SleepWell Trial Codebook

Joshua F. Wiley (joshua.wiley@monash.edu)

04 Mar 2021

Contents

1 Background

This document employs literate programming to include notes and documentation along with the actual source code used for data management.

2 Load R Packages

A number of R packages are used for data management and analyses. Package versions are controlled using the checkpoint package.

```
## running and echo package and function setup code
source('package_function_setup.R',
       echo = TRUE, keep.source = TRUE)

## source("~/OneDrive/RCode/promis_score.R",
##        echo = TRUE, keep.source = TRUE)

Sys.setenv(TZ = "Australia/Melbourne")

loc.data <- "g:/My Drive/CBTI+Light project/Recruitment and data collection/"
```

2.1 RedCap Randomization Data

```
if (FALSE) {
rcon <- redcapConnection(
  url="https://redcap.cdms.org.au/api/",
  token=key_get("redcapapi", "sleepwell"))

rData <- exportRecords(rcon, survey = TRUE)

saveRDS(rData,
        file = file.path(loc.data, "redcapdata.RDS"),
        compress = "xz")
} else {
  rData <- readRDS(file.path(loc.data, "redcapdata.RDS"))
}

rData$baseline_isi_high <- factor(rData$baseline_isi_high,
                                  levels = c(TRUE, FALSE),
```

```
                                  labels = c("ISI >= 8", "ISI <= 7"))
rData$baseline_stage_high <- factor(rData$baseline_stage_high,
                                  levels = c(TRUE, FALSE),
                                  labels = c("Stage >= 3", "Stage <= 2"))
```

2.2 Qualtrics Baseline Data

```
if (FALSE) {
## read screening data in from Qualtrics
d.t0.info <- fread(file.path(loc.data, "Data/SleepWell_Screen.csv"),
                         nrows = 3, header = FALSE)
d.t0 <- fread(file = file.path(loc.data,
  "Data/SleepWell_Screen.csv"), skip = "anonymous", header=FALSE,
  logical01 = FALSE)
setnames(d.t0, make.names(unlist(d.t0.info[1,])))
d.t0[, RecipientFirstName :=as.integer( gsub("SW", "", IDDate_2, ignore.case=TRUE))]

d.t1.info <- fread(file.path(loc.data, "Data/SleepWell_T1.csv"),
                         nrows = 3, header = FALSE)
d.t1 <- fread(file = file.path(loc.data,
  "Data/SleepWell_T1.csv"), skip = "John_Doe@email.com", header=FALSE,
  logical01 = FALSE)
setnames(d.t1, make.names(unlist(d.t1.info[1,])))
## should be empty table
d.t1[is.na(as.integer(RecipientFirstName)), .(RecipientFirstName, ID)]

d.t2.info <- fread(file.path(loc.data, "Data/SleepWell_T2.csv"),
                         nrows = 3, header = FALSE)
d.t2 <- fread(file = file.path(loc.data,
  "Data/SleepWell_T2.csv"), skip = "John_Doe@email.com", header=FALSE,
  logical01 = FALSE)
setnames(d.t2, make.names(unlist(d.t2.info[1,])))
## should be empty table
d.t2[is.na(as.integer(RecipientFirstName)), .(RecipientFirstName, ID)]

d.t3.info <- fread(file.path(loc.data, "Data/SleepWell_T3.csv"),
                         nrows = 3, header = FALSE)
d.t3 <- fread(file = file.path(loc.data,
  "Data/SleepWell_T3.csv"), skip = "John_Doe@email.com", header=FALSE,
  logical01 = FALSE)
setnames(d.t3, make.names(unlist(d.t3.info[1,])))
## should be empty table
d.t3[is.na(as.integer(RecipientFirstName)), .(RecipientFirstName, ID)]

d.t4.info <- fread(file.path(loc.data, "Data/SleepWell_T4.csv"),
                         nrows = 3, header = FALSE)
d.t4 <- fread(file = file.path(loc.data,
  "Data/SleepWell_T4.csv"), skip = "John_Doe@email.com", header=FALSE,
  logical01 = FALSE)
setnames(d.t4, make.names(unlist(d.t4.info[1,])))
## should be empty table
d.t4[is.na(as.integer(RecipientFirstName)), .(RecipientFirstName, ID)]

d.diaries <- fread(
  file = file.path(loc.data, "Data/SleepWell_Diaries.csv"))

d.chemo <- fread(
  file = file.path(loc.data, "Data/SleepWell_ChemoDates.csv"))
```

```r
edate <- "Mar_6_2020"
emails <- Reduce(
  function(d1, d2) {
    merge(d1, d2,
          by = c("ID", "Email1Date"),
          all = TRUE)
  }, list(
  E1 = fread(file.path(loc.data, paste0("Data/CBT campaign mailchimp opens/members_Email_1_opened_", edate, ".
   .(ID, Email1Date, E1 = Opens)],
  E2 = fread(file.path(loc.data, paste0("Data/CBT campaign mailchimp opens/members_Email2_opened_", edate, ".c
   .(ID, Email1Date, E2 = Opens)],
  E3 = fread(file.path(loc.data, paste0("Data/CBT campaign mailchimp opens/members_Email3_opened_", edate, ".c
   .(ID, Email1Date, E3 = Opens)],
  E4a = fread(file.path(loc.data, paste0("Data/CBT campaign mailchimp opens/members_Email4_opened_", edate, ".
   .(ID, Email1Date, E4a = Opens)],
  E4b = fread(file.path(loc.data, paste0("Data/CBT campaign mailchimp opens/members_Email4_B_opened_", edate,
   .(ID, Email1Date, E4b = Opens)],
  E5 = fread(file.path(loc.data, paste0("Data/CBT campaign mailchimp opens/members_Email5_opened_", edate, ".c
   .(ID, Email1Date, E5 = Opens)],
  E6 = fread(file.path(loc.data, paste0("Data/CBT campaign mailchimp opens/members_Email6_opened_", edate, ".c
   .(ID, Email1Date, E6 = Opens)],
  E7 = fread(file.path(loc.data, paste0("Data/CBT campaign mailchimp opens/members_Email7_opened_", edate, ".c
   .(ID, Email1Date, E7 = Opens)],
  R1 = fread(file.path(loc.data, paste0("Data/Relax campaign mailchimp opens/members_Relaxation_1_opened_", ed
   .(ID, Email1Date, R1 = Opens)],
  R2 = fread(file.path(loc.data, paste0("Data/Relax campaign mailchimp opens/members_Relaxation_2_opened_", ed
   .(ID, Email1Date, R2 = Opens)]))

actiFiles <- list.files(
  file.path(loc.data, "ActiGraph_Scoring"),
  pattern = "actigraph_score.*\\.xlsx")

acti <- lapply(actiFiles, function(v) {
  readACTISLEEPXLSX(file.path(loc.data, "ActiGraph_Scoring", v))
})

## assert no duplicated IDs across acti files
stopifnot(identical(0L, anyDuplicated(sapply(acti, function(v) unique(v$ID)))))

actiAll <- do.call(rbind, acti)

rar <- as.data.table(read_excel(file.path(loc.data, "Data/activity_output_2020_12_11.xlsx")))

hashDataset(rar, "rar_hash.txt")
hashDataset(actiAll, "actiAll_hash.txt")
hashDataset(emails, "emails_hash.txt")
hashDataset(d.diaries, "d.diaries_hash.txt")
hashDataset(d.chemo, "d.chemo_hash.txt")
hashDataset(rData, "redcapdata_hash.txt")
hashDataset(d.t0, "d.t0_hash.txt")
hashDataset(d.t1, "d.t1hash.txt")
hashDataset(d.t2, "d.t2_hash.txt")
hashDataset(d.t3, "d.t3_hash.txt")
hashDataset(d.t4, "d.t4_hash.txt")

save(acti, actiAll, emails, d.diaries, d.chemo, rData, d.t0, d.t1, d.t2, d.t3, d.t4, rar,
     file = file.path(loc.data, "Data", "SleepWell_Raw.RData"),
     compress = "xz")
} else {
```

```
  load(file.path(loc.data, "Data", "SleepWell_Raw.RData"))
}
```

```
emails[, Opened := rowSums(!is.na(emails[, .(
  E1, E2, E3, E4a, E4b, E5, E6, E7, R1, R2)]))]
emails[, TotalOpens := rowSums(emails[, .(
 E1, E2, E3, E4a, E4b, E5, E6, E7, R1, R2)], na.rm=TRUE)]
emails[, OldID := ID]
emails[, ID := as.integer(gsub("SW", "", OldID, ignore.case = TRUE))]
emails <- emails[!is.na(ID) & ID < 200]
emails[, Email1Date := as.Date(Email1Date, format = "%Y-%m-%d")]

d.diaries[, SOL := as.numeric(SOL)] ## should not be needed long term
d.diaries[, BTDT := as.POSIXct(paste0(Date, " ", BT, ":00"),
                               format = "%d/%m/%Y %H:%M:%S")]
d.diaries[, RTDT := as.POSIXct(paste0(Date, " ", RT, ":00"),
                               format = "%d/%m/%Y %H:%M:%S")]

d.diaries[, TIBs := as.numeric(difftime(RTDT, BTDT, units = "hours"))]
d.diaries[TIBs < 0, BTDT := BTDT - (60 * 60 * 24)]

d.diaries[, IBDT := as.POSIXct(paste0(Date, " ", IntoB, ":00"),
                               format = "%d/%m/%Y %H:%M:%S")]
d.diaries[, OBDT := as.POSIXct(paste0(Date, " ", OutB, ":00"),
                               format = "%d/%m/%Y %H:%M:%S")]

d.diaries[, TIB2s := as.numeric(difftime(OBDT, IBDT, units = "hours"))]
d.diaries[TIB2s < 0, IBDT := IBDT - (60 * 60 * 24)]

## fix in bed times off by 24 hours due to lack of OB
d.diaries[abs(difftime(BTDT, IBDT, units = "hours")) > 20 & is.na(OBDT), IBDT := IBDT - (60 * 60 * 24)]

## fix sleep times off by 24 hours due to lack of RT
d.diaries[abs(difftime(BTDT, IBDT, units = "hours")) > 20 & is.na(RTDT), BTDT := BTDT - (60 * 60 * 24)]

## remake clean TIBs
d.diaries[, TIBs := as.numeric(difftime(RTDT, BTDT, units = "hours"))]
d.diaries[, TIB2s := as.numeric(difftime(OBDT, IBDT, units = "hours"))]

d.diaries[nzchar(RT),
          RTHabitual := meanCircular(chron(times. = paste0(RT, ":00")), max = 1),
          by = ID]
d.diaries[nzchar(StartG) & StartG %!in% c("N/A", "No", "no", "-"),
          StartGDT := as.POSIXct(paste0(Date, " ", StartG, ":00"),
                                 format = "%d/%m/%Y %H:%M:%S")]

d.diaries[, TSTs := TIBs - (SOL / 60) - (WASO / 60)]
d.diaries[TSTs < 0, TSTs := NA_real_]
d.diaries[, SEs := (TSTs / TIBs) * 100]
d.diaries[, SEstrans := 100 - sqrt(100 - SEs)]
d.diaries[, Date := as.Date(Date, "%d/%m/%Y")]
d.diaries[nzchar(StartG) & StartG %!in% c("N/A", "No", "no", "-"),
          StartG2 := chron(times. = paste0(StartG, ":00"))]
d.diaries[, StartG := StartG2]
d.diaries[, StartG2 := NULL]
d.diaries[nzchar(StopG) & StopG %!in% c("N/A", "N/A`", "No", "no", "n", "-"),
          StopG2 := chron(times. = paste0(StopG, ":00"))]
d.diaries[, StopG := StopG2]
d.diaries[, StopG2 := NULL]
d.diaries[, GTime := as.numeric(StopG - StartG) * 24 * 60]
```

```
d.diaries[, GRTDiff := as.numeric(StartG - RTHabitual) * 24 * 60]
d.diaries[, LightGood := (GRTDiff %gl% c(-60, 240)) & (GTime >= 20)]
d.diaries[, PropLightGoodCons := sum(LightGood, na.rm = TRUE)/42, by = ID]
d.diaries[, PropLightGood := sum(LightGood, na.rm = TRUE)/.N, by = ID]
## d.diaries[!TIBs %gl% c(3.5, 12.5), .(ID, Date, TIBs)]
## d.diaries[TSTs < 0, .(ID, Date, TSTs)]

actiAll[, RTDate := as.Date(format(RTDATETIMEacti, format = "%Y-%m-%d"))]
actiAll[, TSTacti := TSTacti / 60]
actiAll[, TIBacti := TIBacti / 60]
actiAll[, SEactitrans := 100 - sqrt(100 - SEacti)]
## actiAll[!TIBacti %gl% c(3.5, 12.5), .(ID, BTDATETIMEacti, TIBacti)]

d.sleep <- merge(
  d.diaries,
  actiAll,
  by.x = c("ID", "Date"),
  by.y = c("ID", "RTDate"),
  all = TRUE)
d.sleep <- merge(
  d.sleep,
  emails[, .(ID, Email1Date)],
  by = "ID",
  all.x = TRUE)
d.sleep[, DayofWeek := weekdays(Date)]
d.sleep[, Weekend := as.integer(is.weekend(Date))]
d.sleep[, Day := as.integer(Date - Email1Date), by = ID]

## list of all variable names
allnames <- table(c(
  names(d.t0),
  names(d.t1), names(d.t2),
  names(d.t3), names(d.t4)))

rnames <- names(allnames)[allnames > 1]
unames <- c("RecipientFirstName", names(allnames)[allnames == 1])

d.t0u <- d.t0[, unames %sin% names(d.t0), with = FALSE]
d.t1u <- d.t1[, unames %sin% names(d.t1), with = FALSE]
d.t3u <- d.t3[, unames %sin% names(d.t3), with = FALSE]

d.allu <- merge(d.t0u, d.t1u, by = "RecipientFirstName",
                all = TRUE)
d.allu <- merge(d.allu, d.t3u, by = "RecipientFirstName",
                all = TRUE)

stopifnot(!anyDuplicated(d.allu$RecipientFirstName))

d.t0r <- d.t0[, rnames %sin% names(d.t0), with = FALSE]
d.t1r <- d.t1[, rnames %sin% names(d.t1), with = FALSE]
d.t2r <- d.t2[, rnames %sin% names(d.t2), with = FALSE]
d.t3r <- d.t3[, rnames %sin% names(d.t3), with = FALSE]
d.t4r <- d.t4[, rnames %sin% names(d.t4), with = FALSE]

## fill in any missing variables for each dataset
for (v in rnames[!rnames %in% names(d.t0r)]) {
  d.t0r[, (v) := NA]
}
for (v in rnames[!rnames %in% names(d.t1r)]) {
```

```
  d.t1r[, (v) := NA]
}
for (v in rnames[!rnames %in% names(d.t2r)]) {
  d.t2r[, (v) := NA]
}
for (v in rnames[!rnames %in% names(d.t3r)]) {
  d.t3r[, (v) := NA]
}
for (v in rnames[!rnames %in% names(d.t4r)]) {
  d.t4r[, (v) := NA]
}

## order each dataset the same way
setcolorder(d.t0r, rnames)
setcolorder(d.t1r, rnames)
setcolorder(d.t2r, rnames)
setcolorder(d.t3r, rnames)
setcolorder(d.t4r, rnames)

## add survey labels to each dataset
d.t0r[, Survey := "Screening"]
d.t1r[, Survey := "Baseline"]
d.t2r[, Survey := "Midpoint"]
d.t3r[, Survey := "Post"]
d.t4r[, Survey := "Follow-Up"]

## combine all datasets into one long daily one
d.allr <- rbind(d.t0r, d.t1r, d.t2r, d.t3r, d.t4r)
d.allr[, Survey := factor(Survey,
  levels = c("Screening", "Baseline", "Midpoint", "Post", "Follow-Up"))]

d.all <- merge(d.allr, d.allu, by = "RecipientFirstName",
               all = TRUE)

rData$record_id <- as.integer(rData$record_id)
d.all <- merge(d.all, rData,
               by.x = "RecipientFirstName",
               by.y = "record_id", all=TRUE)

d.all <- merge(
  emails[, .(ID, Email1Date, Opened, TotalOpens)],
  d.all[, -which(names(d.all) == "ID"), with = FALSE],
  by.x = "ID",
  by.y = "RecipientFirstName",
  all = TRUE)

d.chemo[, ChemoEnd := ChemoEndDate]
d.chemo[, ChemoEndDate := as.Date(ChemoEndDate)]
d.chemo[ChemoEnd == "ongoing", ChemoEndDate := as.Date("2099-12-31")]
d.chemo[, ID := as.integer(gsub("SW", "", ID))]

d.all <- merge(
  d.all, d.chemo,
  by = "ID",
  all = TRUE)

expect_false(anyNA(d.all$condition))
expect_false(anyNA(d.all$baseline_isi_high))
expect_false(anyNA(d.all$baseline_stage_high))
```

```r
d.all$ISI_Total <- rowMeansHalf(d.all[, paste0("ISI_", 1:7), with = FALSE]) * 7
d.all$PSA_Somatic <- rowMeansHalf(d.all[, paste0("PSA_", 1:8), with = FALSE]) * 8
d.all$PSA_Cognitive <- rowMeansHalf(d.all[, paste0("PSA_", 9:16), with = FALSE]) * 8
d.all$PSA_Total <- rowMeansHalf(d.all[, paste0("PSA_", 1:16), with = FALSE]) * 16
d.all$IES_Intrusive <- rowMeansHalf(d.all[, paste0("IES_", 1:8), with = FALSE]) * 8
d.all$FIRST <- rowMeansHalf(d.all[, paste0("FIRST_", 1:9), with = FALSE]) * 9
d.all$DBAS_Total <- rowMeansHalf(d.all[, paste0("DBAS_", 1:16), with = FALSE])
d.all$DBAS_Consequences <- rowMeansHalf(d.all[, paste0("DBAS_", c(5,7,9,12,16)), with = FALSE])
d.all$DBAS_Worry <- rowMeansHalf(d.all[, paste0("DBAS_", c(3,4,8,10,11,14)), with = FALSE])
d.all$DBAS_Expectations <- rowMeansHalf(d.all[, paste0("DBAS_", c(1,2)), with = FALSE])
d.all$DBAS_Medication <- rowMeansHalf(d.all[, paste0("DBAS_", c(6,13,15)), with = FALSE])

tmp <- promis.sf$DEP8a$Score(d.all[, paste0("DEP_", 1:8), with = FALSE])
d.all$DEP_Total <- tmp$Total
d.all$DEP_TScore <- tmp$TScore

tmp <- promis.sf$ANX8a$Score(d.all[, paste0("ANX_", 1:8), with = FALSE])
d.all$ANX_Total <- tmp$Total
d.all$ANX_TScore <- tmp$TScore

tmp <- promis.sf$FAT8a$Score(d.all[, paste0("FA_", 1:8), with = FALSE])
d.all$FA_Total <- tmp$Total
d.all$FA_TScore <- tmp$TScore

tmp <- promis.sf$SDI8a$Score(d.all[, paste0("SD_", 1:8), with = FALSE])
d.all$SD_Total <- tmp$Total
d.all$SD_TScore <- tmp$TScore

tmp <- promis.sf$SRI8a$Score(d.all[, paste0("SRI_", 1:8), with = FALSE])
d.all$SRI_Total <- tmp$Total
d.all$SRI_TScore <- tmp$TScore

tmp <- promis.sf$PAI3a$Score(d.all[, paste0("PAIN_", 1:3), with = FALSE])
d.all$PAIN_Total <- tmp$Total
d.all$PAIN_TScore <- tmp$TScore

d.all$rMEQ_Total <- rowSums(d.all[, paste0("MEQ_", 1:5), with = FALSE])

d.all[, CEQ_Cred := rowMeansHalf(d.all[, .(EXP_1, EXP_2, EXP_3)])]
d.all[, CEQ_Exp := rowMeansHalf(scale(d.all[, .(EXP_4, EXP_5, EXP_6)]))]

d.all[, EndDate := as.Date(format(as.POSIXct(EndDate, format = "%Y-%m-%d %H:%M:%S"),
                                  format = "%Y-%m-%d"))]
d.all[, DOB := as.Date(DOB, format = "%d/%m/%Y")]
d.all[, Age := as.numeric(EndDate - DOB)/365.25]

d.all[, EDU2 := cut(EDU, breaks = c(0, 4, 8), labels = c("< Bachelor", ">= Bachelor"))]
d.all[, EDU3 := cut(EDU, breaks = c(0, 4, 5, 8), labels = c("< Bachelor", "Bachelor", "> Bachelor"))]
d.all[, EDU := factor(EDU,
  levels = 1:7,
  labels = c("< Year 12", "Year 12", "Cert III/IV",
             "Adv Dip/Dip", "Bachelor",
             "Grad Dip/Cert", "Postgrad"))]

d.all[, ETH2 := factor(fifelse(ETH == 1, "White", "Non-White", na = NA_character_))]
d.all[, ETH := factor(ETH,
  levels = 1:5,
  labels = c("White", "Indigenous", "Asian", "Indian", "Other"))]
```

```
d.all[, MARI2 := factor(fifelse(MARI == 1, "Married/living as", "Other", na = NA_character_))]
d.all[, MARI := factor(MARI,
  levels = 1:5,
  labels = c("Married/living as", "Committed, live sep", "Single",
             "Divorced/separated", "Widowed" ))]


d.all[, CHILD2 := factor(fifelse(CHILD == 0, "No Children", ">= 1 Child", na = NA_character_))]
d.all[CHILD==0, CHILDLV := 0L]

d.all[, CHILDLV18 := factor(fifelse(CHILDLV > 0 & CHILDAG < 18, "Child < 18y", "No Child < 18y"))]

d.all[INCOM == 13, INCOM := NA_integer_]
d.all[, INCOM3 := cut(INCOM, breaks = c(0, 5, 10, 13),
                      labels = c("<= $50,000", "$50,001 - $100,000", "> $100,000"))]

d.all[, EMPL3 := "Not employed"]
d.all[grepl("1", EMPL), EMPL3 := "Retired"]
d.all[grepl("2|3", EMPL), EMPL3 := "Employed"]
d.all[is.na(EMPL) | !nzchar(EMPL), EMPL3 := NA_character_]
d.all[, EMPL3 := factor(EMPL3, levels = c("Employed", "Retired", "Not employed"))]

d.all[, Migr := factor(Migr,
  levels = c(0, 1),
  labels = c("No", "Yes"))]

d.all[, SMO := factor(SMO,
  levels = c(2, 1),
  labels = c("No", "Yes"))]

d.all[, CAN1 := factor(CAN1,
  levels = c(0, 1),
  labels = c("No", "Yes"))]

d.all[CANSTG == 5, CANSTG := CanStag]
d.all[, CANSTG := factor(CANSTG,
  levels = 1:4,
  labels = c(1, 2, 3, 4))]

d.all[CANMET == -9 & CANSTG %in% c("1", "2"), CANMET := 0]
d.all[, CANMET := factor(CANMET,
  levels = c(0, 1),
  labels = c("No", "Yes"))]

d.all[, MENOPA := factor(MENOPA,
  levels = c(-9, 0, 1),
  labels = c("Unknown", "No", "Yes"))]
d.all[, MENOCUR := factor(MENOCUR,
  levels = c(-9, 0, 1),
  labels = c("Unknown", "No", "Yes"))]

d.all[MENOPA == "Yes", MENOPAUSE := "Past"]
d.all[is.na(MENOPAUSE) & MENOCUR == "Yes", MENOPAUSE := "Current"]
d.all[is.na(MENOPAUSE) & MENOCUR == "No", MENOPAUSE := "No"]
d.all[MENOPA == "No" & MENOCUR == "Unknown", MENOPAUSE := "No"]
d.all[MENOPA == "Unknown" & MENOCUR == "Unknown", MENOPAUSE := NA_character_]

d.all[, SURG := factor(SURG,
  levels = c(2, 1),
```

```
  labels = c("No", "Yes"))]

d.all[, CHEM := factor(CHEM,
  levels = c(0, 1),
  labels = c("No", "Yes"))]

d.all[, RADI := factor(RADI,
  levels = c(-9, 0, 1),
  labels = c("Unknown", "No", "Yes"))]
d.all[, RADIPLN := factor(RADIPLN,
  levels = c(-9, 0, 1),
  labels = c("Unknown", "No", "Yes"))]

d.all[RADI == "Yes", RADIATION := "Current"]
d.all[is.na(RADIATION) & RADIPLN == "Yes", RADIATION := "Planned"]
d.all[is.na(RADIATION) & RADIPLN == "No", RADIATION := "No"]
d.all[RADI == "No" & RADIPLN == "Unknown", RADIATION := "UnknownPlan"]
d.all[RADI == "Unknown" & RADIPLN == "Unknown", RADIATION := NA_character_]

d.all[, HORM := factor(HORM,
  levels = c(-9, 0, 1),
  labels = c("Unknown", "No", "Yes"))]
d.all[, HORMPLN := factor(HORMPLN,
  levels = c(-9, 0, 1),
  labels = c("Unknown", "No", "Yes"))]

d.all[HORM == "Yes", HORMONAL := "Current"]
d.all[is.na(HORMONAL) & HORMPLN == "Yes", HORMONAL := "Planned"]
d.all[is.na(HORMONAL) & HORMPLN == "No", HORMONAL := "No"]
d.all[HORM == "No" & HORMPLN == "Unknown", HORMONAL := "UnknownPlan"]
d.all[HORM == "Unknown" & HORMPLN == "Unknown", HORMONAL := NA_character_]

d.all[, BIO := factor(BIO,
  levels = c(-9, 0, 1),
  labels = c("Unknown", "No", "Yes"))]
d.all[, BIOPLN := factor(BIOPLN,
  levels = c(-9, 0, 1),
  labels = c("Unknown", "No", "Yes"))]

d.all[BIO == "Yes", BIOLOGICAL := "Current"]
d.all[is.na(BIOLOGICAL) & BIOPLN == "Yes", BIOLOGICAL := "Planned"]
d.all[is.na(BIOLOGICAL) & BIOPLN == "No", BIOLOGICAL := "No"]
d.all[BIO == "No" & BIOPLN == "Unknown", BIOLOGICAL := "UnknownPlan"]
d.all[BIO == "Unknown" & BIOPLN == "Unknown", BIOLOGICAL := NA_character_]


d.all[PSY < 0, PSY := NA_integer_]
d.all[, PSY := factor(PSY,
  levels = c(0, 1), labels = c("No", "Yes"))]

d.all[GRP < 0, GRP := NA_integer_]
d.all[, GRP := factor(GRP,
  levels = c(0, 1), labels = c("No", "Yes"))]

d.all[MED < 0, MED := NA_integer_]
d.all[, MED := factor(MED,
  levels = c(0, 1), labels = c("No", "Yes"))]

d.all[PSYSL < 0, PSYSL := NA_integer_]
```

```r
d.all[, PSYSL := factor(PSYSL,
  levels = c(0, 1), labels = c("No", "Yes"))]

d.all[MEDSL < 0, MEDSL := NA_integer_]
d.all[, MEDSL := factor(MEDSL,
  levels = c(0, 1), labels = c("No", "Yes"))]

d.all[HERSL < 0, HERSL := NA_integer_]
d.all[, HERSL := factor(HERSL,
  levels = c(0, 1), labels = c("No", "Yes"))]

d.all[, MHTX := factor(fifelse(PSY == "Yes" | GRP == "Yes" | MED == "Yes", "Yes", "No", na = NA_character_),
                       levels = c("No", "Yes"))]
d.all[, SleepTX := factor(fifelse(PSYSL == "Yes" | MEDSL == "Yes" | HERSL == "Yes", "Yes", "No", na = NA_chara
                       levels = c("No", "Yes"))]

d.all[, PropEmailOpen := Opened / ifelse(condition == "CBT+", 8, 2)]
d.all[, Practice := mean(COMP_2, na.rm=TRUE), by = ID]

tmpdat <- d.all[!is.na(Email1Date)][, .(
                 Email1Date, ChemoEnd, ChemoEndDate,
                 Post = na.omit(EndDate[Survey == "Post"])[1],
                 FU = na.omit(EndDate[Survey == "Follow-Up"])[1]), by = ID][
  !duplicated(ID) & !is.na(ChemoEndDate)]
tmpdat[ChemoEndDate <= (Email1Date + 7 * 6), ChemoFinish := "during Tx"]
tmpdat[is.na(ChemoFinish) & ChemoEndDate <= (Email1Date + 7 * 18), ChemoFinish := "during FU"]
tmpdat[is.na(ChemoFinish) & ChemoEndDate > (Email1Date + 7 * 18), ChemoFinish := "ongoing"]

d.all <- merge(
  d.all,
  tmpdat[, .(ID, ChemoFinish)],
  by = "ID", all.x = TRUE)

## create a dropout variable
tmpdat <- d.all[, .(
  MID = length(ISI_Total[Survey == "Midpoint"]) > 0 && !is.na(ISI_Total[Survey == "Midpoint"]),
  POST = length(ISI_Total[Survey == "Post"]) > 0 && !is.na(ISI_Total[Survey == "Post"]),
  FU = length(ISI_Total[Survey == "Follow-Up"]) > 0 && !is.na(ISI_Total[Survey == "Follow-Up"])),
  by = ID][, .(ID, Dropout = factor(as.integer((MID + POST + FU) == 0), levels = 0:1, labels = c("Completer",

d.all <- merge(
  d.all,
  tmpdat,
  by = "ID", all.x = TRUE)

## Remove unnecessary variables
d.all[, DistributionChannel := NULL]
d.all[, ExternalReference := NULL]
d.all[, IPAddress := NULL]
d.all[, LocationLatitude := NULL]
d.all[, LocationLongitude := NULL]
d.all[, ResponseId := NULL]
d.all[, RecipientEmail := NULL]
d.all[, RecipientLastName := NULL]
d.all[, Status := NULL]
d.all[, UserLanguage := NULL]
d.all[, Progress := NULL]

hashDataset(d.all, "d.all_hash.txt")
```

```
hashDataset(d.sleep, "d.sleep_hash.txt")

saveRDS(d.all,
     file = file.path(loc.data, "Data", "SleepWell_Surveys.RDS"),
     compress = "xz")
saveRDS(d.sleep,
     file = file.path(loc.data, "Data", "SleepWell_Sleep.RDS"),
     compress = "xz")
```

Process rest activity rhythm data

```
## convert IDs to numbers
rar[, ID := as.integer(gsub("SW", "", ID))]

d.rar <- merge(
  rar,
  d.all[Survey == "Screening", .(ID, condition, baseline_isi_high, baseline_stage_high, CanStag)],
  by.x = "ID",
  by.y = "ID",
  all.x = TRUE)

hashDataset(d.rar, "d.rar_merged_hash.txt")

saveRDS(d.rar,
     file = file.path(loc.data, "Data", "SleepWell_RAR.RDS"),
     compress = "xz")

## m.isnull <- lmer(IS ~ factor(Block) + baseline_isi_high + baseline_stage_high + (1 | ID2), data = rar2)
## m.is <- update(m.isnull, . ~ . + condition)
## anova(m.isnull, m.is)
## plot(emmeans(m.is, ~ condition | Block))

## m.ivnull <- lmer(IV ~ factor(Block) + baseline_isi_high + baseline_stage_high + (1 | ID2), data = rar2)
## m.iv <- update(m.ivnull, . ~ . + condition)
## anova(m.ivnull, m.iv)
## plot(emmeans(m.iv, ~ condition | Block))

## m.ranull <- lmer(RA ~ factor(Block) + baseline_isi_high + baseline_stage_high + (1 | ID2), data = rar2)
## m.ra <- update(m.ranull, . ~ . + condition)
## anova(m.ranull, m.ra)
## plot(emmeans(m.ra, ~ condition | Block))

## m.l5null <- lmer(L5 ~ factor(Block) + baseline_isi_high + baseline_stage_high + (1 | ID2), data = rar2)
## m.l5 <- update(m.l5null, . ~ . + condition)
## anova(m.l5null, m.l5)
## plot(emmeans(m.l5, ~ condition | Block))

## m.m10null <- lmer(M10 ~ factor(Block) + baseline_isi_high + baseline_stage_high + (1 | ID2), data = rar2)
## m.m10 <- update(m.m10null, . ~ . + condition)
## anova(m.m10null, m.m10)
## plot(emmeans(m.m10, ~ condition | Block))
```