Pre Post Analysis

Max Bi

2025-03-01

```
KOH1 = as.Date("2023-04-05")
o1_bp = read_csv("Analysis Data/Obj1BPPrePost.csv")
```

```
## New names:
## Rows: 240 Columns: 16
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (1): Sex dbl (13): ...1, UniqueIdentifier, sys, dia, KOH, koh.counts, KOH.none,
## KOH.... date (2): BPDate, KOHDate
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
names(o1_bp)[1] = "row_id"
o1_a1c = read_csv("Analysis Data/Obj1A1cPrePost.csv")
```

```
## New names:
## Rows: 394 Columns: 15
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (1): Sex dbl (12): ...1, UniqueIdentifier, A1c, KOH, koh.counts, KOH.none,
## KOH.one, ... date (2): A1cDate, KOHDate
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
names(o1_a1c)[1] = "row_id"
o2_bp = read_csv("Analysis Data/Obj2BPPrePost.csv")
```

```
## New names:
## Rows: 9642 Columns: 12
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (2): Group, Sex dbl (9): ...1, UniqueIdentifier, Systolic, Diastolic, Marsh,
## age, IncomeLev... date (1): Date
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
names(o2_bp)[1] = "row_id"
o2_a1c = read_csv("Analysis Data/Obj2A1cPrePost.csv")
```

```
## New names:
## Rows: 5118 Columns: 11
## -- Column specification
```

```
## ------------------------------------------------------- Delimiter: "," chr
## (2): Group, Sex dbl (8): ...1, UniqueIdentifier, A1c, Marsh, age, IncomeLevel,
## BLACERISK, a... date (1): Date
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
names(o2_a1c)[1] = "row_id"
DAYS_IN_MONTH = 30.4
```

```
# attendees
#   measured before 1st KOH visit: change date to 1st KOH visit
#   measured after 1st KOH visit: keep date
# non-attendees
#   measured before KOH 1: change date to KOH 1
#   measured after KOH 1: DNE
std_times = o1_bp %>%
  group_by(UniqueIdentifier) %>%
  summarize(
    pre_Date = min(BPDate),
    post_Date = max(BPDate),
    std_time = ifelse(BPDate == pre_Date, 0, ifelse(
      KOH == 0,
      as.numeric(post_Date - KOH1),
      ifelse(pre_Date < KOH1, as.numeric(post_Date - KOH1), as.numeric(post_Date - KOHDate))
    ))
  ) %>%
  select(std_time) %>%
  ungroup() %>%
  mutate(
    row_id = as.numeric(o1_bp[["row_id"]]),
    std_months = std_time / DAYS_IN_MONTH
  ) %>%
  select(std_time, row_id)
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## `summarise()` has grouped output by 'UniqueIdentifier'. You can override using
## the `.groups` argument.
## Adding missing grouping variables: `UniqueIdentifier`
```

```
o1_bp = left_join(o1_bp, std_times, by="row_id")
o1_bp = o1_bp %>% mutate(
  Sex = ifelse(o1_bp$Sex == "M", 1, 0),
  koh_cat = case_when(
    KOH.none == 1 ~ "None",
    KOH.one == 1 ~ "One",
    KOH.mult == 1 ~ "Multiple"
  )
)
```

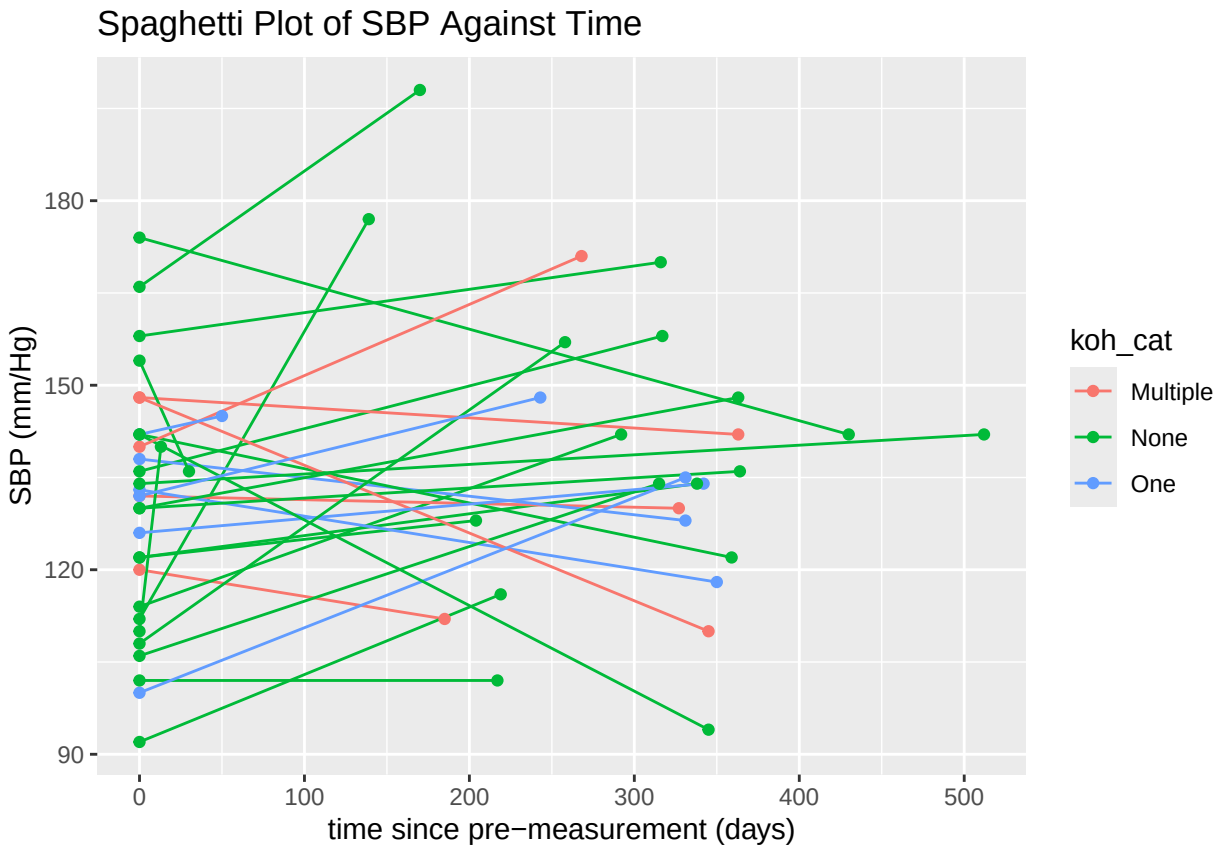
```
std_times = o1_a1c %>%
  group_by(UniqueIdentifier) %>%
  summarize(
    pre_Date = min(A1cDate),
    post_Date = max(A1cDate),
    std_time = ifelse(A1cDate == pre_Date, 0, ifelse(
      KOH == 0,
      as.numeric(post_Date - KOH1),
      ifelse(pre_Date < KOH1, as.numeric(post_Date - KOH1), as.numeric(post_Date - KOHDate))
    ))
  ) %>%
  select(std_time) %>%
  ungroup() %>%
  mutate(
    row_id = as.numeric(o1_a1c[["row_id"]]),
    std_months = std_time / DAYS_IN_MONTH
  ) %>%
  select(std_time, row_id)
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

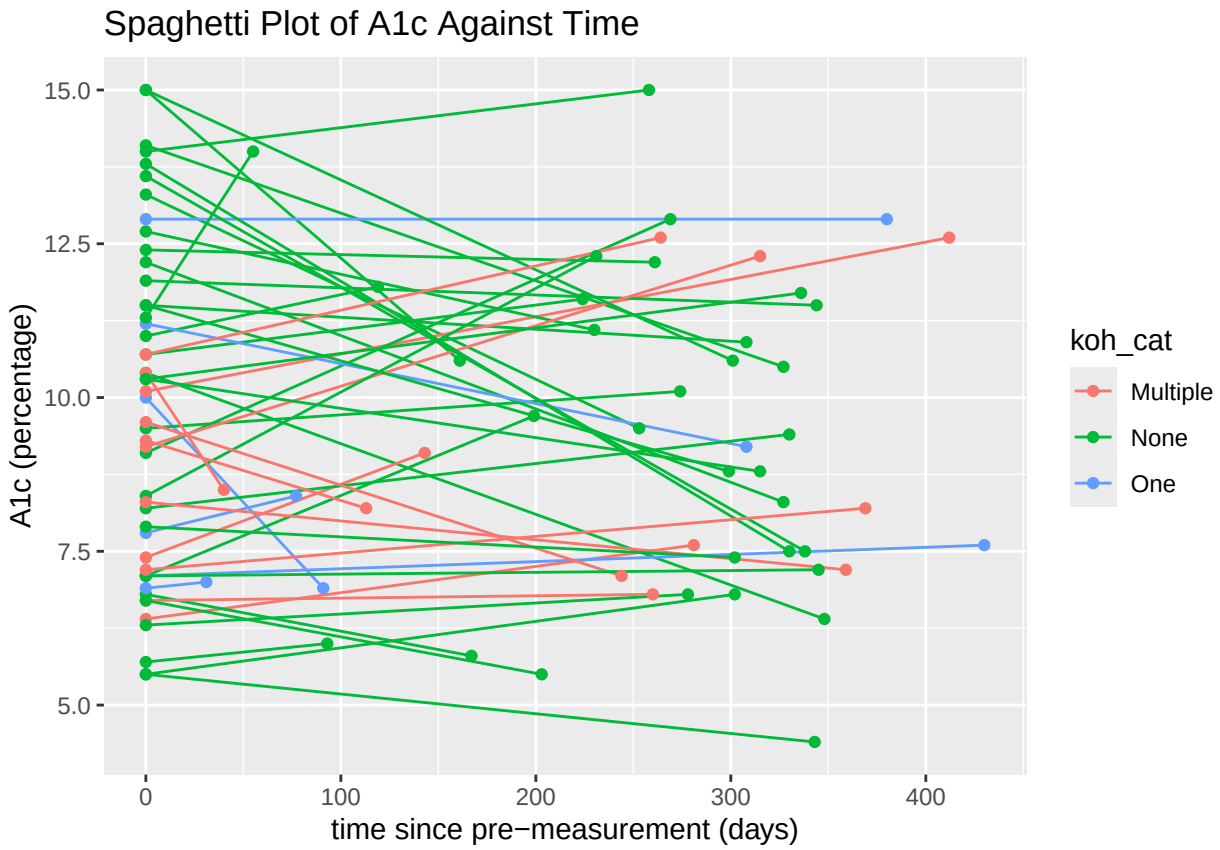
```
## `summarise()` has grouped output by 'UniqueIdentifier'. You can override using
## the `.groups` argument.
## Adding missing grouping variables: `UniqueIdentifier`
```

```
o1_a1c = left_join(o1_a1c, std_times, by="row_id")
o1_a1c = o1_a1c %>% mutate(
  Sex = ifelse(o1_a1c$Sex == "M", 1, 0),
  koh_cat = case_when(
    KOH.none == 1 ~ "None",
    KOH.one == 1 ~ "One",
    KOH.mult == 1 ~ "Multiple"
  )
)
```

```
o1_bp_ids = o1_bp$UniqueIdentifier %>% unique() %>% sample(0.5 * 0.25 * nrow(o1_bp))
o1_bp_pdt = o1_bp %>% filter(UniqueIdentifier %in% o1_bp_ids)
ggplot(data=o1_bp_pdt, mapping=aes(x=std_time, y=sys, group=UniqueIdentifier, color=koh_cat)) +
  geom_line() +
  geom_point() +
  labs(
    title="Spaghetti Plot of SBP Against Time",
    x="time since pre-measurement (days)",
    y="SBP (mm/Hg)"
  )
```




```
o1_a1c_ids = o1_a1c$UniqueIdentifier %>% unique() %>% sample(0.5 * 0.25 * nrow(o1_a1c))
o1_a1c_pdt = o1_a1c %>% filter(UniqueIdentifier %in% o1_a1c_ids)
ggplot(data=o1_a1c_pdt, mapping=aes(x=std_time, y=A1c, group=UniqueIdentifier, color=koh_cat)) +
  geom_line() +
  geom_point() +
  labs(
    title="Spaghetti Plot of A1c Against Time",
    x="time since pre-measurement (days)",
    y="A1c (percentage)"
  )
```




```
mod_o1_bp = lme(
  fixed = sys ~ std_time * koh_cat + age + Sex + IncomeLevel + BLACERISK + avg.bmi,
  random = ~ std_time | UniqueIdentifier,
  data = o1_bp,
  method = "REML"
)
summary(mod_o1_bp)
```

```
## Linear mixed-effects model fit by REML
##   Data: o1_bp
##        AIC      BIC    logLik
##   2136.146 2187.651 -1053.073
##
## Random effects:
##  Formula: ~std_time | UniqueIdentifier
##  Structure: General positive-definite, Log-Cholesky parametrization
##             StdDev       Corr
## (Intercept) 1.156318e+01 (Intr)
## std_time    2.799223e-05 0
## Residual    1.640765e+01
##
## Fixed effects:  sys ~ std_time * koh_cat + age + Sex + IncomeLevel + BLACERISK +      avg.bmi
##                          Value Std.Error  DF   t-value p-value
## (Intercept)          102.23340 15.629251 117  6.541158  0.0000
## std_time               0.00200  0.017866 117  0.111783  0.9112
## koh_catNone           -1.47425  4.681832 112 -0.314887  0.7534
## koh_catOne             0.29680  6.613289 112  0.044879  0.9643
```

```
## age                    0.15774  0.158445 112  0.995541  0.3216
## Sex                    5.75144  3.171216 112  1.813640  0.0724
## IncomeLevel           -0.00014  0.025000 112 -0.005722  0.9954
## BLACERISK             -0.63313  0.739841 112 -0.855770  0.3940
## avg.bmi                0.59073  0.297424 112  1.986151  0.0495
## std_time:koh_catNone  -0.00214  0.020023 117 -0.107107  0.9149
## std_time:koh_catOne    0.01113  0.029265 117  0.380448  0.7043
##  Correlation:
##                       (Intr) std_tm kh_ctN kh_ctO age    Sex    IncmLv BLACER
## std_time              -0.104
## koh_catNone           -0.338  0.456
## koh_catOne            -0.157  0.315  0.540
## age                   -0.775  0.008  0.207  0.040
## Sex                   -0.097  0.010  0.008 -0.024  0.036
## IncomeLevel           -0.228 -0.032  0.064  0.147  0.374 -0.176
## BLACERISK             -0.104 -0.041  0.036  0.053 -0.159 -0.194 -0.060
## avg.bmi               -0.685 -0.051 -0.058 -0.079  0.185  0.054 -0.132  0.131
## std_time:koh_catNone   0.123 -0.891 -0.515 -0.279 -0.026 -0.028  0.035  0.035
## std_time:koh_catOne    0.029 -0.610 -0.277 -0.515  0.026  0.036  0.015 -0.014
##                       avg.bm st_:_N
## std_time
## koh_catNone
## koh_catOne
## age
## Sex
## IncomeLevel
## BLACERISK
## avg.bmi
## std_time:koh_catNone   0.017
## std_time:koh_catOne    0.062  0.542
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -2.43656177 -0.57800078 -0.07670638  0.49839836  2.74007188
##
## Number of Observations: 240
## Number of Groups: 120
```

```
mod_o1_a1c = lme(
  fixed = A1c ~ std_time * koh_cat + age + Sex + IncomeLevel + BLACERISK + avg.bmi,
  random = ~ std_time | UniqueIdentifier,
  data = o1_a1c,
  method = "REML"
)
summary(mod_o1_a1c)
```

```
## Linear mixed-effects model fit by REML
##   Data: o1_a1c
##        AIC      BIC    logLik
##   1790.262 1849.482 -880.1309
##
## Random effects:
##  Formula: ~std_time | UniqueIdentifier
##  Structure: General positive-definite, Log-Cholesky parametrization
##             StdDev      Corr
```

```
## (Intercept) 2.235421644 (Intr)
## std_time    0.005043043 -0.483
## Residual    1.138103773
##
## Fixed effects:  A1c ~ std_time * koh_cat + age + Sex + IncomeLevel + BLACERISK +      avg.bmi
##                         Value Std.Error  DF   t-value p-value
## (Intercept)          14.596954 1.4762887 194  9.887602  0.0000
## std_time             -0.000538 0.0013744 194 -0.391276  0.6960
## koh_catNone           0.375221 0.4847303 189  0.774082  0.4398
## koh_catOne            0.683874 0.6466081 189  1.057633  0.2916
## age                  -0.056176 0.0151347 189 -3.711750  0.0003
## Sex                   0.001795 0.3252390 189  0.005520  0.9956
## IncomeLevel           0.001183 0.0022470 189  0.526627  0.5991
## BLACERISK            -0.050551 0.0818920 189 -0.617295  0.5378
## avg.bmi              -0.069183 0.0290626 189 -2.380485  0.0183
## std_time:koh_catNone -0.001170 0.0015414 194 -0.759173  0.4487
## std_time:koh_catOne  -0.000495 0.0021659 194 -0.228653  0.8194
##  Correlation:
##                      (Intr) std_tm kh_ctN kh_ctO age    Sex    IncmLv BLACER
## std_time             -0.140
## koh_catNone          -0.398  0.430
## koh_catOne           -0.284  0.324  0.585
## age                  -0.710 -0.006  0.170 -0.017
## Sex                  -0.049  0.000 -0.037 -0.028  0.015
## IncomeLevel          -0.283 -0.005  0.046  0.026  0.314 -0.178
## BLACERISK            -0.080  0.008  0.001  0.100 -0.183 -0.131 -0.035
## avg.bmi              -0.707  0.002  0.059  0.144  0.138  0.000  0.020  0.078
## std_time:koh_catNone  0.126 -0.892 -0.483 -0.289  0.000 -0.001  0.000 -0.012
## std_time:koh_catOne   0.098 -0.634 -0.274 -0.454 -0.002  0.010 -0.020 -0.004
##                      avg.bm st_:_N
## std_time
## koh_catNone
## koh_catOne
## age
## Sex
## IncomeLevel
## BLACERISK
## avg.bmi
## std_time:koh_catNone  0.003
## std_time:koh_catOne  -0.008  0.566
##
## Standardized Within-Group Residuals:
##        Min         Q1        Med         Q3        Max
## -1.8967875 -0.4061735 -0.0484614  0.3957781  2.0900603
##
## Number of Observations: 394
## Number of Groups: 197
```

```
# measured before KOH 1: change date to KOH 1
# measured after KOH 1: keep date
std_times = o2_bp %>%
  group_by(UniqueIdentifier) %>%
  summarize(
    pre_Date = min(Date),
```

```
    post_Date = max(Date),
    std_time = ifelse(
      Date == pre_Date,
      0,
      # ifelse(pre_Date < KOH1, as.numeric(post_Date - KOH1), as.numeric(post_Date - pre_Date))
      as.numeric(post_Date - pre_Date)
    )
  ) %>%
  select(std_time) %>%
  ungroup() %>%
  mutate(
    row_id = as.numeric(o2_bp[["row_id"]]),
    std_months = std_time / DAYS_IN_MONTH
  ) %>%
  select(std_time, row_id)
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## `summarise()` has grouped output by 'UniqueIdentifier'. You can override using
## the `.groups` argument.
## Adding missing grouping variables: `UniqueIdentifier`
```

```
o2_bp = left_join(o2_bp, std_times, by="row_id")
o2_bp = o2_bp %>% mutate(Sex = ifelse(o2_bp$Sex == "M", 1, 0))

std_times = o2_a1c %>%
  group_by(UniqueIdentifier) %>%
  summarize(
    pre_Date = min(Date),
    post_Date = max(Date),
    std_time = ifelse(
      Date == pre_Date,
      0,
      # ifelse(pre_Date < KOH1, as.numeric(post_Date - KOH1), as.numeric(post_Date - pre_Date))
      as.numeric(post_Date - KOH1)
    )
  ) %>%
  select(std_time) %>%
  ungroup() %>%
  mutate(
    row_id = as.numeric(o2_a1c[["row_id"]]),
    std_months = std_time / DAYS_IN_MONTH
  ) %>%
  select(std_time, row_id)
```

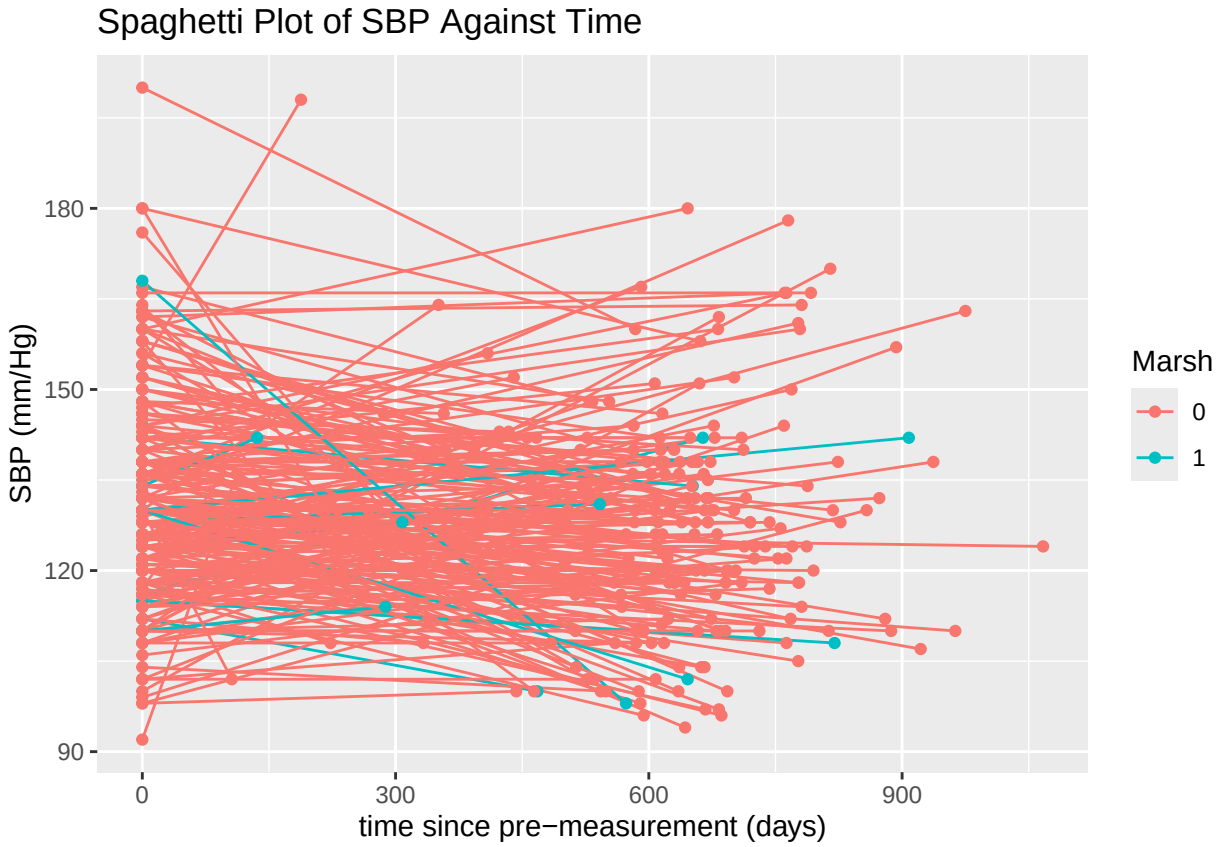
```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
```

```
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## `summarise()` has grouped output by 'UniqueIdentifier'. You can override using
## the `.groups` argument.
## Adding missing grouping variables: `UniqueIdentifier`
```

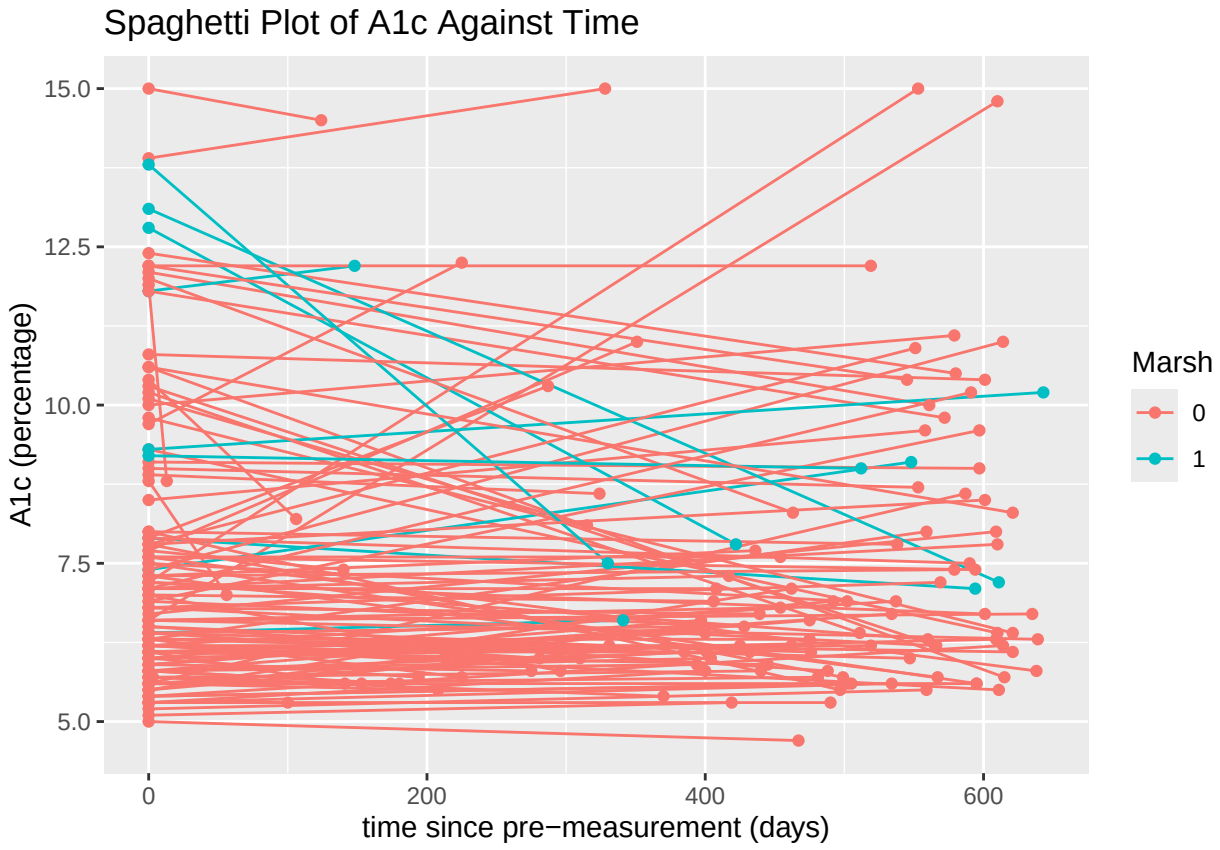
```
o2_a1c = left_join(o2_a1c, std_times, by="row_id")
o2_a1c = o2_a1c %>% mutate(Sex = ifelse(o2_a1c$Sex == "M", 1, 0))
```

```
o2_bp_ids = o2_bp$UniqueIdentifier %>% unique() %>% sample(0.5 * 0.05 * nrow(o2_bp))
o2_bp_pdt = o2_bp %>% filter(UniqueIdentifier %in% o2_bp_ids) %>% mutate(Marsh = as.factor(Marsh))
ggplot(data=o2_bp_pdt, mapping=aes(x=std_time, y=Systolic, group=UniqueIdentifier, color=Marsh)) +
  geom_line() +
  geom_point() +
  labs(
    title="Spaghetti Plot of SBP Against Time",
    x="time since pre-measurement (days)",
    y="SBP (mm/Hg)"
  )
```




```
o2_a1c_ids = o2_a1c$UniqueIdentifier %>% unique() %>% sample(0.5 * 0.05 * nrow(o2_a1c))
o2_a1c_pdt = o2_a1c %>% filter(UniqueIdentifier %in% o2_a1c_ids) %>% mutate(Marsh = as.factor(Marsh))
ggplot(data=o2_a1c_pdt, mapping=aes(x=std_time, y=A1c, group=UniqueIdentifier, color=Marsh)) +
  geom_line() +
  geom_point() +
  labs(
    title="Spaghetti Plot of A1c Against Time",
```

```
    x="time since pre-measurement (days)",
    y="A1c (percentage)"
  )
```




```
mod_o2_bp = lme(
  fixed = Systolic ~ std_time * Marsh + age + Sex + IncomeLevel + avg.bmi,
  random = ~ std_time | UniqueIdentifier,
  data = o2_bp,
  method = "REML"
)
summary(mod_o2_bp)
```

```
## Linear mixed-effects model fit by REML
##   Data: o2_bp
##        AIC      BIC    logLik
##   81390.64 81476.72 -40683.32
##
## Random effects:
##  Formula: ~std_time | UniqueIdentifier
##  Structure: General positive-definite, Log-Cholesky parametrization
##             StdDev       Corr
## (Intercept)  9.193759759 (Intr)
## std_time     0.009777679 -0.189
## Residual    13.686181446
##
## Fixed effects:  Systolic ~ std_time * Marsh + age + Sex + IncomeLevel + avg.bmi
##                     Value Std.Error   DF  t-value p-value
```

```
## (Intercept)    123.73351 1.3728458 4819 90.12921  0.0000
## std_time        -0.00271 0.0004762 4819 -5.68909  0.0000
## Marsh            0.27156 1.4537770 4815  0.18680  0.8518
## age              0.03676 0.0150363 4815  2.44473  0.0145
## Sex              2.20864 0.3886544 4815  5.68278  0.0000
## IncomeLevel     -0.00013 0.0004892 4815 -0.25924  0.7955
## avg.bmi          0.08346 0.0230930 4815  3.61420  0.0003
## std_time:Marsh  -0.00270 0.0030042 4819 -0.89890  0.3687
##  Correlation:
##                (Intr) std_tm Marsh  age    Sex    IncmLv avg.bm
## std_time       -0.085
## Marsh          -0.047  0.092
## age            -0.805 -0.011 -0.007
## Sex            -0.281  0.002  0.014  0.126
## IncomeLevel    -0.040 -0.002  0.031  0.002 -0.036
## avg.bmi        -0.743 -0.010  0.035  0.264  0.119 -0.016
## std_time:Marsh  0.013 -0.159 -0.561  0.000  0.003  0.000  0.003
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -6.45620460 -0.51551699 -0.04222218  0.43530387 14.47876181
##
## Number of Observations: 9642
## Number of Groups: 4821
```

```
mod_o2_a1c = lme(
  fixed = A1c ~ std_time * Marsh + age + Sex + IncomeLevel + avg.bmi,
  random = ~ std_time | UniqueIdentifier,
  data = o2_a1c,
  method = "REML"
)
summary(mod_o2_a1c)
```

```
## Linear mixed-effects model fit by REML
##   Data: o2_a1c
##        AIC      BIC    logLik
##   19651.47 19729.94 -9813.736
##
## Random effects:
##  Formula: ~std_time | UniqueIdentifier
##  Structure: General positive-definite, Log-Cholesky parametrization
##             StdDev      Corr
## (Intercept) 1.598645983 (Intr)
## std_time    0.001806903 -0.456
## Residual    0.997944892
##
## Fixed effects:  A1c ~ std_time * Marsh + age + Sex + IncomeLevel + avg.bmi
##                    Value  Std.Error   DF  t-value p-value
## (Intercept)     8.350299 0.23551043 2557 35.45618  0.0000
## std_time       -0.000054 0.00007196 2557 -0.74930  0.4537
## Marsh           2.292475 0.13655150 2553 16.78836  0.0000
## age            -0.011512 0.00256433 2553 -4.48932  0.0000
## Sex             0.261052 0.06538829 2553  3.99234  0.0001
## IncomeLevel    -0.000025 0.00004435 2553 -0.57190  0.5674
## avg.bmi        -0.020124 0.00376179 2553 -5.34963  0.0000
```

```
## std_time:Marsh -0.000393 0.00025104 2557 -1.56567  0.1175
##  Correlation:
##                (Intr) std_tm Marsh  age    Sex    IncmLv avg.bm
## std_time       -0.072
## Marsh          -0.186  0.140
## age            -0.796 -0.010  0.059
## Sex            -0.265  0.005  0.068  0.059
## IncomeLevel    -0.028 -0.001  0.029  0.018 -0.013
## avg.bmi        -0.765 -0.006  0.163  0.275  0.175 -0.022
## std_time:Marsh  0.021 -0.287 -0.491  0.001  0.001  0.000  0.002
##
## Standardized Within-Group Residuals:
##        Min         Q1        Med         Q3        Max
## -3.2108163 -0.3304825 -0.1259279  0.1928614  4.9335781
##
## Number of Observations: 5118
## Number of Groups: 2559
```

Tables of Regression Results

```
coef_names = c("std_time:koh_catNone", "std_time:koh_catOne")
col_names = c("lower", "est.", "upper")

obj1_res = data.frame(
  rbind(
    cbind(intervals(mod_o1_a1c)$fixed[coef_names, col_names], summary(mod_o1_a1c)$tTable[coef_names, "p
    cbind(intervals(mod_o1_bp, which="fixed")$fixed[coef_names, col_names], summary(mod_o1_bp)$tTable[c
  )
)

rownames(obj1_res) = c("months:KOHnever", "months:KOHonce", "months:KOHnever ", "months:KOHonce ")
colnames(obj1_res)[4] = "p-value"

kable(obj1_res) %>% pack_rows("a1c_pre_post", 1, 2) %>% pack_rows("sbp_pre_post", 3, 4)
```

	lower	est.	upper	p-value
a1c_pre_post				
months:KOHnever	-0.0042101	-0.0011702	0.0018698	0.4486703
months:KOHonce	-0.0047671	-0.0004953	0.0037766	0.8193793
sbp_pre_post				
months:KOHnever	-0.0417995	-0.0021446	0.0375103	0.9148876
months:KOHonce	-0.0468241	0.0111339	0.0690919	0.7043025

Code Appendix

```
knitr::opts_chunk$set(echo = TRUE)
library(tidyverse)
library(nlme)
library(kableExtra)
# source("primary_obj_data.R")
# source("secondary_obj_data.R")
KOH1 = as.Date("2023-04-05")
o1_bp = read_csv("Analysis Data/Obj1BPPrePost.csv")
names(o1_bp)[1] = "row_id"
o1_a1c = read_csv("Analysis Data/Obj1A1cPrePost.csv")
names(o1_a1c)[1] = "row_id"
o2_bp = read_csv("Analysis Data/Obj2BPPrePost.csv")
names(o2_bp)[1] = "row_id"
o2_a1c = read_csv("Analysis Data/Obj2A1cPrePost.csv")
names(o2_a1c)[1] = "row_id"
DAYS_IN_MONTH = 30.4
# attendees
#   measured before 1st KOH visit: change date to 1st KOH visit
#   measured after 1st KOH visit: keep date
# non-attendees
#   measured before KOH 1: change date to KOH 1
#   measured after KOH 1: DNE
std_times = o1_bp %>%
  group_by(UniqueIdentifier) %>%
  summarize(
    pre_Date = min(BPDate),
    post_Date = max(BPDate),
    std_time = ifelse(BPDate == pre_Date, 0, ifelse(
      KOH == 0,
      as.numeric(post_Date - KOH1),
      ifelse(pre_Date < KOH1, as.numeric(post_Date - KOH1), as.numeric(post_Date - KOHDate))
    ))
  ) %>%
  select(std_time) %>%
  ungroup() %>%
  mutate(
    row_id = as.numeric(o1_bp[["row_id"]]),
    std_months = std_time / DAYS_IN_MONTH
  ) %>%
  select(std_time, row_id)

o1_bp = left_join(o1_bp, std_times, by="row_id")
o1_bp = o1_bp %>% mutate(
  Sex = ifelse(o1_bp$Sex == "M", 1, 0),
  koh_cat = case_when(
    KOH.none == 1 ~ "None",
    KOH.one == 1 ~ "One",
    KOH.mult == 1 ~ "Multiple"
  )
)
```

```
std_times = o1_a1c %>%
  group_by(UniqueIdentifier) %>%
  summarize(
    pre_Date = min(A1cDate),
    post_Date = max(A1cDate),
    std_time = ifelse(A1cDate == pre_Date, 0, ifelse(
      KOH == 0,
      as.numeric(post_Date - KOH1),
      ifelse(pre_Date < KOH1, as.numeric(post_Date - KOH1), as.numeric(post_Date - KOHDate))
    ))
  ) %>%
  select(std_time) %>%
  ungroup() %>%
  mutate(
    row_id = as.numeric(o1_a1c[["row_id"]]),
    std_months = std_time / DAYS_IN_MONTH
  ) %>%
  select(std_time, row_id)

o1_a1c = left_join(o1_a1c, std_times, by="row_id")
o1_a1c = o1_a1c %>% mutate(
  Sex = ifelse(o1_a1c$Sex == "M", 1, 0),
  koh_cat = case_when(
    KOH.none == 1 ~ "None",
    KOH.one == 1 ~ "One",
    KOH.mult == 1 ~ "Multiple"
  )
)
o1_bp_ids = o1_bp$UniqueIdentifier %>% unique() %>% sample(0.5 * 0.25 * nrow(o1_bp))
o1_bp_pdt = o1_bp %>% filter(UniqueIdentifier %in% o1_bp_ids)
ggplot(data=o1_bp_pdt, mapping=aes(x=std_time, y=sys, group=UniqueIdentifier, color=koh_cat)) +
  geom_line() +
  geom_point() +
  labs(
    title="Spaghetti Plot of SBP Against Time",
    x="time since pre-measurement (days)",
    y="SBP (mm/Hg)"
  )

o1_a1c_ids = o1_a1c$UniqueIdentifier %>% unique() %>% sample(0.5 * 0.25 * nrow(o1_a1c))
o1_a1c_pdt = o1_a1c %>% filter(UniqueIdentifier %in% o1_a1c_ids)
ggplot(data=o1_a1c_pdt, mapping=aes(x=std_time, y=A1c, group=UniqueIdentifier, color=koh_cat)) +
  geom_line() +
  geom_point() +
  labs(
    title="Spaghetti Plot of A1c Against Time",
    x="time since pre-measurement (days)",
    y="A1c (percentage)"
  )
mod_o1_bp = lme(
  fixed = sys ~ std_time * koh_cat + age + Sex + IncomeLevel + BLACERISK + avg.bmi,
  random = ~ std_time | UniqueIdentifier,
  data = o1_bp,
```

```
  method = "REML"
)
summary(mod_o1_bp)

mod_o1_a1c = lme(
  fixed = A1c ~ std_time * koh_cat + age + Sex + IncomeLevel + BLACERISK + avg.bmi,
  random = ~ std_time | UniqueIdentifier,
  data = o1_a1c,
  method = "REML"
)
summary(mod_o1_a1c)
# measured before KOH 1: change date to KOH 1
# measured after KOH 1: keep date
std_times = o2_bp %>%
  group_by(UniqueIdentifier) %>%
  summarize(
    pre_Date = min(Date),
    post_Date = max(Date),
    std_time = ifelse(
      Date == pre_Date,
      0,
      # ifelse(pre_Date < KOH1, as.numeric(post_Date - KOH1), as.numeric(post_Date - pre_Date))
      as.numeric(post_Date - pre_Date)
    )
  ) %>%
  select(std_time) %>%
  ungroup() %>%
  mutate(
    row_id = as.numeric(o2_bp[["row_id"]]),
    std_months = std_time / DAYS_IN_MONTH
  ) %>%
  select(std_time, row_id)

o2_bp = left_join(o2_bp, std_times, by="row_id")
o2_bp = o2_bp %>% mutate(Sex = ifelse(o2_bp$Sex == "M", 1, 0))

std_times = o2_a1c %>%
  group_by(UniqueIdentifier) %>%
  summarize(
    pre_Date = min(Date),
    post_Date = max(Date),
    std_time = ifelse(
      Date == pre_Date,
      0,
      # ifelse(pre_Date < KOH1, as.numeric(post_Date - KOH1), as.numeric(post_Date - pre_Date))
      as.numeric(post_Date - KOH1)
    )
  ) %>%
  select(std_time) %>%
  ungroup() %>%
  mutate(
    row_id = as.numeric(o2_a1c[["row_id"]]),
    std_months = std_time / DAYS_IN_MONTH
```

```
  ) %>%
  select(std_time, row_id)

o2_a1c = left_join(o2_a1c, std_times, by="row_id")
o2_a1c = o2_a1c %>% mutate(Sex = ifelse(o2_a1c$Sex == "M", 1, 0))
o2_bp_ids = o2_bp$UniqueIdentifier %>% unique() %>% sample(0.5 * 0.05 * nrow(o2_bp))
o2_bp_pdt = o2_bp %>% filter(UniqueIdentifier %in% o2_bp_ids) %>% mutate(Marsh = as.factor(Marsh))
ggplot(data=o2_bp_pdt, mapping=aes(x=std_time, y=Systolic, group=UniqueIdentifier, color=Marsh)) +
  geom_line() +
  geom_point() +
  labs(
    title="Spaghetti Plot of SBP Against Time",
    x="time since pre-measurement (days)",
    y="SBP (mm/Hg)"
  )

o2_a1c_ids = o2_a1c$UniqueIdentifier %>% unique() %>% sample(0.5 * 0.05 * nrow(o2_a1c))
o2_a1c_pdt = o2_a1c %>% filter(UniqueIdentifier %in% o2_a1c_ids) %>% mutate(Marsh = as.factor(Marsh))
ggplot(data=o2_a1c_pdt, mapping=aes(x=std_time, y=A1c, group=UniqueIdentifier, color=Marsh)) +
  geom_line() +
  geom_point() +
  labs(
    title="Spaghetti Plot of A1c Against Time",
    x="time since pre-measurement (days)",
    y="A1c (percentage)"
  )
mod_o2_bp = lme(
  fixed = Systolic ~ std_time * Marsh + age + Sex + IncomeLevel + avg.bmi,
  random = ~ std_time | UniqueIdentifier,
  data = o2_bp,
  method = "REML"
)
summary(mod_o2_bp)

mod_o2_a1c = lme(
  fixed = A1c ~ std_time * Marsh + age + Sex + IncomeLevel + avg.bmi,
  random = ~ std_time | UniqueIdentifier,
  data = o2_a1c,
  method = "REML"
)
summary(mod_o2_a1c)
coef_names = c("std_time:koh_catNone", "std_time:koh_catOne")
col_names = c("lower", "est.", "upper")

obj1_res = data.frame(
  rbind(
    cbind(intervals(mod_o1_a1c)$fixed[coef_names, col_names], summary(mod_o1_a1c)$tTable[coef_names, "p
    cbind(intervals(mod_o1_bp, which="fixed")$fixed[coef_names, col_names], summary(mod_o1_bp)$tTable[c
  )
)

rownames(obj1_res) = c("months:KOHnever", "months:KOHonce", "months:KOHnever ", "months:KOHonce ")
colnames(obj1_res)[4] = "p-value"
```

```
kable(obj1_res) %>% pack_rows("a1c_pre_post", 1, 2) %>% pack_rows("sbp_pre_post", 3, 4)
```