CCM,
            by = "CUSIP8") %>%
  filter(!is.na(GVKEY)) %>%
  filter(!is.na(RET)) %>%
  select(GVKEY, DATE, RET) %>%
  arrange(GVKEY, DATE)

Peers <- left_join(x = dbSendQuery(wrds, "select cik, fiscalyear, grantid
                   from iss_incentive_lab.gpbagrant") %>% dbFetch %>% unique(),
                   y = dbSendQuery(wrds, "select cik, grantid, peercik
                   from iss_incentive_lab.gpbarelpeer") %>% dbFetch,
                   by = c("cik", "grantid")) %>%
  na.omit() %>%
  transmute(CIK = as.numeric(cik),
            FYEAR = fiscalyear,
            PEER_CIK = as.numeric(peercik)) %>%
  unique %>%
  filter(CIK != PEER_CIK) %>%
  arrange(CIK, FYEAR) %>%
  left_join(IDs %>%
              select(GVKEY, FYEAR, DATE, CIK),
            by = c("CIK", "FYEAR")) %>%
  select(-CIK) %>%
  rename(CIK = PEER_CIK) %>%
  left_join(IDs %>%
              select(PEER_GVKEY = GVKEY, FYEAR, CIK),
            by = c("CIK", "FYEAR")) %>%
  select(GVKEY, FYEAR, PEER_GVKEY, DATE) %>%
```

```
  na.omit() %>%
  mutate(DATE = as.Date(paste0(DATE, "-01")) %m+% months(1), # beginning of next month
         DATE1 = DATE %m-% years(3),
         DATE2 = DATE %m+% years(3)) %>%
  na.omit() %>%
  filter(FYEAR >= 2006)

Synchronicity <- Peer_synchro(); rm(CRSP)

##### IMPORT DATA: RPE - PEER GROUPS #####
Peers_desc <- left_join(x = dbSendQuery(wrds, "select cik, fiscalyear, grantid
                    from iss_incentive_lab.gpbagrant") %>% dbFetch %>% unique(),
                         y = dbSendQuery(wrds, "select cik, grantid, peercik
                    from iss_incentive_lab.gpbarelpeer") %>% dbFetch,
                         by = c("cik", "grantid")) %>%
  na.omit() %>%
  transmute(CIK = as.numeric(cik),
            FYEAR = fiscalyear,
            PEER_CIK = as.numeric(peercik)) %>%
  unique %>%
  filter(CIK != PEER_CIK)

##### IMPORT DATA: RPE - PEER OVERLAP #####
Overlap <- left_join(x = dbSendQuery(wrds, "select cik, fiscalyear, grantid
                    from iss_incentive_lab.gpbagrant") %>% dbFetch %>% unique(),
                      y = dbSendQuery(wrds, "select cik, grantid, peercik
                    from iss_incentive_lab.gpbarelpeer") %>% dbFetch,
                      by = c("cik", "grantid")) %>%
  na.omit() %>%
  transmute(CIK = as.numeric(cik),
            FYEAR = fiscalyear,
            PEER_CIK = as.numeric(peercik)) %>%
  unique %>%
  filter(CIK != PEER_CIK)

row_match <- apply(Overlap, 1, function(dr) paste0(sort(dr), collapse = "_"))

Overlap$OVERLAP <- as.numeric(duplicated(row_match) | duplicated(row_match, fromLast = T)); rm(r
ow_match)

Lag_Overlap <- Overlap %>%
  mutate(FYEAR = FYEAR + 1,
         LAG_OVERLAP = OVERLAP) %>%
  select(CIK, FYEAR, PEER_CIK, LAG_OVERLAP)

Overlap <- Overlap %>% left_join(Lag_Overlap, by = c("CIK", "PEER_CIK", "FYEAR")); rm(Lag_Overla
p)

Overlap <- Overlap %>%
  mutate(NEW_EXOGENOUS_OVERLAP = as.numeric(OVERLAP == 1 & LAG_OVERLAP == 0),
         NEW_ENDOGENOUS_OVERLAP = as.numeric(OVERLAP == 1 & is.na(LAG_OVERLAP))) %>%
  group_by(CIK, FYEAR) %>%
```

```
  summarize(PG_OVERLAP = sum(OVERLAP, na.rm = T) / n(),
            N_NEW_EXOGENOUS_OVERLAP = sum(NEW_EXOGENOUS_OVERLAP, na.rm = T),
            N_NEW_ENDOGENOUS_OVERLAP = sum(NEW_ENDOGENOUS_OVERLAP, na.rm = T))

##### IMPORT DATA: RPE - MATCH BETWEEN FIRMS #####
setwd("...") # set working directory

Endogenous_match <- left_join(x = dbSendQuery(wrds, "select cik, fiscalyear, grantid
                   from iss_incentive_lab.gpbagrant") %>% dbFetch %>% unique(),
                              y = dbSendQuery(wrds, "select cik, grantid, peercik
                   from iss_incentive_lab.gpbarelpeer") %>% dbFetch,
                              by = c("cik", "grantid")) %>%
  na.omit() %>%
  transmute(CIK = as.numeric(cik),
            FYEAR = fiscalyear,
            PEER_CIK = as.numeric(peercik)) %>%
  unique %>%
  filter(CIK != PEER_CIK) %>%
  select(CIK, FYEAR, PEER_CIK) %>%
  unique()

row_match <- apply(Endogenous_match, 1, function(dr) paste0(sort(dr), collapse = "_"))

Endogenous_match$OVERLAP <- as.numeric(duplicated(row_match) | duplicated(row_match, fromLast =
T)); rm(row_match)

Lag_Endogenous_match <- Endogenous_match %>%
  mutate(FYEAR = FYEAR + 1,
         LAG_OVERLAP = OVERLAP) %>%
  select(CIK, FYEAR, PEER_CIK, LAG_OVERLAP)

Endogenous_match <- Endogenous_match %>% left_join(Lag_Endogenous_match, by = c("CIK", "PEER_CI
K", "FYEAR")); rm(Lag_Endogenous_match)

Endogenous_match <- Endogenous_match %>%
  filter(FYEAR > 2006) %>%
  mutate(NEW_EXOGENOUS_OVERLAP = as.numeric(OVERLAP == 1 & LAG_OVERLAP == 0),
         NEW_ENDOGENOUS_OVERLAP = as.numeric(OVERLAP == 1 & is.na(LAG_OVERLAP))) %>%
  mutate(NEW_EXOGENOUS_OVERLAP = replace_na(NEW_EXOGENOUS_OVERLAP, 0)) %>%
  select(-LAG_OVERLAP)

row_match <- apply(Endogenous_match %>%
                     select(CIK, FYEAR, PEER_CIK, NEW_ENDOGENOUS_OVERLAP), 1, function(dr) paste
0(sort(dr), collapse = "_"))

Endogenous_match$NEW_SIMULTANEOUS_OVERLAP <- as.numeric(duplicated(row_match) | duplicated(row_m
atch, fromLast = T)); rm(row_match)

Endogenous_match <- Endogenous_match %>%
  mutate(NEW_SIMULTANEOUS_OVERLAP = ifelse(NEW_ENDOGENOUS_OVERLAP == 0, 0, NEW_SIMULTANEOUS_OVER
LAP)) %>%
  mutate(NEW_ENDOGENOUS_OVERLAP = ifelse(NEW_SIMULTANEOUS_OVERLAP == 1, 0, NEW_ENDOGENOUS_OVERLA
```

```
P))

Endogenous_match <- Endogenous_match %>%
  mutate(FYEAR = FYEAR - 1) %>% # lead overlap variables
  left_join(IDs %>%
              select(CIK, GVKEY, FYEAR, FF_INDUSTRY_48),
            by = c("CIK", "FYEAR")) %>%
  left_join(GICS %>%
              select(GVKEY, GICS2, GICS4, GICS6),
            by = "GVKEY") %>%
  left_join(Funda %>%
              select(GVKEY, FYEAR, MVE, BTM, LEVERAGE, GROWTH, RET, SD_RET),
            by = c("GVKEY", "FYEAR")) %>%
  left_join(Hoberg_Phillips %>%
              select(GVKEY, FYEAR, HP.COMPETITION, HP.SIMILARITY),
            by = c("GVKEY", "FYEAR")) %>%
  left_join(Endogenous_match %>%
              group_by(CIK, FYEAR) %>%
              summarize(LAG_OVERLAP = sum(OVERLAP) / n()) %>%
              ungroup(),
            by = c("CIK", "FYEAR")) %>%
  rename(FIRM_CIK = CIK,
         FIRM_GVKEY = GVKEY,
         CIK = PEER_CIK) %>%
  left_join(IDs %>%
              select(CIK, GVKEY, FYEAR, FF_INDUSTRY_48) %>%
              rename(FF_INDUSTRY_48.y = FF_INDUSTRY_48),
            by = c("CIK", "FYEAR")) %>%
  left_join(GICS %>%
              select(GVKEY, GICS2, GICS4, GICS6) %>%
              rename(GICS2.y = GICS2,
                     GICS4.y = GICS4,
                     GICS6.y = GICS6),
            by = "GVKEY") %>%
  left_join(Funda %>%
              select(GVKEY, FYEAR, MVE, BTM, LEVERAGE, GROWTH, RET, SD_RET) %>%
              rename(MVE.y = MVE, BTM.y = BTM, LEVERAGE.y = LEVERAGE, GROWTH.y = GROWTH, RET.y =
RET, SD_RET.y = SD_RET),
            by = c("GVKEY", "FYEAR")) %>%
  left_join(Hoberg_Phillips %>%
              select(GVKEY, FYEAR, HP.COMPETITION, HP.SIMILARITY) %>%
              rename(HP.COMPETITION.y = HP.COMPETITION, HP.SIMILARITY.y = HP.SIMILARITY),
            by = c("GVKEY", "FYEAR")) %>%
  left_join(Endogenous_match %>%
              group_by(CIK, FYEAR) %>%
              summarize(LAG_OVERLAP.y = sum(OVERLAP) / n()) %>%
              ungroup(),
            by = c("CIK", "FYEAR")) %>%
  rename(PEER_GVKEY = GVKEY) %>%
  mutate(LAG_OVERLAP = replace_na(LAG_OVERLAP, 0),
         LAG_OVERLAP.y = replace_na(LAG_OVERLAP.y, 0)) %>%
  mutate(FIRM_YEAR_PEER = paste0(FIRM_GVKEY, FYEAR, PEER_GVKEY)) %>%
```

```
  left_join(read_csv("tnic3_data.csv") %>%
              select(FIRM_GVKEY = gvkey1,
                     FYEAR = year,
                     PEER_GVKEY = gvkey2) %>%
              transmute(FIRM_YEAR_PEER = paste0(FIRM_GVKEY, FYEAR, PEER_GVKEY),
                        HP_MATCH = 1),
            by = "FIRM_YEAR_PEER") %>%
  mutate(HP_MATCH = replace_na(HP_MATCH, 0)) %>%
  mutate(MVE = winsorize(MVE, 1, 99),
         MVE.y = winsorize(MVE.y, 1, 99),
         BTM = winsorize(BTM, 1, 99),
         BTM.y = winsorize(BTM.y, 1, 99),
         LEVERAGE = winsorize(LEVERAGE, 1, 99),
         LEVERAGE.y = winsorize(LEVERAGE.y, 1, 99),
         GROWTH = winsorize(GROWTH, 1, 99),
         GROWTH.y = winsorize(GROWTH.y, 1, 99),
         RET = winsorize(RET, 1, 99),
         RET.y = winsorize(RET.y, 1, 99),
         SD_RET = winsorize(SD_RET, 1, 99),
         SD_RET.y = winsorize(SD_RET.y, 1, 99),
         HP.COMPETITION = winsorize(HP.COMPETITION, 1, 99),
         HP.COMPETITION.y = winsorize(HP.COMPETITION.y, 1, 99),
         HP.SIMILARITY = winsorize(HP.SIMILARITY, 1, 99),
         HP.SIMILARITY.y = winsorize(HP.SIMILARITY.y, 1, 99)) %>%
  transmute(GVKEY = FIRM_GVKEY, PEER_GVKEY, FYEAR, OVERLAP, NEW_EXOGENOUS_OVERLAP, NEW_ENDOGENOU
S_OVERLAP, NEW_SIMULTANEOUS_OVERLAP, LAG_OVERLAP,
            FF_INDUSTRY_48, MVE, BTM, LEVERAGE, GROWTH, RET, SD_RET,
            DIFF_FF_48 = as.numeric(FF_INDUSTRY_48 != FF_INDUSTRY_48.y),
            DIFF_GICS6 = as.numeric(GICS6 != GICS6.y),
            DIFF_HP = as.numeric(HP_MATCH == 0),

            DIFF_MVE = abs(log1p(MVE) - log1p(MVE.y)),
            DIFF_BTM = abs(BTM - BTM.y),
            DIFF_LEVERAGE = abs(LEVERAGE - LEVERAGE.y),
            DIFF_GROWTH = abs(GROWTH - GROWTH.y),
            DIFF_RET = abs(RET - RET.y),
            DIFF_SD_RET = abs(SD_RET - SD_RET.y),
            DIFF_HP_COMP = abs(log1p(HP.COMPETITION) - log1p(HP.COMPETITION.y)),
            DIFF_HP_SIMILARITY = abs(HP.SIMILARITY - HP.SIMILARITY.y),
            DIFF_LAG_OVERLAP = abs(LAG_OVERLAP - LAG_OVERLAP.y)) %>%
  na.omit()

Endogenous_match <- Endogenous_match %>%
  group_by(GVKEY, PEER_GVKEY, FYEAR) %>%
  slice(1) %>%  # drop duplicates
  ungroup() %>%
  mutate(DIFF_MVE = DIFF_MVE %>% winsorize(., 1, 99),
         DIFF_BTM = DIFF_BTM %>% winsorize(., 1, 99),
         DIFF_LEVERAGE = DIFF_LEVERAGE %>% winsorize(., 1, 99),
         DIFF_GROWTH = DIFF_GROWTH %>% winsorize(., 1, 99),
         DIFF_RET = DIFF_RET %>% winsorize(., 1, 99),
         DIFF_SD_RET = DIFF_SD_RET %>% winsorize(., 1, 99),
```

```
         DIFF_HP_COMP = DIFF_HP_COMP %>% winsorize(., 1, 99),
         DIFF_HP_SIMILARITY = DIFF_HP_SIMILARITY %>% winsorize(., 1, 99))

##### IMPORT DATA: RPE - DURATION FIRM-PEER RELATION #####
RPE_duration <- left_join(x = dbSendQuery(wrds, "select cik, fiscalyear, grantid
                  from iss_incentive_lab.gpbagrant") %>% dbFetch %>% unique(),
                         y = dbSendQuery(wrds, "select cik, grantid, peercik
                  from iss_incentive_lab.gpbarelpeer") %>% dbFetch,
                         by = c("cik", "grantid")) %>%
  na.omit() %>%
  transmute(CIK = as.numeric(cik),
            FYEAR = fiscalyear,
            PEER_CIK = as.numeric(peercik)) %>%
  unique %>%
  filter(CIK != PEER_CIK)

RPE_duration <- RPE_duration %>%
  left_join(RPE_duration %>%
              select(CIK, FYEAR) %>%
              unique() %>%
              group_by(CIK) %>%
              summarize(N_YEARS_RPE = n()),
            by = "CIK") %>%
  group_by(CIK, PEER_CIK, N_YEARS_RPE) %>%
  summarize(N_YEARS_PEER = n()) %>%
  mutate(PERC_TIME_RPE_PEER = N_YEARS_PEER / N_YEARS_RPE) %>%
  group_by(CIK) %>%
  summarize(PERC_TIME_RPE_PEER = mean(PERC_TIME_RPE_PEER),
            PERC_TIME_RPE_PEER = median(PERC_TIME_RPE_PEER))

##### IMPORT DATA: RPE - GRANT SIZE SIMILARITY #####
### PRICE CURRENT DATA ###
CRSP <- dbSendQuery(wrds, "select date, permco, prc
                  from crsp.msf") %>%
  dbFetch %>%
  mutate(PERMCO = permco,
         DATE = date,
         PRICE = as.numeric(prc)) %>%
  arrange(PERMCO, DATE)

CCM <- dbSendQuery(wrds, "select lpermco, cusip, tic, cik
                  from ccm_lookup") %>% dbFetch %>%
  na.omit() %>%
  unique() %>%
  transmute(CUSIP8 = substr(cusip, 1, 8),
            CIK = as.numeric(cik),
            TICKER = tic,
            PERMCO = lpermco) %>%
  distinct(PERMCO, .keep_all = TRUE)

CRSP <- CRSP %>%
  left_join(CCM,
```

```
            by = "PERMCO") %>%
  select(CIK, DATE, PRICE) %>%
  na.omit() %>%
  filter(PRICE > 0) %>%
  mutate(DATE = format(base::as.Date(DATE, "%Y%m%d"), format = "%Y-%m")) %>%
  filter(!is.na(DATE) & !is.na(CIK)) %>%
  mutate(CIK_DATE = paste(CIK, DATE, sep = "_")) %>%
  distinct(CIK_DATE, .keep_all = TRUE) %>%
  select(-CIK_DATE); rm(CCM)

### CURRENT COMPENSATION ###
Execucomp <- dbSendQuery(wrds, "select gvkey, co_per_rol, year, ceoann, becameceo, salary, bonu
s, noneq_incent, age, tdc1
                   from anncomp") %>% dbFetch; n <- nrow(Execucomp)

Execucomp <- Execucomp[!duplicated(Execucomp), ] %>%
  filter(!is.na(year) & !is.na(gvkey) & !is.na(co_per_rol)) %>%
  mutate(GVKEY_YEAR_ROL = paste(gvkey, year, co_per_rol, sep = "_"))

Execucomp <- Execucomp %>%
  mutate(na_count = rowSums(is.na(Execucomp)),
         zero_count = rowSums(Execucomp == 0)) %>%
  arrange(na_count, zero_count) %>% distinct(GVKEY_YEAR_ROL, .keep_all = TRUE) %>%
  select(-c(GVKEY_YEAR_ROL, na_count, zero_count)); cat("Observations dropped:", n - nrow(Execuc
omp)); rm(n)

Execucomp <- Execucomp %>%
  filter(ceoann == "CEO")
transmute(GVKEY = as.numeric(gvkey),
          FYEAR = as.numeric(year),
          CO_PER_ROL = co_per_rol,
          CEO = ceoann,
          AGE = age,
          TOTAL_COMP = replace_na(tdc1, 0),
          CASH_COMP = rowSums(. [, c("salary", "bonus", "noneq_incent")], na.rm = T))

### RPE GRANT SIZE ###
# ALL PLANS #
RPE_Targets <- left_join(x = dbSendQuery(wrds, "select grantid, relid, periodid, metrictype, rel
ativebenchmark, relativebenchmarkother
                   from iss_incentive_lab.gpbarel") %>%
                           dbFetch,
                         y = dbSendQuery(wrds, "select cik, fiscalyear, grantid, equitytarget, e
quitythreshold, equitymax, nonequitytarget, nonequitythreshold, nonequitymax
                   from iss_incentive_lab.gpbagrant") %>% dbFetch,
                         by = "grantid") %>%
  filter(!is.na(relativebenchmark)) %>%
  # filter(!is.na(equitytarget)) %>%
  # filter(!is.na(nonequitytarget)) %>%
  filter(fiscalyear >= 2006) %>%
  transmute(CIK = as.numeric(cik),
            FYEAR = as.numeric(fiscalyear),
```

```
            TARGET_UNITS = replace_na(equitymax, 0), # equitymax, equitythreshold, equitytarget
            TARGET_DOLLAR = replace_na(nonequitymax, 0)) %>% # nonequitymax, nonequitythreshold,
nonequitytarget
  group_by(CIK, FYEAR) %>%
  summarize(TARGET_UNITS = max(TARGET_UNITS, na.rm = T),
            TARGET_DOLLAR = max(TARGET_DOLLAR, na.rm = T)) %>%
  filter(TARGET_UNITS > 0 | TARGET_DOLLAR > 0) %>%
  mutate(TARGET_DOLLAR = ifelse(TARGET_UNITS > 0, 0, TARGET_DOLLAR)) %>%
  left_join(IDs,
            by = c("CIK", "FYEAR")) %>%
  left_join(CRSP,
            by = c("CIK", "DATE")) %>%
  mutate(TARGET_UNITS_USD = TARGET_UNITS * PRICE) %>%
  mutate(RPE_TARGET_USD = ifelse(TARGET_UNITS_USD > 0, TARGET_UNITS_USD, TARGET_DOLLAR)) %>%
  select(CIK, FYEAR, RPE_TARGET_USD) %>%
  na.omit()

setwd("...") # set working directory

RPE_Incentives <- left_join(x = dbSendQuery(wrds, "select cik, fiscalyear, grantid
                   from iss_incentive_lab.gpbagrant") %>% dbFetch %>% unique(),
                   y = dbSendQuery(wrds, "select cik, grantid, peercik
                   from iss_incentive_lab.gpbarelpeer") %>% dbFetch,
                   by = c("cik", "grantid")) %>%
  na.omit() %>%
  transmute(CIK = as.numeric(cik),
            FYEAR = fiscalyear,
            PEER_CIK = as.numeric(peercik)) %>%
  unique %>%
  filter(CIK != PEER_CIK)

row_match <- apply(RPE_Incentives, 1, function(dr) paste0(sort(dr), collapse = "_"))

RPE_Incentives$OVERLAP <- as.numeric(duplicated(row_match) | duplicated(row_match, fromLast = T
)); rm(row_match)

RPE_Incentives <- RPE_Incentives %>%
  filter(OVERLAP == 1) %>%
  left_join(RPE_Targets,
            by = c("CIK", "FYEAR")) %>%
  left_join(IDs %>%
              select(CIK, FYEAR, GVKEY),
            by = c("CIK", "FYEAR")) %>%
  left_join(Execucomp %>%
              mutate(FYEAR = FYEAR + 1) %>%
              select(GVKEY, FYEAR, LAG_TOTAL_COMP = TOTAL_COMP, LAG_CASH_COMP = CASH_COMP),
            by = c("GVKEY", "FYEAR")) %>%
  mutate(RPE_TARGET_USD = RPE_TARGET_USD / (LAG_TOTAL_COMP * 1000)) %>%
  select(-c(GVKEY, LAG_TOTAL_COMP, LAG_CASH_COMP)) %>%
  rename(FIRM_CIK = CIK,
         CIK = PEER_CIK) %>%
  left_join(RPE_Targets %>%
```

```
              rename(PEER_RPE_TARGET_USD = RPE_TARGET_USD),
            by = c("CIK", "FYEAR")) %>%
  left_join(IDs %>%
              select(CIK, FYEAR, GVKEY),
            by = c("CIK", "FYEAR")) %>%
  left_join(Execucomp %>%
              mutate(FYEAR = FYEAR + 1) %>%
              select(GVKEY, FYEAR, LAG_TOTAL_COMP = TOTAL_COMP, LAG_CASH_COMP = CASH_COMP),
            by = c("GVKEY", "FYEAR")) %>%
  mutate(PEER_RPE_TARGET_USD = (PEER_RPE_TARGET_USD) / (LAG_TOTAL_COMP * 1000)) %>%
  select(-c(GVKEY, LAG_TOTAL_COMP, LAG_CASH_COMP)) %>%
  rename(PEER_CIK = CIK,
         CIK = FIRM_CIK) %>%
  na.omit() %>%
  group_by(CIK, FYEAR) %>%
  summarize(ABN_RPE_TARGET_USD = sqrt(sum((RPE_TARGET_USD - PEER_RPE_TARGET_USD)^2))) # euclidea
n

RPE_Targets <- RPE_Targets %>%
  left_join(IDs %>%
              select(GVKEY, CIK, FYEAR),
            by = c("CIK", "FYEAR")) %>%
  left_join(Execucomp %>%
              mutate(FYEAR = FYEAR + 1) %>%
              select(GVKEY, FYEAR, LAG_TOTAL_COMP = TOTAL_COMP, LAG_CASH_COMP = CASH_COMP),
            by = c("GVKEY", "FYEAR")) %>%
  mutate(RPE_TARGET_USD = (RPE_TARGET_USD) / (LAG_TOTAL_COMP * 1000))

##### IMPORT DATA: RPE - CRITERIA SIMILARITY #####
RPE_criteria <- left_join(x = dbSendQuery(wrds, "select cik, fiscalyear, grantid
                    from iss_incentive_lab.gpbagrant") %>% dbFetch %>% unique(),
                    y = dbSendQuery(wrds, "select cik, grantid, peercik
                    from iss_incentive_lab.gpbarelpeer") %>% dbFetch,
                    by = c("cik", "grantid")) %>%
  na.omit() %>%
  transmute(CIK = as.numeric(cik),
            FYEAR = fiscalyear,
            PEER_CIK = as.numeric(peercik)) %>%
  unique %>%
  filter(CIK != PEER_CIK)

row_match <- apply(RPE_criteria, 1, function(dr) paste0(sort(dr), collapse = "_"))

RPE_criteria$OVERLAP <- as.numeric(duplicated(row_match) | duplicated(row_match, fromLast = T));
rm(row_match)

RPE_criteria <- RPE_criteria %>%
  filter(OVERLAP == 1) %>%
  left_join(RPE %>%
              filter(RPE_SELF_SELECTED == 1) %>%
              select(CIK, FYEAR, RPE_PRICE, RPE_ACCOUNTING),
            by = c("CIK", "FYEAR")) %>%
```

```
  rename(FIRM_CIK = CIK,
         CIK = PEER_CIK) %>%
  left_join(RPE %>%
              filter(RPE_SELF_SELECTED == 1) %>%
              select(CIK, FYEAR, RPE_PRICE, RPE_ACCOUNTING) %>%
              rename(PEER_RPE_PRICE = RPE_PRICE,
                     PEER_RPE_ACCOUNTING = RPE_ACCOUNTING),
            by = c("CIK", "FYEAR")) %>%
  rename(PEER_CIK = CIK,
         CIK = FIRM_CIK) %>%
  na.omit() %>%
  group_by(CIK, FYEAR) %>%
  summarize(ABN_RPE_PRICE = sqrt(sum((RPE_PRICE - PEER_RPE_PRICE)^2)))

##### IMPORT DATA: FUNDA - PEER GROUP RANKED #####
Funda.ranked <- left_join(x = dbSendQuery(wrds, "select cik, fiscalyear, grantid
                    from iss_incentive_lab.gpbagrant") %>% dbFetch %>% unique(),
                          y = dbSendQuery(wrds, "select cik, grantid, peercik
                    from iss_incentive_lab.gpbarelpeer") %>% dbFetch,
                          by = c("cik", "grantid")) %>%
  na.omit() %>%
  transmute(CIK = as.numeric(cik),
            FYEAR = fiscalyear,
            PEER_CIK = as.numeric(peercik)) %>%
  unique %>%
  filter(CIK != PEER_CIK)

row_match <- apply(Funda.ranked, 1, function(dr) paste0(sort(dr), collapse = "_"))

Funda.ranked$OVERLAP <- as.numeric(duplicated(row_match) | duplicated(row_match, fromLast = T));
rm(row_match)

Overlap <- Funda.ranked %>%
  group_by(CIK, FYEAR) %>%
  summarize(PG_OVERLAP = sum(OVERLAP) / n())

Funda.ranked <- Funda.ranked %>%
  select(-OVERLAP) %>%
  left_join(Funda %>%
              left_join(IDs %>% select(GVKEY, CIK, FYEAR),
                        by = c("GVKEY", "FYEAR")),
            by = c("CIK", "FYEAR")) %>%
  left_join(Hoberg_Phillips,
            by = c("GVKEY", "FYEAR")) %>%
  left_join(Overlap,
            by = c("CIK", "FYEAR")) %>%
  select(-GVKEY) %>%
  rename(FIRM_CIK = CIK,
         CIK = PEER_CIK) %>%
  left_join(Funda %>%
              left_join(IDs %>% select(GVKEY, CIK, FYEAR),
                        by = c("GVKEY", "FYEAR")),
```

```
            by = c("CIK", "FYEAR")) %>%
  left_join(Hoberg_Phillips,
            by = c("GVKEY", "FYEAR")) %>%
  left_join(Overlap,
            by = c("CIK", "FYEAR")) %>%
  select(-GVKEY) %>%
  rename(PEER_CIK = CIK,
         CIK = FIRM_CIK) %>%
  mutate(CIK_FYEAR = paste0(CIK, "_", FYEAR))

Funda.ranked <- ranked_fundamentals()

##### IMPORT DATA: PEERS/GICS MATCH #####
setwd("...") # set working directory

Overlap_match <- Peers_desc %>%
  select(CIK, FYEAR, PEER_CIK) %>%
  filter(CIK != PEER_CIK) %>%
  left_join(IDs %>%
              select(CIK, GVKEY, FYEAR, SIC4, SIC2, FF_INDUSTRY_48) %>%
              mutate(SIC1.x = substr(SIC4, 1, 1),
                     SIC3.x = substr(SIC4, 1, 3)) %>%
              mutate(SIC2.x = SIC2,
                     SIC4.x = SIC4,
                     FF_INDUSTRY_48.X = FF_INDUSTRY_48),
            by = c("CIK", "FYEAR")) %>%
  left_join(GICS %>%
              select(GVKEY, GICS2, GICS4, GICS6) %>%
              mutate(GICS2.x = GICS2,
                     GICS4.x = GICS4,
                     GICS6.x = GICS6),
            by = "GVKEY") %>%
  rename(FIRM_CIK = CIK,
         FIRM_GVKEY = GVKEY,
         CIK = PEER_CIK) %>%
  left_join(IDs %>%
              select(CIK, GVKEY, FYEAR, SIC4, SIC2, FF_INDUSTRY_48) %>%
              mutate(SIC1.y = substr(SIC4, 1, 1),
                     SIC3.y = substr(SIC4, 1, 3)) %>%
              mutate(SIC2.y = SIC2,
                     SIC4.y = SIC4,
                     FF_INDUSTRY_48.y = FF_INDUSTRY_48),
            by = c("CIK", "FYEAR")) %>%
  left_join(GICS %>%
              select(GVKEY, GICS2, GICS4, GICS6) %>%
              mutate(GICS2.y = GICS2,
                     GICS4.y = GICS4,
                     GICS6.y = GICS6),
            by = "GVKEY") %>%
  rename(PEER_GVKEY = GVKEY) %>%
  select(FIRM_GVKEY, FYEAR, PEER_GVKEY, SIC1.x, SIC1.y, SIC2.x, SIC2.y, SIC3.x, SIC3.y, SIC4.x,
 SIC4.y, GICS2.x, GICS4.x, GICS6.x, GICS2.y, GICS4.y, GICS6.y, FF_INDUSTRY_48.x, FF_INDUSTRY_48.
```

```
y) %>%
  left_join(read_csv("tnic3_data.csv") %>%
              select(FIRM_GVKEY = gvkey1,
                     FYEAR = year,
                     PEER_GVKEY = gvkey2,
                     SCORE = score) %>%
              mutate(IN_HP = 1),
            by = c("FIRM_GVKEY", "FYEAR", "PEER_GVKEY"))

row_match <- apply(Overlap_match, 1, function(dr) paste0(sort(dr), collapse = "_"))

Overlap_match$OVERLAP <- as.numeric(duplicated(row_match) | duplicated(row_match, fromLast = T
)); rm(row_match)

Overlap_match <- Overlap_match %>%
  filter(!is.na(FIRM_GVKEY)) %>%
  mutate(IN_HP = replace_na(IN_HP, 0)) %>%
  mutate(
    # SAME_SIC = as.numeric(SIC1.x == SIC1.y),
    SAME_SIC = as.numeric(SIC2.x == SIC2.y),
    # SAME_SIC = as.numeric(SIC3.x == SIC3.y),
    # SAME_SIC = as.numeric(SIC4.x == SIC4.y),
    # SAME_GICS = as.numeric(GICS2.x == GICS2.y),
    # SAME_GICS = as.numeric(GICS4.x == GICS4.y),
    SAME_GICS = as.numeric(GICS6.x == GICS6.y),
    SAME_FF = as.numeric(FF_INDUSTRY_48.x == FF_INDUSTRY_48.y),
  ) %>%
  mutate(OVERLAP_IN_HP = as.numeric(OVERLAP == 1 & IN_HP == 1),
         OVERLAP_NOT_IN_HP = as.numeric(OVERLAP == 1 & IN_HP == 0),
         OVERLAP_IN_SIC = as.numeric(OVERLAP == 1 & SAME_SIC == 1),
         OVERLAP_NOT_IN_SIC = as.numeric(OVERLAP == 1 & SAME_SIC == 0),
         OVERLAP_IN_GICS = as.numeric(OVERLAP == 1 & SAME_GICS == 1),
         OVERLAP_NOT_IN_GICS = as.numeric(OVERLAP == 1 & SAME_GICS == 0),
         OVERLAP_IN_BOTH = as.numeric(OVERLAP == 1 & SAME_GICS == 1 & IN_HP == 1),
         OVERLAP_IN_GICS_NOT_IN_HP = as.numeric(OVERLAP == 1 & SAME_GICS == 1 & IN_HP == 0),
         OVERLAP_NOT_IN_GICS_NOT_IN_HP = as.numeric(OVERLAP == 1 & SAME_GICS == 0 & IN_HP == 0),
         OVERLAP_IN_FF = as.numeric(OVERLAP == 1 & SAME_FF == 1),
         OVERLAP_NOT_IN_FF = as.numeric(OVERLAP == 1 & SAME_FF == 0)) %>%
  group_by(FIRM_GVKEY, FYEAR) %>%
  summarize(PG_IN_HP = sum(IN_HP, na.rm = T) / n(),
            PG_IN_SIC = sum(SAME_SIC, na.rm = T) / n(),
            PG_IN_GICS = sum(SAME_GICS, na.rm = T) / n(),
            OVERLAP_IN_HP = sum(OVERLAP_IN_HP, na.rm = T) / n(),
            OVERLAP_NOT_IN_HP = sum(OVERLAP_NOT_IN_HP, na.rm = T) / n(),
            OVERLAP_IN_SIC = sum(OVERLAP_IN_SIC, na.rm = T) / n(),
            OVERLAP_NOT_IN_SIC = sum(OVERLAP_NOT_IN_SIC, na.rm = T) / n(),
            OVERLAP_IN_GICS = sum(OVERLAP_IN_GICS, na.rm = T) / n(),
            OVERLAP_NOT_IN_GICS = sum(OVERLAP_NOT_IN_GICS, na.rm = T) / n(),
            OVERLAP_IN_BOTH = sum(OVERLAP_IN_BOTH, na.rm = T) / n(),
            OVERLAP_IN_GICS_NOT_IN_HP = sum(OVERLAP_IN_GICS_NOT_IN_HP, na.rm = T) / n(),
            OVERLAP_NOT_IN_GICS_NOT_IN_HP = sum(OVERLAP_NOT_IN_GICS_NOT_IN_HP, na.rm = T) / n(),
            OVERLAP_IN_FF = sum(OVERLAP_IN_FF, na.rm = T) / n(),
```

```
            OVERLAP_NOT_IN_FF = sum(OVERLAP_NOT_IN_FF, na.rm = T) / n()) %>%
  ungroup() %>%
  rename(GVKEY = FIRM_GVKEY) # how many (overlapping) peers from peer group are in same H&P/SIC/
GICS?
```