Baseline analysis supplement

Welcome to the LMI baseline supplement! This supplement will walk you through the paper and provides additional graphs.

# Measures used in the study

Clicking the “Code”-buttons on the right shows code for each chunk.

To start data analysis, we set up the basics to enable compiling the document:

source("baseline-datasetup.R")

# Means with CIs taking clustering into account

## Description

The code chunks create linear models for all variables. The results are used to estimate confidence intervals and intra-class correlations (ICC).

Browse throught the tabs above to see different information.

Code chunk below shows how the values are calculated.

# Create a vector with all names of the variables we want. Exclude T3 variables.  
names <- df %>% select(-id, -intervention, -group, -school, -girl, -track, -trackSchool, -contains("\_T3"), -contains("\_diff")) %>% names(.)  
  
# Create empty soon-to-be-filled objects  
m <- NA  
mean <- NA  
m\_p <- NA  
ci\_low <- NA  
ci\_high <- NA  
ICC\_group <- NA  
ICC\_School <- NA  
nonmissings <- NA  
  
## Use this to test a single variable:  
# dftest <- df #%>% na.omit()  
# m <- lme4::lmer(sitLieAccelerometer\_diff ~ (1|school) + (1|group), data=dftest)  
# mean <- lme4::fixef(m)  
# m\_p <- profile(m, which = "beta\_")  
# ci\_low <- confint(m\_p)[, 1]  
# ci\_high <- confint(m\_p)[, 2]  
# ICC\_group <- icc(m)[1]  
# ICC\_School <- icc(m)[2]  
# nonmissings <- length(m@resp$y)  
  
  
# Loop over each variable name, extract statistics:   
  
# all participants  
for (i in names){  
m <- lme4::lmer(paste0(i," ~ (1|school) + (1|group)"), data=df)  
mean[i] <- lme4::fixef(m)  
m\_p <- profile(m, which = "beta\_")  
ci\_low[i] <- confint(m\_p)[, 1]  
ci\_high[i] <- confint(m\_p)[, 2]  
ICC\_group[i] <- sjstats::icc(m)[1]  
ICC\_School[i] <- sjstats::icc(m)[2]  
nonmissings[i] <- length(m@resp$y)  
}  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m, which = "beta\_"): non-monotonic profile for  
## (Intercept)  
## Warning in confint.thpr(m\_p): bad spline fit for (Intercept): falling back  
## to linear interpolation  
  
## Warning in confint.thpr(m\_p): bad spline fit for (Intercept): falling back  
## to linear interpolation  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m, which = "beta\_"): non-monotonic profile for  
## (Intercept)  
## Warning in confint.thpr(m\_p): bad spline fit for (Intercept): falling back  
## to linear interpolation  
  
## Warning in confint.thpr(m\_p): bad spline fit for (Intercept): falling back  
## to linear interpolation  
  
  
vardatatable <- data\_frame(  
 Variable = stringr::str\_sub(names(mean), end = -4), #remove "\_T1" from names  
 "Mean" = round(mean, 1),  
 "CI95" = paste(round(ci\_low, 1), " - ", round(ci\_high, 2)),   
 "ICC class" =ifelse(round(ICC\_group, 3) == 0, "< 0.001", round(ICC\_group, 3)),   
 "ICC school" = ifelse(round(ICC\_School, 3) == 0, "< 0.001", round(ICC\_School, 3)),  
 n = nonmissings) %>%   
 dplyr::arrange(Variable) %>%   
 filter(Variable != "") %>% #remove first row, which is empty  
 mutate(`ICC class` = as.numeric(`ICC class`)) %>%   
 as.data.frame()  
## Warning in evalq(as.numeric(`ICC class`), <environment>): NAs introduced by  
## coercion  
  
ci\_total <- data\_frame(  
 "ciLo" = ci\_low,  
 "mean" = mean,  
 "ciHi" = ci\_high,   
 "diamondlabels" = names(mean)) %>%   
 filter(diamondlabels != "") %>% #remove first row, which is empty  
 as.data.frame()

## Primary outcome variables

options(tibble.print\_max = 1000)  
  
vardatatable2 <- vardatatable  
vardatatable2$Variable <- gsub("paAccelerometer", "PA minutes", vardatatable2$Variable)  
vardatatable2$Variable <- gsub("padaysLastweek", "Number of MVPA days last week (self-report)", vardatatable2$Variable)  
vardatatable2$Variable <- gsub("sitLieAccelerometer", "Minutes spent sitting or lying down", vardatatable2$Variable)  
vardatatable2$Variable <- gsub("sitBreaks", "Number of times sitting was interrupted", vardatatable2$Variable)  
  
vardatatable2 %>%   
 filter(Variable == "MVPA minutes" |   
 Variable == "Number of MVPA days last week (self-report)" |   
 Variable == "Daily minutes spent sitting or lying down" |   
 Variable == "Number of times sitting was interrupted daily" ) %>%   
 papaja::apa\_table(  
 caption = "Primary outcome variables with their class and school ICCs",  
 escape = T,  
 format.args = list(digits = c(1, 3, 0), margin = 2))

(#tab:selected-total)

*Primary outcome variables with their class and school ICCs*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Variable | Mean | CI95 | ICC class | ICC school | n |
| Number of MVPA days last week (self-report) | 2.8 | 2.6  -  2.95 | 0.047 | < 0.001 | 1,082 |

## All main variables, their means, CIs and ICCs.

options(tibble.print\_max = 1000)  
   
vardatatable %>%  
 papaja::apa\_table(  
 caption = "All variables with their class and school ICCs",  
 escape = T,  
 format.args = list(digits = c(1, 3, 0), margin = 2))

(#tab:allvars-total)

*All variables with their class and school ICCs*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Variable | Mean | CI95 | ICC class | ICC school | n |
| b5agr | 5.3 | 5.2  -  5.5 | 0.033 | 0.012 | 1,068 |
| b5agr\_01 | 5.5 | 5.4  -  5.66 | 0.045 | 0.002 | 1,061 |
| b5agrReverseCoded\_02 | 5.2 | 5  -  5.37 | 0.005 | 0.01 | 1,064 |
| b5cons | 5.0 | 4.9  -  5.17 | NA | 0.007 | 1,067 |
| b5cons\_01 | 5.4 | 5.2  -  5.65 | 0.014 | 0.015 | 1,059 |
| b5consReverseCoded\_02 | 4.6 | 4.5  -  4.78 | NA | 0.002 | 1,065 |
| b5ext | 4.5 | 4.3  -  4.77 | 0.020 | 0.022 | 1,068 |
| b5ext\_01 | 5.0 | 4.7  -  5.24 | 0.048 | 0.013 | 1,063 |
| b5extReverseCoded\_02 | 4.1 | 3.8  -  4.32 | 0.002 | 0.012 | 1,062 |
| b5neur | 3.8 | 3.6  -  4 | 0.014 | 0.012 | 1,068 |
| b5neur\_01 | 4.1 | 3.8  -  4.39 | 0.017 | 0.014 | 1,064 |
| b5neurReverseCoded\_02 | 3.5 | 3.4  -  3.67 | 0.011 | 0.002 | 1,061 |
| b5open | 4.9 | 4.7  -  5.12 | 0.010 | 0.03 | 1,068 |
| b5open\_01 | 5.4 | 5.1  -  5.59 | 0.025 | 0.016 | 1,062 |
| b5openReverseCoded\_02 | 4.4 | 4.2  -  4.67 | NA | 0.016 | 1,066 |
| fatpct | 25.5 | 22.4  -  28.67 | 0.003 | 0.123 | 942 |
| PA\_actCop | 2.6 | 2.6  -  2.69 | 0.035 | < 0.001 | 1,073 |
| PA\_actionplan | 2.8 | 2.7  -  2.83 | 0.038 | < 0.001 | 1,073 |
| PA\_actplan\_01 | 2.6 | 2.5  -  2.68 | 0.032 | < 0.001 | 1,073 |
| PA\_actplan\_02 | 2.8 | 2.7  -  2.88 | 0.028 | < 0.001 | 1,073 |
| PA\_actplan\_03 | 2.9 | 2.8  -  2.93 | 0.037 | < 0.001 | 1,073 |
| PA\_actplan\_04 | 2.8 | 2.7  -  2.86 | 0.038 | < 0.001 | 1,073 |
| PA\_agrbct | 3.1 | 3  -  3.23 | 0.052 | < 0.001 | 1,071 |
| PA\_agrbct\_01 | 3.7 | 3.5  -  3.85 | 0.061 | < 0.001 | 1,071 |
| PA\_agrbct\_02 | 3.2 | 3  -  3.33 | 0.025 | < 0.001 | 1,071 |
| PA\_agrbct\_03 | 2.4 | 2.3  -  2.57 | 0.009 | < 0.001 | 1,071 |
| PA\_agrbct\_04 | 2.5 | 2.4  -  2.67 | 0.035 | < 0.001 | 1,071 |
| PA\_agrbct\_05 | 2.6 | 2.5  -  2.78 | 0.055 | < 0.001 | 1,071 |
| PA\_agrbct\_06 | 3.2 | 3.1  -  3.38 | 0.028 | 0.001 | 1,071 |
| PA\_agrbct\_07 | 3.0 | 2.9  -  3.16 | 0.022 | < 0.001 | 1,071 |
| PA\_agrbct\_08 | 3.1 | 3  -  3.25 | 0.023 | < 0.001 | 1,071 |
| PA\_agrbct\_09 | 3.7 | 3.5  -  3.85 | 0.032 | 0.001 | 1,071 |
| PA\_agrbct\_10 | 3.6 | 3.5  -  3.79 | 0.044 | < 0.001 | 1,071 |
| PA\_amotivation | 1.5 | 1.5  -  1.61 | 0.035 | 0.002 | 1,072 |
| PA\_amotivation\_01 | 1.5 | 1.4  -  1.59 | 0.031 | < 0.001 | 1,062 |
| PA\_amotivation\_02 | 1.5 | 1.4  -  1.63 | 0.026 | 0.005 | 1,057 |
| PA\_amotivation\_03 | 1.6 | 1.5  -  1.69 | 0.032 | < 0.001 | 1,067 |
| PA\_amotivation\_04 | 1.5 | 1.4  -  1.55 | 0.027 | < 0.001 | 1,066 |
| PA\_autonomous | 3.4 | 3.3  -  3.52 | 0.073 | < 0.001 | 1,078 |
| PA\_autonomous\_01 | 3.1 | 3  -  3.29 | 0.002 | 0.012 | 1,063 |
| PA\_autonomous\_02 | 3.8 | 3.7  -  3.86 | 0.031 | < 0.001 | 1,063 |
| PA\_autonomous\_03 | 3.4 | 3.2  -  3.51 | 0.058 | < 0.001 | 1,070 |
| PA\_autonomous\_04 | 3.3 | 3.2  -  3.47 | 0.059 | 0.004 | 1,070 |
| PA\_autonomous\_05 | 3.3 | 3.1  -  3.39 | 0.048 | < 0.001 | 1,062 |
| PA\_autonomous\_06 | 3.1 | 3  -  3.22 | 0.072 | < 0.001 | 1,066 |
| PA\_autonomous\_07 | 3.7 | 3.6  -  3.88 | 0.072 | 0.005 | 1,070 |
| PA\_autonomous\_08 | 3.6 | 3.5  -  3.76 | 0.057 | < 0.001 | 1,066 |
| PA\_autonomous\_09 | 3.5 | 3.4  -  3.62 | 0.055 | < 0.001 | 1,057 |
| PA\_controlled | 1.8 | 1.8  -  1.9 | 0.005 | < 0.001 | 1,073 |
| PA\_controlled\_01 | 1.7 | 1.6  -  1.73 | 0.010 | < 0.001 | 1,068 |
| PA\_controlled\_02 | 1.5 | 1.4  -  1.55 | 0.024 | < 0.001 | 1,069 |
| PA\_controlled\_03 | 1.5 | 1.5  -  1.61 | 0.011 | < 0.001 | 1,064 |
| PA\_controlled\_04 | 2.2 | 2.1  -  2.28 | NA | 0.001 | 1,068 |
| PA\_controlled\_05 | 2.3 | 2.2  -  2.4 | 0.006 | < 0.001 | 1,066 |
| PA\_copingplan | 2.5 | 2.4  -  2.56 | 0.023 | < 0.001 | 1,073 |
| PA\_copplan\_01 | 2.4 | 2.3  -  2.47 | 0.007 | < 0.001 | 1,073 |
| PA\_copplan\_02 | 2.4 | 2.4  -  2.5 | 0.004 | < 0.001 | 1,073 |
| PA\_copplan\_03 | 2.5 | 2.4  -  2.54 | 0.035 | < 0.001 | 1,073 |
| PA\_copplan\_04 | 2.7 | 2.6  -  2.77 | 0.035 | < 0.001 | 1,073 |
| PA\_dnorm | 4.4 | 4.2  -  4.58 | 0.050 | 0.003 | 1,073 |
| PA\_dnorm\_01 | 4.5 | 4.3  -  4.71 | 0.050 | 0.004 | 1,073 |
| PA\_dnorm\_02 | 4.3 | 4.1  -  4.52 | 0.027 | 0.004 | 1,073 |
| PA\_extrinsic | 1.6 | 1.5  -  1.62 | 0.022 | < 0.001 | 1,069 |
| PA\_extrinsic\_01 | 1.7 | 1.6  -  1.73 | 0.010 | < 0.001 | 1,068 |
| PA\_extrinsic\_02 | 1.5 | 1.4  -  1.55 | 0.024 | < 0.001 | 1,069 |
| PA\_extrinsic\_03 | 1.5 | 1.5  -  1.61 | 0.011 | < 0.001 | 1,064 |
| PA\_frqbct | 2.5 | 2.4  -  2.62 | 0.027 | < 0.001 | 1,070 |
| PA\_frqbct\_01 | 3.7 | 3.6  -  3.84 | 0.014 | < 0.001 | 1,070 |
| PA\_frqbct\_02 | 1.9 | 1.8  -  1.98 | 0.024 | < 0.001 | 1,070 |
| PA\_frqbct\_03 | 2.0 | 1.9  -  2.13 | 0.007 | < 0.001 | 1,070 |
| PA\_frqbct\_04 | 1.9 | 1.8  -  2.02 | 0.035 | < 0.001 | 1,070 |
| PA\_frqbct\_05 | 2.5 | 2.4  -  2.64 | 0.033 | < 0.001 | 1,070 |
| PA\_frqbct\_06 | 3.4 | 3.3  -  3.54 | 0.016 | < 0.001 | 1,070 |
| PA\_frqbct\_07 | 2.2 | 2.1  -  2.3 | 0.003 | < 0.001 | 1,070 |
| PA\_frqbct\_08 | 2.4 | 2.3  -  2.5 | 0.014 | < 0.001 | 1,070 |
| PA\_frqbct\_09 | 2.8 | 2.7  -  2.9 | 0.007 | < 0.001 | 1,070 |
| PA\_goal | 0.6 | 0.5  -  0.59 | 0.046 | < 0.001 | 1,061 |
| PA\_goal\_01 | 0.6 | 0.5  -  0.59 | 0.046 | < 0.001 | 1,061 |
| PA\_identified | 3.4 | 3.3  -  3.54 | 0.040 | 0.004 | 1,074 |
| PA\_identified\_01 | 3.1 | 3  -  3.29 | 0.002 | 0.012 | 1,063 |
| PA\_identified\_02 | 3.8 | 3.7  -  3.86 | 0.031 | < 0.001 | 1,063 |
| PA\_identified\_03 | 3.4 | 3.2  -  3.51 | 0.058 | < 0.001 | 1,070 |
| PA\_inorm | 4.7 | 4.5  -  4.84 | 0.006 | 0.003 | 1,073 |
| PA\_inorm\_01 | 4.7 | 4.5  -  4.84 | 0.006 | 0.003 | 1,073 |
| PA\_integrated | 3.2 | 3.1  -  3.34 | 0.071 | < 0.001 | 1,073 |
| PA\_integrated\_01 | 3.3 | 3.2  -  3.47 | 0.059 | 0.004 | 1,070 |
| PA\_integrated\_02 | 3.3 | 3.1  -  3.39 | 0.048 | < 0.001 | 1,062 |
| PA\_integrated\_03 | 3.1 | 3  -  3.22 | 0.072 | < 0.001 | 1,066 |
| PA\_intention | 5.4 | 5.2  -  5.57 | 0.105 | < 0.001 | 1,073 |
| PA\_intention\_01 | 5.4 | 5.2  -  5.55 | 0.088 | < 0.001 | 1,073 |
| PA\_intention\_02 | 5.4 | 5.2  -  5.61 | 0.111 | < 0.001 | 1,073 |
| PA\_intrinsic | 3.6 | 3.5  -  3.74 | 0.074 | < 0.001 | 1,074 |
| PA\_intrinsic\_01 | 3.7 | 3.6  -  3.88 | 0.072 | 0.005 | 1,070 |
| PA\_intrinsic\_02 | 3.6 | 3.5  -  3.76 | 0.057 | < 0.001 | 1,066 |
| PA\_intrinsic\_03 | 3.5 | 3.4  -  3.62 | 0.055 | < 0.001 | 1,057 |
| PA\_introjected | 2.2 | 2.2  -  2.33 | NA | < 0.001 | 1,073 |
| PA\_introjected\_01 | 2.2 | 2.1  -  2.28 | NA | 0.001 | 1,068 |
| PA\_introjected\_02 | 2.3 | 2.2  -  2.4 | 0.006 | < 0.001 | 1,066 |
| PA\_opportunities | 5.1 | 5.1  -  5.19 | 0.018 | < 0.001 | 1,075 |
| PA\_opportunities\_01 | 4.7 | 4.3  -  4.98 | 0.037 | 0.02 | 1,070 |
| PA\_opportunities\_02 | 5.5 | 5.4  -  5.62 | 0.018 | < 0.001 | 1,067 |
| PA\_opportunities\_04 | 5.6 | 5.4  -  5.75 | 0.021 | 0.008 | 1,070 |
| PA\_opportunities\_05 | 5.3 | 5.2  -  5.43 | NA | < 0.001 | 1,064 |
| PA\_opportunities\_07 | 4.6 | 4.5  -  4.77 | 0.009 | 0.001 | 1,068 |
| PA\_opportunitiesReverseCoded\_03 | 5.1 | 5  -  5.23 | NA | < 0.001 | 1,065 |
| PA\_opportunitiesReverseCoded\_06 | 3.6 | 3.3  -  3.95 | NA | 0.024 | 1,059 |
| PA\_opportunitiesReverseCoded\_08 | 6.6 | 6.5  -  6.68 | 0.020 | < 0.001 | 1,066 |
| PA\_outcomeExpectations | 4.6 | 4.5  -  4.74 | 0.034 | 0.013 | 1,078 |
| PA\_outcomeExpectations\_01 | 5.5 | 5.3  -  5.7 | 0.042 | 0.011 | 1,071 |
| PA\_outcomeExpectations\_03 | 5.2 | 4.9  -  5.47 | 0.008 | 0.018 | 1,064 |
| PA\_outcomeExpectations\_04 | 5.5 | 5.3  -  5.65 | 0.009 | 0.007 | 1,071 |
| PA\_outcomeExpectations\_05 | 5.2 | 5  -  5.39 | 0.023 | 0.011 | 1,067 |
| PA\_outcomeExpectations\_06 | 5.4 | 5.2  -  5.63 | 0.030 | 0.016 | 1,071 |
| PA\_outcomeExpectations\_07 | 4.9 | 4.7  -  5.13 | 0.039 | 0.008 | 1,062 |
| PA\_outcomeExpectations\_10 | 5.4 | 5.2  -  5.54 | 0.019 | 0.006 | 1,069 |
| PA\_outcomeExpectations\_11 | 5.1 | 4.9  -  5.38 | 0.055 | 0.011 | 1,063 |
| PA\_outcomeExpectations\_12 | 5.2 | 4.9  -  5.45 | 0.042 | 0.014 | 1,070 |
| PA\_outcomeExpectationsNegative\_02 | 2.9 | 2.7  -  3.02 | 0.046 | < 0.001 | 1,066 |
| PA\_outcomeExpectationsNegative\_08 | 2.5 | 2.3  -  2.69 | 0.038 | 0.004 | 1,063 |
| PA\_outcomeExpectationsNegative\_09 | 2.6 | 2.5  -  2.68 | 0.023 | < 0.001 | 1,064 |
| PA\_pbc | 5.3 | 5.2  -  5.45 | 0.028 | 0.004 | 1,071 |
| PA\_pbc\_01 | 5.9 | 5.7  -  5.98 | 0.032 | < 0.001 | 1,069 |
| PA\_pbc\_03 | 5.2 | 5  -  5.4 | 0.014 | 0.01 | 1,066 |
| PA\_pbcReverseCoded\_02 | 4.9 | 4.7  -  5.11 | 0.005 | 0.005 | 1,063 |
| PA\_selfefficacy | 5.2 | 5.1  -  5.35 | 0.007 | 0.005 | 1,071 |
| PA\_selfefficacy\_01 | 5.9 | 5.8  -  5.96 | 0.009 | < 0.001 | 1,071 |
| PA\_selfefficacyReverseCoded\_02 | 4.6 | 4.4  -  4.77 | 0.002 | 0.005 | 1,071 |
| PA\_sePbc | 5.3 | 5.1  -  5.4 | 0.028 | 0.007 | 1,071 |
| paAccelerometer | 183.1 | 166  -  200.45 | 0.072 | 0.096 | 706 |
| padaysLastweek | 2.8 | 2.6  -  2.95 | 0.047 | < 0.001 | 1,082 |
| pafreqUsually | 2.6 | 2.4  -  2.73 | 0.094 | 0.003 | 1,082 |
| pahrsLastweek | 5.4 | 4.7  -  6.16 | 0.024 | 0.007 | 1,082 |
| pahrsUsually | 3.1 | 2.9  -  3.36 | 0.085 | 0.005 | 1,082 |
| paLastweek | 339.7 | 296  -  382.07 | 0.024 | 0.006 | 1,082 |
| paminLastweek | 1.4 | 1.4  -  1.5 | NA | 0.01 | 1,082 |
| SB\_dnorm | 3.2 | 3.1  -  3.41 | 0.003 | 0.007 | 1,062 |
| SB\_dnorm\_01 | 3.1 | 2.9  -  3.22 | NA | 0.004 | 1,059 |
| SB\_dnorm\_02 | 3.4 | 3.2  -  3.59 | 0.010 | 0.007 | 1,060 |
| SB\_inorm | 4.0 | 3.9  -  4.14 | 0.012 | 0.004 | 1,064 |
| SB\_inorm\_01 | 3.9 | 3.8  -  4.11 | NA | 0.008 | 1,056 |
| SB\_inorm\_02 | 4.1 | 3.9  -  4.28 | 0.030 | 0.005 | 1,053 |
| SB\_intention | 3.7 | 3.4  -  4.06 | 0.014 | 0.035 | 1,064 |
| SB\_intention\_01 | 3.3 | 3  -  3.64 | NA | 0.025 | 1,059 |
| SB\_intention\_02 | 3.5 | 3.2  -  3.81 | NA | 0.034 | 1,056 |
| SB\_intention\_03 | 4.0 | 3.7  -  4.37 | 0.023 | 0.027 | 1,054 |
| SB\_intention\_04 | 4.1 | 3.8  -  4.43 | 0.028 | 0.027 | 1,055 |
| SB\_outcomeExpectations | 4.4 | 4.3  -  4.53 | 0.029 | 0.005 | 1,064 |
| SB\_outcomeExpectations\_02 | 4.8 | 4.6  -  5.06 | 0.017 | 0.014 | 1,056 |
| SB\_outcomeExpectations\_03 | 4.6 | 4.4  -  4.88 | 0.024 | 0.014 | 1,058 |
| SB\_outcomeExpectations\_04 | 4.9 | 4.7  -  5.12 | 0.038 | 0.005 | 1,063 |
| SB\_outcomeExpectations\_05 | 4.9 | 4.7  -  5.09 | 0.040 | 0.004 | 1,060 |
| SB\_outcomeExpectations\_06 | 4.9 | 4.7  -  5.11 | 0.047 | 0.011 | 1,058 |
| SB\_outcomeExpectationsNegative\_01 | 3.4 | 3.3  -  3.51 | 0.008 | < 0.001 | 1,063 |
| SB\_outcomeExpectationsNegative\_07 | 3.3 | 3.2  -  3.51 | 0.012 | 0.003 | 1,057 |
| SB\_sePbc | 5.0 | 4.8  -  5.14 | 0.015 | 0.011 | 1,064 |
| SB\_sePbc\_01 | 4.4 | 4.1  -  4.6 | 0.010 | 0.014 | 1,062 |
| SB\_sePbc\_02 | 4.3 | 4.1  -  4.54 | 0.003 | 0.015 | 1,061 |
| SB\_sePbc\_03 | 5.7 | 5.6  -  5.87 | 0.026 | < 0.001 | 1,060 |
| SB\_sePbc\_04 | 5.7 | 5.6  -  5.88 | 0.034 | < 0.001 | 1,060 |
| SB\_sePbc\_05 | 4.7 | 4.5  -  4.91 | 0.005 | 0.012 | 1,054 |
| sitBreaks | 28.0 | 24.7  -  31.37 | 0.058 | 0.085 | 706 |
| sitLieAccelerometer | 573.9 | 535.5  -  611.8 | 0.086 | 0.145 | 706 |
| symptom | 2.0 | 1.8  -  2.11 | 0.023 | 0.039 | 1,084 |
| symptom\_headAche | 2.0 | 1.8  -  2.16 | 0.007 | 0.027 | 1,055 |
| symptom\_irritabilityAngerbursts | 1.9 | 1.8  -  2.04 | 0.026 | 0.01 | 1,039 |
| symptom\_lowerBackPain | 1.8 | 1.7  -  1.85 | 0.004 | 0.005 | 1,053 |
| symptom\_neckShoulderPain | 2.0 | 1.8  -  2.17 | 0.011 | 0.033 | 1,047 |
| symptom\_sleepDifficulty | 2.1 | 1.9  -  2.27 | 0.010 | 0.023 | 1,052 |
| symptom\_stomachAche | 1.7 | 1.6  -  1.86 | 0.015 | 0.023 | 1,026 |
| symptom\_tensionNervousness | 1.9 | 1.8  -  2.08 | 0.007 | 0.024 | 1,051 |
| symptom\_tirednessFaintness | 2.3 | 2.2  -  2.4 | 0.025 | 0.009 | 1,058 |

## ICC top 20 items, sorted by classroom ICC

vardatatable %>%   
 select(Variable, `ICC class`, `ICC school`, n) %>%   
 arrange(desc(`ICC class`)) %>% slice(1:20) %>%   
 as.tbl() %>%   
 papaja::apa\_table(caption = "Intra-class correlations sorted by classroom ICC",  
 format.args = list(digits = c(3, 0), margin = 2))

(#tab:icc-school-class-sortbyclass)

*Intra-class correlations sorted by classroom ICC*

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | ICC class | ICC school | n |
| PA\_intention\_02 | 0.111 | < 0.001 | 1,073 |
| PA\_intention | 0.105 | < 0.001 | 1,073 |
| pafreqUsually | 0.094 | 0.003 | 1,082 |
| PA\_intention\_01 | 0.088 | < 0.001 | 1,073 |
| sitLieAccelerometer | 0.086 | 0.145 | 706 |
| pahrsUsually | 0.085 | 0.005 | 1,082 |
| PA\_intrinsic | 0.074 | < 0.001 | 1,074 |
| PA\_autonomous | 0.073 | < 0.001 | 1,078 |
| PA\_autonomous\_06 | 0.072 | < 0.001 | 1,066 |
| PA\_autonomous\_07 | 0.072 | 0.005 | 1,070 |
| PA\_integrated\_03 | 0.072 | < 0.001 | 1,066 |
| PA\_intrinsic\_01 | 0.072 | 0.005 | 1,070 |
| paAccelerometer | 0.072 | 0.096 | 706 |
| PA\_integrated | 0.071 | < 0.001 | 1,073 |
| PA\_agrbct\_01 | 0.061 | < 0.001 | 1,071 |
| PA\_autonomous\_04 | 0.059 | 0.004 | 1,070 |
| PA\_integrated\_01 | 0.059 | 0.004 | 1,070 |
| PA\_autonomous\_03 | 0.058 | < 0.001 | 1,070 |
| PA\_identified\_03 | 0.058 | < 0.001 | 1,070 |
| sitBreaks | 0.058 | 0.085 | 706 |

## ICC top 20 items, sorted by school ICC

vardatatable %>% select(Variable, `ICC class`, `ICC school`, n) %>%   
 arrange(desc(`ICC school`)) %>%   
 slice(1:20) %>%   
 papaja::apa\_table(caption = "Intra-class correlations sorted by school ICC",  
 format.args = list(digits = c(3, 0), margin = 2))

(#tab:icc-school-class-sortbyschool)

*Intra-class correlations sorted by school ICC*

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | ICC class | ICC school | n |
| sitLieAccelerometer | 0.086 | 0.145 | 706 |
| fatpct | 0.003 | 0.123 | 942 |
| paAccelerometer | 0.072 | 0.096 | 706 |
| sitBreaks | 0.058 | 0.085 | 706 |
| symptom | 0.023 | 0.039 | 1,084 |
| SB\_intention | 0.014 | 0.035 | 1,064 |
| SB\_intention\_02 | NA | 0.034 | 1,056 |
| symptom\_neckShoulderPain | 0.011 | 0.033 | 1,047 |
| b5open | 0.010 | 0.03 | 1,068 |
| SB\_intention\_03 | 0.023 | 0.027 | 1,054 |
| SB\_intention\_04 | 0.028 | 0.027 | 1,055 |
| symptom\_headAche | 0.007 | 0.027 | 1,055 |
| SB\_intention\_01 | NA | 0.025 | 1,059 |
| PA\_opportunitiesReverseCoded\_06 | NA | 0.024 | 1,059 |
| symptom\_tensionNervousness | 0.007 | 0.024 | 1,051 |
| symptom\_sleepDifficulty | 0.010 | 0.023 | 1,052 |
| symptom\_stomachAche | 0.015 | 0.023 | 1,026 |
| b5ext | 0.020 | 0.022 | 1,068 |
| PA\_opportunities\_01 | 0.037 | 0.02 | 1,070 |
| PA\_outcomeExpectations\_03 | 0.008 | 0.018 | 1,064 |

## Model with educational track

The model below is the same as the one above, except it adds educational track.

As before, we create a table with all variables, their means, CIs and ICCs.

#library(sjstats)  
#library(lme4)  
  
names <- df %>% select(-id, -intervention, -group, -school, -girl, -track, -trackSchool, -contains("\_T3"), -contains("\_diff")) %>% names(.)  
  
# Create empty soon-to-be-filled objects  
m <- NA  
mean <- NA  
m\_p <- NA  
ci\_low <- NA  
ci\_high <- NA  
ICC\_group <- NA  
ICC\_School <- NA  
nonmissings <- NA  
ICC\_track <- NA  
  
  
## To test the code with a single variable:  
# dftest <- df #%>% na.omit()  
# m <- lmer(sitLieAccelerometer\_T1 ~ (1|school) + (1|group) + (1|track), data=df)  
# mean <- fixef(m)  
# m\_p <- profile(m)  
# ci\_low <- confint(m\_p)[5]  
# ci\_high <- confint(m\_p)[10]  
# ICC\_group <- icc(m)[1]  
# ICC\_School <- icc(m)[2]  
# nonmissings <- length(m@resp$y)  
  
# Loop over each variable name, extract statistics:   
  
# all participants   
for (i in names){  
m <- lme4::lmer(paste0(i," ~ (1|school) + (1|group) + (1|track)"), data=df)  
mean[i] <- lme4::fixef(m)  
m\_p <- profile(m, which = "beta\_")  
ci\_low[i] <- confint(m\_p)[5]  
ci\_high[i] <- confint(m\_p)[10]  
ICC\_group[i] <- sjstats::icc(m)[1]  
ICC\_School[i] <- sjstats::icc(m)[3]  
ICC\_track[i] <- sjstats::icc(m)[2]  
nonmissings[i] <- length(m@resp$y)  
}  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in profile.merMod(m, which = "beta\_"): non-monotonic profile for  
## (Intercept)  
## Warning in confint.thpr(m\_p): bad spline fit for (Intercept): falling back  
## to linear interpolation  
  
## Warning in confint.thpr(m\_p): bad spline fit for (Intercept): falling back  
## to linear interpolation  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m, which = "beta\_"): non-monotonic profile for  
## (Intercept)  
## Warning in confint.thpr(m\_p): bad spline fit for (Intercept): falling back  
## to linear interpolation  
  
## Warning in confint.thpr(m\_p): bad spline fit for (Intercept): falling back  
## to linear interpolation  
  
vardatatable\_containing\_edutrack <- data\_frame(  
 Variable = stringr::str\_sub(names(mean), end = -4), #remove "\_T1" from names  
 Mean = round(mean, 1),   
 CI95 = paste(round(ci\_low, 2), "-", round(ci\_high, 2)),   
 "ICC class" =ifelse(round(ICC\_group, 3) == 0, "< 0.001", round(ICC\_group, 3)),   
 "ICC school" = ifelse(round(ICC\_School, 3) == 0, "< 0.001", round(ICC\_School, 3)),  
 "ICC educational track" = ifelse(round(ICC\_track, 3) == 0, "< 0.001", round(ICC\_track, 3)),  
 n = nonmissings) %>%   
 dplyr::arrange(Variable) %>%   
 filter(Variable != "") #remove first row, which is empty  
  
options(tibble.print\_max = 15)

### Top 20 items, sorted by school

vardatatable\_containing\_edutrack %>%   
 select(Variable, `ICC educational track`, `ICC class`, `ICC school`, n) %>%   
 arrange(desc(`ICC school`)) %>%   
 top\_n(20) %>%   
 papaja::apa\_table(caption = "Intra-class correlations sorted by school ICC",  
 format.args = list(digits = c(0), margin = 2))  
## Selecting by n

(#tab:icc-school-class-track-sortbyschool)

*Intra-class correlations sorted by school ICC*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Variable | ICC educational track | ICC class | ICC school | n |
| PA\_dnorm\_01 | 0.054 | < 0.001 | 0.012 | 1,073 |
| paminLastweek | 0.005 | < 0.001 | 0.011 | 1,082 |
| PA\_integrated | 0.082 | < 0.001 | 0.008 | 1,073 |
| PA\_autonomous | 0.073 | 0.005 | 0.006 | 1,078 |
| PA\_intrinsic | 0.071 | 0.009 | 0.006 | 1,074 |
| pafreqUsually | 0.069 | 0.023 | 0.005 | 1,082 |
| PA\_dnorm | 0.052 | 0.005 | 0.004 | 1,073 |
| pahrsUsually | 0.051 | 0.033 | 0.004 | 1,082 |
| PA\_inorm | < 0.001 | 0.006 | 0.003 | 1,073 |
| PA\_inorm\_01 | < 0.001 | 0.006 | 0.003 | 1,073 |
| PA\_outcomeExpectations | 0.024 | 0.029 | 0.003 | 1,078 |
| PA\_actCop | 0.036 | 0.006 | < 0.001 | 1,073 |
| PA\_actionplan | 0.031 | 0.013 | < 0.001 | 1,073 |
| PA\_actplan\_01 | 0.033 | 0.006 | < 0.001 | 1,073 |
| PA\_actplan\_02 | 0.018 | 0.015 | < 0.001 | 1,073 |
| PA\_actplan\_03 | 0.02 | 0.021 | < 0.001 | 1,073 |
| PA\_actplan\_04 | 0.038 | 0.008 | < 0.001 | 1,073 |
| PA\_controlled | 0.051 | 0.004 | < 0.001 | 1,073 |
| PA\_copingplan | 0.034 | < 0.001 | < 0.001 | 1,073 |
| PA\_copplan\_01 | 0.022 | < 0.001 | < 0.001 | 1,073 |
| PA\_copplan\_02 | 0.015 | < 0.001 | < 0.001 | 1,073 |
| PA\_copplan\_03 | 0.034 | 0.011 | < 0.001 | 1,073 |
| PA\_copplan\_04 | 0.033 | 0.01 | < 0.001 | 1,073 |
| PA\_dnorm\_02 | 0.023 | 0.013 | < 0.001 | 1,073 |
| PA\_identified | 0.044 | 0.006 | < 0.001 | 1,074 |
| PA\_intention | 0.064 | 0.039 | < 0.001 | 1,073 |
| PA\_intention\_01 | 0.061 | 0.027 | < 0.001 | 1,073 |
| PA\_intention\_02 | 0.064 | 0.044 | < 0.001 | 1,073 |
| PA\_introjected | 0.001 | < 0.001 | < 0.001 | 1,073 |
| PA\_opportunities | < 0.001 | 0.018 | < 0.001 | 1,075 |
| padaysLastweek | 0.04 | 0.006 | < 0.001 | 1,082 |
| pahrsLastweek | 0.036 | 0.004 | < 0.001 | 1,082 |
| paLastweek | 0.037 | 0.004 | < 0.001 | 1,082 |
| symptom | 0.066 | 0.013 | < 0.001 | 1,084 |

### Top 20 items, sorted by classroom

vardatatable\_containing\_edutrack %>% select(Variable, `ICC educational track`, `ICC class`, `ICC school`, n) %>% arrange(desc(`ICC class`)) %>% top\_n(20) %>% papaja::apa\_table(caption = "Intra-class correlations sorted by classroom ICC",  
 format.args = list(digits = c(0), margin = 2))  
## Selecting by n

(#tab:icc-school-class-track-sortbyclass)

*Intra-class correlations sorted by classroom ICC*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Variable | ICC educational track | ICC class | ICC school | n |
| PA\_intention\_02 | 0.064 | 0.044 | < 0.001 | 1,073 |
| PA\_intention | 0.064 | 0.039 | < 0.001 | 1,073 |
| pahrsUsually | 0.051 | 0.033 | 0.004 | 1,082 |
| PA\_outcomeExpectations | 0.024 | 0.029 | 0.003 | 1,078 |
| PA\_intention\_01 | 0.061 | 0.027 | < 0.001 | 1,073 |
| pafreqUsually | 0.069 | 0.023 | 0.005 | 1,082 |
| PA\_actplan\_03 | 0.02 | 0.021 | < 0.001 | 1,073 |
| PA\_opportunities | < 0.001 | 0.018 | < 0.001 | 1,075 |
| PA\_actplan\_02 | 0.018 | 0.015 | < 0.001 | 1,073 |
| PA\_actionplan | 0.031 | 0.013 | < 0.001 | 1,073 |
| PA\_dnorm\_02 | 0.023 | 0.013 | < 0.001 | 1,073 |
| symptom | 0.066 | 0.013 | < 0.001 | 1,084 |
| PA\_copplan\_03 | 0.034 | 0.011 | < 0.001 | 1,073 |
| PA\_copplan\_04 | 0.033 | 0.01 | < 0.001 | 1,073 |
| PA\_intrinsic | 0.071 | 0.009 | 0.006 | 1,074 |
| PA\_actplan\_04 | 0.038 | 0.008 | < 0.001 | 1,073 |
| PA\_actCop | 0.036 | 0.006 | < 0.001 | 1,073 |
| PA\_actplan\_01 | 0.033 | 0.006 | < 0.001 | 1,073 |
| PA\_identified | 0.044 | 0.006 | < 0.001 | 1,074 |
| PA\_inorm | < 0.001 | 0.006 | 0.003 | 1,073 |
| PA\_inorm\_01 | < 0.001 | 0.006 | 0.003 | 1,073 |
| padaysLastweek | 0.04 | 0.006 | < 0.001 | 1,082 |
| PA\_autonomous | 0.073 | 0.005 | 0.006 | 1,078 |
| PA\_dnorm | 0.052 | 0.005 | 0.004 | 1,073 |
| PA\_controlled | 0.051 | 0.004 | < 0.001 | 1,073 |
| pahrsLastweek | 0.036 | 0.004 | < 0.001 | 1,082 |
| paLastweek | 0.037 | 0.004 | < 0.001 | 1,082 |
| PA\_copingplan | 0.034 | < 0.001 | < 0.001 | 1,073 |
| PA\_copplan\_01 | 0.022 | < 0.001 | < 0.001 | 1,073 |
| PA\_copplan\_02 | 0.015 | < 0.001 | < 0.001 | 1,073 |
| PA\_dnorm\_01 | 0.054 | < 0.001 | 0.012 | 1,073 |
| PA\_integrated | 0.082 | < 0.001 | 0.008 | 1,073 |
| PA\_introjected | 0.001 | < 0.001 | < 0.001 | 1,073 |
| paminLastweek | 0.005 | < 0.001 | 0.011 | 1,082 |

### Top 20 items, sorted by educational track

vardatatable\_containing\_edutrack %>% select(Variable, `ICC educational track`, `ICC class`, `ICC school`, n) %>% arrange(desc(`ICC educational track`)) %>% top\_n(20) %>% papaja::apa\_table(caption = "Intra-class correlations sorted by educational track ICC",  
 format.args = list(digits = c(0), margin = 2))  
## Selecting by n

(#tab:icc-school-class-track-sortbytrack)

*Intra-class correlations sorted by educational track ICC*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Variable | ICC educational track | ICC class | ICC school | n |
| PA\_integrated | 0.082 | < 0.001 | 0.008 | 1,073 |
| PA\_autonomous | 0.073 | 0.005 | 0.006 | 1,078 |
| PA\_intrinsic | 0.071 | 0.009 | 0.006 | 1,074 |
| pafreqUsually | 0.069 | 0.023 | 0.005 | 1,082 |
| symptom | 0.066 | 0.013 | < 0.001 | 1,084 |
| PA\_intention | 0.064 | 0.039 | < 0.001 | 1,073 |
| PA\_intention\_02 | 0.064 | 0.044 | < 0.001 | 1,073 |
| PA\_intention\_01 | 0.061 | 0.027 | < 0.001 | 1,073 |
| PA\_dnorm\_01 | 0.054 | < 0.001 | 0.012 | 1,073 |
| PA\_dnorm | 0.052 | 0.005 | 0.004 | 1,073 |
| PA\_controlled | 0.051 | 0.004 | < 0.001 | 1,073 |
| pahrsUsually | 0.051 | 0.033 | 0.004 | 1,082 |
| PA\_identified | 0.044 | 0.006 | < 0.001 | 1,074 |
| padaysLastweek | 0.04 | 0.006 | < 0.001 | 1,082 |
| PA\_actplan\_04 | 0.038 | 0.008 | < 0.001 | 1,073 |
| paLastweek | 0.037 | 0.004 | < 0.001 | 1,082 |
| PA\_actCop | 0.036 | 0.006 | < 0.001 | 1,073 |
| pahrsLastweek | 0.036 | 0.004 | < 0.001 | 1,082 |
| PA\_copingplan | 0.034 | < 0.001 | < 0.001 | 1,073 |
| PA\_copplan\_03 | 0.034 | 0.011 | < 0.001 | 1,073 |
| PA\_actplan\_01 | 0.033 | 0.006 | < 0.001 | 1,073 |
| PA\_copplan\_04 | 0.033 | 0.01 | < 0.001 | 1,073 |
| PA\_actionplan | 0.031 | 0.013 | < 0.001 | 1,073 |
| PA\_outcomeExpectations | 0.024 | 0.029 | 0.003 | 1,078 |
| PA\_dnorm\_02 | 0.023 | 0.013 | < 0.001 | 1,073 |
| PA\_copplan\_01 | 0.022 | < 0.001 | < 0.001 | 1,073 |
| PA\_actplan\_03 | 0.02 | 0.021 | < 0.001 | 1,073 |
| PA\_actplan\_02 | 0.018 | 0.015 | < 0.001 | 1,073 |
| PA\_copplan\_02 | 0.015 | < 0.001 | < 0.001 | 1,073 |
| paminLastweek | 0.005 | < 0.001 | 0.011 | 1,082 |
| PA\_introjected | 0.001 | < 0.001 | < 0.001 | 1,073 |
| PA\_inorm | < 0.001 | 0.006 | 0.003 | 1,073 |
| PA\_inorm\_01 | < 0.001 | 0.006 | 0.003 | 1,073 |
| PA\_opportunities | < 0.001 | 0.018 | < 0.001 | 1,075 |

### Model with educational track only

#library(sjstats)  
#library(lme4)  
  
names <- df %>% select(-id, -intervention, -group, -school, -girl, -track, -trackSchool, -contains("\_T3"), -contains("\_diff")) %>% names(.)  
  
# Create empty soon-to-be-filled objects  
m <- NA  
mean <- NA  
m\_p <- NA  
ci\_low <- NA  
ci\_high <- NA  
ICC\_group <- NA  
ICC\_School <- NA  
nonmissings <- NA  
ICC\_track <- NA  
  
  
## To test the code with a single variable:  
# dftest <- df #%>% na.omit()  
# m <- lmer(sitLieAccelerometer\_T1 ~ (1|school) + (1|group) + (1|track), data=df)  
# mean <- fixef(m)  
# m\_p <- profile(m)  
# ci\_low <- confint(m\_p)[5]  
# ci\_high <- confint(m\_p)[10]  
# ICC\_group <- icc(m)[1]  
# ICC\_School <- icc(m)[2]  
# nonmissings <- length(m@resp$y)  
  
# Loop over each variable name, extract statistics:   
  
# all participants   
for (i in names){  
m <- lme4::lmer(paste0(i," ~ (1|school) + (1|group) + (1|track)"), data=df)  
mean[i] <- lme4::fixef(m)  
m\_p <- profile(m, which = "beta\_")  
ci\_low[i] <- confint(m\_p)[5]  
ci\_high[i] <- confint(m\_p)[10]  
ICC\_group[i] <- sjstats::icc(m)[1]  
ICC\_School[i] <- sjstats::icc(m)[3]  
ICC\_track[i] <- sjstats::icc(m)[2]  
nonmissings[i] <- length(m@resp$y)  
}  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): Last two rows have  
## identical or NA .zeta values: using minstep  
## Warning in sqrt(fv - base): NaNs produced  
## Warning in profile.merMod(m, which = "beta\_"): non-monotonic profile for  
## (Intercept)  
## Warning in confint.thpr(m\_p): bad spline fit for (Intercept): falling back  
## to linear interpolation  
  
## Warning in confint.thpr(m\_p): bad spline fit for (Intercept): falling back  
## to linear interpolation  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m, which = "beta\_"): non-monotonic profile for  
## (Intercept)  
## Warning in confint.thpr(m\_p): bad spline fit for (Intercept): falling back  
## to linear interpolation  
  
## Warning in confint.thpr(m\_p): bad spline fit for (Intercept): falling back  
## to linear interpolation  
  
vardatatable\_containing\_edutrack <- data\_frame(  
 Variable = stringr::str\_sub(names(mean), end = -4), #remove "\_T1" from names  
 Mean = round(mean, 1),   
 CI95 = paste(round(ci\_low, 1), "-", round(ci\_high, 1)),   
 "ICC class" =ifelse(round(ICC\_group, 3) == 0, "< 0.001", round(ICC\_group, 3)),   
 "ICC school" = ifelse(round(ICC\_School, 3) == 0, "< 0.001", round(ICC\_School, 3)),  
 "ICC educational track" = ifelse(round(ICC\_track, 3) == 0, "< 0.001", round(ICC\_track, 3)),  
 n = nonmissings) %>%   
 dplyr::arrange(Variable) %>%   
 filter(Variable != "") #remove first row, which is empty  
  
options(tibble.print\_max = 15)

Top 20 items, sorted by ICC educational track

vardatatable\_containing\_edutrack %>%   
 select(Variable, `ICC educational track`, n) %>%   
 arrange(desc(`ICC educational track`)) %>%   
 top\_n(20) %>%   
 papaja::apa\_table(caption = "Intra-class correlations sorted by educational track ICC",  
 format.args = list(digits = c(0), margin = 2))  
## Selecting by n

(#tab:icc-edu-track-only)

*Intra-class correlations sorted by educational track ICC*

|  |  |  |
| --- | --- | --- |
| Variable | ICC educational track | n |
| PA\_integrated | 0.082 | 1,073 |
| PA\_autonomous | 0.073 | 1,078 |
| PA\_intrinsic | 0.071 | 1,074 |
| pafreqUsually | 0.069 | 1,082 |
| symptom | 0.066 | 1,084 |
| PA\_intention | 0.064 | 1,073 |
| PA\_intention\_02 | 0.064 | 1,073 |
| PA\_intention\_01 | 0.061 | 1,073 |
| PA\_dnorm\_01 | 0.054 | 1,073 |
| PA\_dnorm | 0.052 | 1,073 |
| PA\_controlled | 0.051 | 1,073 |
| pahrsUsually | 0.051 | 1,082 |
| PA\_identified | 0.044 | 1,074 |
| padaysLastweek | 0.04 | 1,082 |
| PA\_actplan\_04 | 0.038 | 1,073 |
| paLastweek | 0.037 | 1,082 |
| PA\_actCop | 0.036 | 1,073 |
| pahrsLastweek | 0.036 | 1,082 |
| PA\_copingplan | 0.034 | 1,073 |
| PA\_copplan\_03 | 0.034 | 1,073 |
| PA\_actplan\_01 | 0.033 | 1,073 |
| PA\_copplan\_04 | 0.033 | 1,073 |
| PA\_actionplan | 0.031 | 1,073 |
| PA\_outcomeExpectations | 0.024 | 1,078 |
| PA\_dnorm\_02 | 0.023 | 1,073 |
| PA\_copplan\_01 | 0.022 | 1,073 |
| PA\_actplan\_03 | 0.02 | 1,073 |
| PA\_actplan\_02 | 0.018 | 1,073 |
| PA\_copplan\_02 | 0.015 | 1,073 |
| paminLastweek | 0.005 | 1,082 |
| PA\_introjected | 0.001 | 1,073 |
| PA\_inorm | < 0.001 | 1,073 |
| PA\_inorm\_01 | < 0.001 | 1,073 |
| PA\_opportunities | < 0.001 | 1,075 |

## Create confidence bounds for all items by gender and intervention allocation

These numbers are visualised with diamond plots later.

names <- df %>% select(-id, -intervention, -group, -school, -girl, -track, -trackSchool, -contains("\_T3"), -contains("\_diff")) %>% names(.)  
  
# Intercepts for boys; when boy is 1, girl is 0, but boy is a factor, so intercept is for boys even though boy is 1 for boys and 0 for girls.  
m.boys <- NA  
mean.boys <- NA  
m\_p.boys <- NA  
ci\_low.boys <- NA  
ci\_high.boys <- NA  
ICC\_group.boys <- NA  
ICC\_School.boys <- NA  
nonmissings.boys <- NA  
  
df.boys <- df %>% mutate(boy = factor(ifelse(girl == "girl", 0, 1), levels = c(1, 0)))  
  
for (i in names){  
m.boys <- lme4::lmer(paste0(i," ~ (1|school) + (1|group) + boy"), data=df.boys)  
mean.boys[i] <- lme4::fixef(m.boys)[1]  
m\_p.boys <- profile(m.boys, which = "beta\_")  
ci\_low.boys[i] <- confint(m\_p.boys)[1, 1]  
ci\_high.boys[i] <- confint(m\_p.boys)[1, 2]  
ICC\_group.boys[i] <- sjstats::icc(m.boys)[1]  
ICC\_School.boys[i] <- sjstats::icc(m.boys)[2]  
nonmissings.boys[i] <- length(m.boys@resp$y)  
}  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.boys, which = "beta\_"): non-monotonic profile  
## for (Intercept)  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in confint.thpr(m\_p.boys): bad spline fit for (Intercept): falling  
## back to linear interpolation  
  
## Warning in confint.thpr(m\_p.boys): bad spline fit for (Intercept): falling  
## back to linear interpolation  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in doTryCatch(return(expr), name, parentenv, handler): restarting  
## interrupted promise evaluation  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower =  
## fitted@lower): convergence code 1 from bobyqa: bobyqa -- maximum number of  
## function evaluations exceeded  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
cat("The labels are arranged such that intercept is not for girls:", labels(lme4::fixef(m.boys))[2] == "boy0")  
## The labels are arranged such that intercept is not for girls: TRUE  
  
ci\_boys <- data.frame(ciLo = ci\_low.boys, mean = mean.boys, ciHi = ci\_high.boys)  
diamondlabels <- labels(ci\_boys)[[1]]  
ci\_boys <- data.frame(ci\_boys, diamondlabels)  
  
# Intercepts for girls; when boy is 1, girl is 0, but boy is a factor, so intercept is for girls even though girl is 1 for girls and 0 for boys.  
m.girls <- NA  
mean.girls <- NA  
m\_p.girls <- NA  
ci\_low.girls <- NA  
ci\_high.girls <- NA  
ICC\_group.girls <- NA  
ICC\_School.girls <- NA  
nonmissings.girls <- NA  
  
for (i in names){  
m.girls <- lme4::lmer(paste0(i," ~ (1|school) + (1|group) + girl"), data=df)  
mean.girls[i] <- lme4::fixef(m.girls)[1]  
m\_p.girls <- profile(m.girls, which = "beta\_")  
ci\_low.girls[i] <- confint(m\_p.girls)[1, 1]  
ci\_high.girls[i] <- confint(m\_p.girls)[1, 2]  
ICC\_group.girls[i] <- sjstats::icc(m.girls)[1]  
ICC\_School.girls[i] <- sjstats::icc(m.girls)[2]  
nonmissings.girls[i] <- length(m.girls@resp$y)  
}  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in optwrap(optimizer, devfun, x@theta, lower = x@lower, calc.derivs  
## = TRUE, : convergence code 3 from bobyqa: bobyqa -- a trust region step  
## failed to reduce q  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.girls, which = "beta\_"): non-monotonic profile  
## for (Intercept)  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in confint.thpr(m\_p.girls): bad spline fit for (Intercept): falling  
## back to linear interpolation  
  
## Warning in confint.thpr(m\_p.girls): bad spline fit for (Intercept): falling  
## back to linear interpolation  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in optwrap(optimizer, devfun, x@theta, lower = x@lower, calc.derivs  
## = TRUE, : convergence code 3 from bobyqa: bobyqa -- a trust region step  
## failed to reduce q  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.girls, which = "beta\_"): non-monotonic profile  
## for (Intercept)  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in confint.thpr(m\_p.girls): bad spline fit for (Intercept): falling  
## back to linear interpolation  
  
## Warning in confint.thpr(m\_p.girls): bad spline fit for (Intercept): falling  
## back to linear interpolation  
  
cat("The labels are arranged such that intercept is not for boys:", labels(lme4::fixef(m.girls))[2] == "girl0")  
## The labels are arranged such that intercept is not for boys: FALSE  
  
ci\_girls <- data.frame(ciLo = ci\_low.girls, mean = mean.girls, ciHi = ci\_high.girls)  
diamondlabels <- labels(ci\_girls)[[1]]  
ci\_girls <- data.frame(ci\_girls, diamondlabels)  
  
  
# Intercepts for intervention  
m.intervention <- NA  
mean.intervention <- NA  
m\_p.intervention <- NA  
ci\_low.intervention <- NA  
ci\_high.intervention <- NA  
ICC\_group.intervention <- NA  
ICC\_School.intervention <- NA  
nonmissings.intervention <- NA  
  
## change "intervention" to be consistent regarding level order with "girl".  
df.intervention <- df %>% mutate(intervention = factor(intervention, levels = c(1, 0)))  
  
for (i in names){  
m.intervention <- lme4::lmer(paste0(i," ~ (1|school) + (1|group) + intervention"), data=df.intervention)  
mean.intervention[i] <- lme4::fixef(m.intervention)[1]  
m\_p.intervention <- profile(m.intervention, which = "beta\_")  
ci\_low.intervention[i] <- confint(m\_p.intervention)[1, 1]  
ci\_high.intervention[i] <- confint(m\_p.intervention)[1, 2]  
ICC\_group.intervention[i] <- sjstats::icc(m.intervention)[1]  
ICC\_School.intervention[i] <- sjstats::icc(m.intervention)[2]  
nonmissings.intervention[i] <- length(m.intervention@resp$y)  
}  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.intervention, which = "beta\_"): non-monotonic  
## profile for (Intercept)  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for (Intercept):  
## falling back to linear interpolation  
  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for (Intercept):  
## falling back to linear interpolation  
## Warning in optwrap(optimizer, devfun, x@theta, lower = x@lower, calc.derivs  
## = TRUE, : convergence code 3 from bobyqa: bobyqa -- a trust region step  
## failed to reduce q  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.intervention, which = "beta\_"): non-monotonic  
## profile for (Intercept)  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for (Intercept):  
## falling back to linear interpolation  
  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for (Intercept):  
## falling back to linear interpolation  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.intervention, which = "beta\_"): non-monotonic  
## profile for (Intercept)  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.intervention, which = "beta\_"): non-monotonic  
## profile for intervention0  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for (Intercept):  
## falling back to linear interpolation  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for  
## intervention0: falling back to linear interpolation  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for (Intercept):  
## falling back to linear interpolation  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for  
## intervention0: falling back to linear interpolation  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in profile.merMod(m.intervention, which = "beta\_"): non-monotonic  
## profile for (Intercept)  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.intervention, which = "beta\_"): non-monotonic  
## profile for intervention0  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for (Intercept):  
## falling back to linear interpolation  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for  
## intervention0: falling back to linear interpolation  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for (Intercept):  
## falling back to linear interpolation  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for  
## intervention0: falling back to linear interpolation  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.intervention, which = "beta\_"): non-monotonic  
## profile for (Intercept)  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.intervention, which = "beta\_"): non-monotonic  
## profile for intervention0  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for (Intercept):  
## falling back to linear interpolation  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for  
## intervention0: falling back to linear interpolation  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for (Intercept):  
## falling back to linear interpolation  
## Warning in confint.thpr(m\_p.intervention): bad spline fit for  
## intervention0: falling back to linear interpolation  
  
cat("The labels are arranged such that intercept is not for control:", labels(lme4::fixef(m.intervention))[2] == "intervention0")  
## The labels are arranged such that intercept is not for control: TRUE  
  
ci\_intervention <- data.frame(ciLo = ci\_low.intervention, mean = mean.intervention, ciHi = ci\_high.intervention)  
diamondlabels <- labels(ci\_intervention)[[1]]  
ci\_intervention <- data.frame(ci\_intervention, diamondlabels)  
  
# Intercepts for control  
  
m.control <- NA  
mean.control <- NA  
m\_p.control <- NA  
ci\_low.control <- NA  
ci\_high.control <- NA  
ICC\_group.control <- NA  
ICC\_School.control <- NA  
nonmissings.control <- NA  
  
df.control <- df %>% mutate(control = factor(ifelse(intervention == 1, 0, 1), levels = c(1, 0)))  
  
for (i in names){  
m.control <- lme4::lmer(paste0(i," ~ (1|school) + (1|group) + control"), data=df.control)  
mean.control[i] <- lme4::fixef(m.control)[1]  
m\_p.control <- profile(m.control, which = "beta\_")  
ci\_low.control[i] <- confint(m\_p.control)[1, 1]  
ci\_high.control[i] <- confint(m\_p.control)[1, 2]  
ICC\_group.control[i] <- sjstats::icc(m.control)[1]  
ICC\_School.control[i] <- sjstats::icc(m.control)[2]  
nonmissings.control[i] <- length(m.control@resp$y)  
}  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.control, which = "beta\_"): non-monotonic  
## profile for (Intercept)  
## Warning in confint.thpr(m\_p.control): bad spline fit for (Intercept):  
## falling back to linear interpolation  
  
## Warning in confint.thpr(m\_p.control): bad spline fit for (Intercept):  
## falling back to linear interpolation  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.control, which = "beta\_"): non-monotonic  
## profile for control0  
## Warning in confint.thpr(m\_p.control): bad spline fit for control0: falling  
## back to linear interpolation  
  
## Warning in confint.thpr(m\_p.control): bad spline fit for control0: falling  
## back to linear interpolation  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.control, which = "beta\_"): non-monotonic  
## profile for (Intercept)  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.control, which = "beta\_"): non-monotonic  
## profile for control0  
## Warning in confint.thpr(m\_p.control): bad spline fit for (Intercept):  
## falling back to linear interpolation  
## Warning in confint.thpr(m\_p.control): bad spline fit for control0: falling  
## back to linear interpolation  
## Warning in confint.thpr(m\_p.control): bad spline fit for (Intercept):  
## falling back to linear interpolation  
## Warning in confint.thpr(m\_p.control): bad spline fit for control0: falling  
## back to linear interpolation  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
  
## Warning in optwrap(optimizer, par = thopt, fn = mkdevfun(rho, 0L), lower  
## = fitted@lower): convergence code 3 from bobyqa: bobyqa -- a trust region  
## step failed to reduce q  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.control, which = "beta\_"): non-monotonic  
## profile for control0  
## Warning in confint.thpr(m\_p.control): bad spline fit for control0: falling  
## back to linear interpolation  
  
## Warning in confint.thpr(m\_p.control): bad spline fit for control0: falling  
## back to linear interpolation  
## Warning in nextpar(mat, cc, i, delta, lowcut, upcut): unexpected decrease  
## in profile: using minstep  
## Warning in profile.merMod(m.control, which = "beta\_"): non-monotonic  
## profile for (Intercept)  
## Warning in confint.thpr(m\_p.control): bad spline fit for (Intercept):  
## falling back to linear interpolation  
  
## Warning in confint.thpr(m\_p.control): bad spline fit for (Intercept):  
## falling back to linear interpolation  
  
cat("The labels are arranged such that intercept is not for intervention:", labels(lme4::fixef(m.control))[2] == "control0")  
## The labels are arranged such that intercept is not for intervention: TRUE  
  
ci\_control <- data.frame(ciLo = ci\_low.control, mean = mean.control, ciHi = ci\_high.control)  
diamondlabels <- labels(ci\_control)[[1]]  
ci\_control <- data.frame(ci\_control, diamondlabels)  
  
# Same ICC results you'd get with e.g.:  
# m1 <- as.data.frame(VarCorr(m))  
# m1$vcov[1] / (m1$vcov[1] + m1$vcov[3])  
  
# Or from broom:  
# tidy(m)$estimate[2]^2 / (tidy(m)$estimate[2]^2 + tidy(m)$estimate[4]^2)  
  
# Or from sjstats:  
# sum(get\_re\_var(m)) / (sum(get\_re\_var(m)) + get\_re\_var(m, "sigma\_2"))

# Descriptive tables

## prepare data

demographics <- lmi %>% dplyr::select(id = ID,  
birthYear = Kys0004.1,  
intervention = ryhma,  
school = Aineisto.1,  
girl = Kys0013.1,  
ethnicity = Kys0005.1,  
studyYear = Kys0014.1) %>%   
 mutate(age = 2016 - birthYear,  
 intervention = ifelse(intervention == 1, 1, 0),  
 intervention = as.numeric(intervention),  
 girl = ifelse(girl == 2, 1, 0),  
 girl = as.numeric(girl, levels = c("1", "0")),  
 school = factor(school, levels = c("1", "2", "3", "4", "5")),  
 bornInFinland = as.numeric(ifelse(ethnicity == 1, 1, 0)),  
 studyYear = ifelse(studyYear == 0, NA, studyYear)) # Remove "other" from study year  
  
# Insert track variable with those who answered "other" with one of the actual category labels given the appropriate category:  
Track <- lmi %>% select(Kys0016.1, Kys0017.1) %>% mutate(  
 Kys0016.1 = ifelse(Kys0017.1 == "Merkonomi" | Kys0017.1 == "merkonomi", 3,  
 ifelse(Kys0017.1 == "Datanomi" | Kys0017.1 == "datanomi", 2, Kys0016.1)),  
 Track = factor(Kys0016.1, # Fix track labels first  
 levels = c(0, 1, 2, 3, 4),  
 labels = c("Other", "Business IT", "Business Admin", "HRC", "Nursing"))) %>%   
 select(-Kys0016.1, -Kys0017.1)  
  
demographics <- bind\_cols(demographics, Track)

## Create demographic tables

### By educational track

demotable <- demographics %>%   
 group\_by(Track) %>%   
 summarise("Mean age (range)" = paste0(round(mean(age, na.rm = T), 1)," (",   
 range(age, na.rm = T)[1], "-",  
 range(age, na.rm = T)[2], ")"),  
 "Mean study year (sd)" = paste0(round(mean(studyYear, na.rm = T), 1)," (",   
 round(sd(studyYear, na.rm = T), 1), ")"),  
 "% girl" = round(mean(girl, na.rm = T)\*100, 1),  
 "% intervention" = round(mean(intervention, na.rm = T)\*100, 1),  
 "% born in Finland" = round(mean(bornInFinland, na.rm = T)\*100, 1),  
 n = n()) %>%   
 filter(complete.cases(.)) %>%   
 arrange(desc(Track))   
## Warning in min(x): no non-missing arguments to min; returning Inf  
## Warning in max(x): no non-missing arguments to max; returning -Inf  
## Warning in min(x): no non-missing arguments to min; returning Inf  
## Warning in max(x): no non-missing arguments to max; returning -Inf  
  
demotable\_total <- demographics %>%   
 summarise("Track" = "Full sample",  
 "Mean age (range)" = paste0(round(mean(age, na.rm = T), 1)," (",   
 range(age, na.rm = T)[1], "-",  
 range(age, na.rm = T)[2], ")"),  
 "Mean study year (sd)" = paste0(round(mean(studyYear, na.rm = T), 1)," (",   
 round(sd(studyYear, na.rm = T), 1), ")"),  
 "% girl" = round(mean(girl, na.rm = T)\*100, 1),  
 "% intervention" = round(mean(intervention, na.rm = T)\*100, 1),  
 "% born in Finland" = round(mean(bornInFinland, na.rm = T)\*100, 1),  
 n = n())  
  
demotable <- bind\_rows(demotable, demotable\_total)  
## Warning in bind\_rows\_(x, .id): binding factor and character vector,  
## coercing into character vector  
## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector  
  
demotable <- demotable %>% tidyr::gather(Variable, val, 2:ncol(demotable)) %>% tidyr::spread(Track, val)  
  
# For some reason, sum of n's of all tracks is 1084.  
  
papaja::apa\_table(demotable, caption = "Baseline demographics of educational tracks", digits = c(0, 1, 1, 1, 1, 1, 0))

(#tab:demographics-table-track)

*Baseline demographics of educational tracks*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Variable | Business Admin | Business IT | Full sample | HRC | Nursing | Other |
| % born in Finland | 87.1 | 87.9 | 83.1 | 88.2 | 80 | 54.2 |
| % girl | 39 | 16 | 56.5 | 60.6 | 82.3 | 70.8 |
| % intervention | 53.5 | 46.6 | 53.6 | 31.5 | 68.9 | 41.7 |
| Mean age (range) | 19 (17-36) | 19.5 (18-44) | 19.3 (17-50) | 18.5 (17-27) | 19.8 (17-50) | 21 (17-45) |
| Mean study year (sd) | 1.7 (0.9) | 1.7 (0.9) | 1.7 (0.9) | 1.9 (0.7) | 1.7 (0.9) | 2 (1.4) |
| n | 282 | 163 | 1165 | 213 | 402 | 24 |

### By gender

demographics$Girl <- factor(demographics$girl)  
  
demotable <- demographics %>%   
 group\_by(Girl) %>%   
 summarise("Mean age (range)" = paste0(round(mean(age, na.rm = T), 1)," (",   
 range(age, na.rm = T)[1], "-",  
 range(age, na.rm = T)[2], ")"),  
 "Mean study year (sd)" = paste0(round(mean(studyYear, na.rm = T), 1)," (",   
 round(sd(studyYear, na.rm = T), 1), ")"),  
 "% intervention" = round(mean(intervention, na.rm = T)\*100, 1),  
 "% born in Finland" = round(mean(bornInFinland, na.rm = T)\*100, 1),  
 n = n()) %>%   
 filter(complete.cases(.)) %>%   
 arrange(desc(Girl))   
## Warning in min(x): no non-missing arguments to min; returning Inf  
## Warning in max(x): no non-missing arguments to max; returning -Inf  
## Warning in min(x): no non-missing arguments to min; returning Inf  
## Warning in max(x): no non-missing arguments to max; returning -Inf  
  
demotable\_total <- demographics %>%   
 summarise("Girl" = "Full sample",  
 "Mean age (range)" = paste0(round(mean(age, na.rm = T), 1)," (",   
 range(age, na.rm = T)[1], "-",  
 range(age, na.rm = T)[2], ")"),  
 "Mean study year (sd)" = paste0(round(mean(studyYear, na.rm = T), 1)," (",   
 round(sd(studyYear, na.rm = T), 1), ")"),  
 "% intervention" = round(mean(intervention, na.rm = T)\*100, 1),  
 "% born in Finland" = round(mean(bornInFinland, na.rm = T)\*100, 1),  
 n = n())  
  
demotable <- bind\_rows(demotable, demotable\_total)  
## Warning in bind\_rows\_(x, .id): binding factor and character vector,  
## coercing into character vector  
## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector  
  
demotable <- demotable %>% tidyr::gather(Variable, val, 2:ncol(demotable)) %>% tidyr::spread(Girl, val)  
  
names(demotable) <- c("", "Boy", "Girl", "Full sample")  
  
# For some reason, sum of n's of all tracks is 1084.  
  
papaja::apa\_table(demotable, caption = "Baseline demographics of educational tracks", digits = c(0, 1, 1, 1, 1, 0))

(#tab:demographics-table-gender)

*Baseline demographics of educational tracks*

|  |  |  |  |
| --- | --- | --- | --- |
|  | Boy | Girl | Full sample |
| % born in Finland | 88.2 | 80.1 | 83.1 |
| % intervention | 50.5 | 56 | 53.6 |
| Mean age (range) | 19.1 (17-36) | 19.5 (17-50) | 19.3 (17-50) |
| Mean study year (sd) | 1.7 (0.9) | 1.7 (0.8) | 1.7 (0.9) |
| n | 471 | 613 | 1165 |

### By intervention participation

demographics$intervention <- factor(demographics$intervention)  
demographics$girl <- as.numeric(demographics$girl)  
  
demotable <- demographics %>%   
 group\_by(intervention) %>%   
 summarise("Mean age (range)" = paste0(round(mean(age, na.rm = T), 1)," (",   
 range(age, na.rm = T)[1], "-",  
 range(age, na.rm = T)[2], ")"),  
 "Mean study year (sd)" = paste0(round(mean(studyYear, na.rm = T), 1)," (",   
 round(sd(studyYear, na.rm = T), 1), ")"),  
 "% girl" = round(mean(girl, na.rm = T)\*100, 1),  
 "% born in Finland" = round(mean(bornInFinland, na.rm = T)\*100, 1),  
 n = n()) %>%   
 filter(complete.cases(.)) %>%   
 arrange(desc(intervention))   
## Warning in min(x): no non-missing arguments to min; returning Inf  
## Warning in max(x): no non-missing arguments to max; returning -Inf  
## Warning in min(x): no non-missing arguments to min; returning Inf  
## Warning in max(x): no non-missing arguments to max; returning -Inf  
  
demotable\_total <- demographics %>%   
 summarise("intervention" = "Full sample",  
 "Mean age (range)" = paste0(round(mean(age, na.rm = T), 1)," (",   
 range(age, na.rm = T)[1], "-",  
 range(age, na.rm = T)[2], ")"),  
 "Mean study year (sd)" = paste0(round(mean(studyYear, na.rm = T), 1)," (",   
 round(sd(studyYear, na.rm = T), 1), ")"),  
 "% girl" = round(mean(girl, na.rm = T)\*100, 1),  
 "% born in Finland" = round(mean(bornInFinland, na.rm = T)\*100, 1),  
 n = n())  
  
demotable <- bind\_rows(demotable, demotable\_total)  
## Warning in bind\_rows\_(x, .id): binding factor and character vector,  
## coercing into character vector  
## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector  
  
demotable <- demotable %>% tidyr::gather(Variable, val, 2:ncol(demotable)) %>% tidyr::spread(intervention, val)  
  
names(demotable) <- c("", "Control", "Intervention", "Full sample")  
  
# For some reason, sum of n's of all tracks is 1084.  
  
papaja::apa\_table(demotable, caption = "Baseline demographics of educational tracks", digits = c(0, 1, 1, 1, 1, 0))

(#tab:demographics-table-intervention)

*Baseline demographics of educational tracks*

|  |  |  |  |
| --- | --- | --- | --- |
|  | Control | Intervention | Full sample |
| % born in Finland | 87.5 | 79.6 | 83.1 |
| % girl | 53.7 | 59 | 56.5 |
| Mean age (range) | 19 (17-39) | 19.6 (17-50) | 19.3 (17-50) |
| Mean study year (sd) | 1.7 (0.9) | 1.7 (0.9) | 1.7 (0.9) |
| n | 503 | 581 | 1165 |

## Create outcome tables

outtable\_track <- df %>%  
 select(paAccelerometer\_T1, sitLieAccelerometer\_T1, sitBreaks\_T1, # Main outcomes   
 PA\_actionplan\_T1,  
 PA\_copingplan\_T1,  
 PA\_agrbct\_T1,  
 PA\_frqbct\_T1,  
 PA\_amotivation\_T1,  
 PA\_autonomous\_T1,  
 PA\_controlled\_T1,  
 PA\_goal\_T1,  
 PA\_inorm\_T1,  
 PA\_dnorm\_T1,  
 PA\_intention\_T1,  
 PA\_outcomeExpectations\_T1,  
 PA\_opportunities\_T1,  
 PA\_pbc\_T1,  
 PA\_selfefficacy\_T1,  
 SB\_dnorm\_T1,  
 SB\_inorm\_T1,  
 SB\_intention\_T1,  
 SB\_outcomeExpectations\_T1,  
 SB\_sePbc\_T1,  
 track, girl, intervention) %>%   
 group\_by(track) %>%   
 summarise(  
 'PA action planning' = paste0(round(mean(PA\_actionplan\_T1, na.rm = TRUE), 1), " (", round(sd(PA\_actionplan\_T1, na.rm = T), 1), ")"),  
 'PA coping planning' = paste0(round(mean(PA\_copingplan\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_copingplan\_T1, na.rm = T), 1), ')'),  
 'PA agreement-BCTs' = paste0(round(mean(PA\_agrbct\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_agrbct\_T1, na.rm = T), 1), ')'),  
 'PA frequency-BCTs' = paste0(round(mean(PA\_frqbct\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_frqbct\_T1, na.rm = T), 1), ')'),  
 'PA amotivation' = paste0(round(mean(PA\_amotivation\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_amotivation\_T1, na.rm = T), 1), ')'),  
 'PA autonomous regulation' = paste0(round(mean(PA\_autonomous\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_autonomous\_T1, na.rm = T), 1), ')'),  
 'PA controlled regulation' = paste0(round(mean(PA\_controlled\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_controlled\_T1, na.rm = T), 1), ')'),  
 'PA injunctive norm' = paste0(round(mean(PA\_inorm\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_inorm\_T1, na.rm = T), 1), ')'),  
 'PA descriptive norm' = paste0(round(mean(PA\_dnorm\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_dnorm\_T1, na.rm = T), 1), ')'),  
 'PA intention' = paste0(round(mean(PA\_intention\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_intention\_T1, na.rm = T), 1), ')'),  
 'PA outcome expectations' = paste0(round(mean(PA\_outcomeExpectations\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_outcomeExpectations\_T1, na.rm = T), 1), ')'),  
 'PA opportunities' = paste0(round(mean(PA\_opportunities\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_opportunities\_T1, na.rm = T), 1), ')'),  
 'PA perceived behavioural control' = paste0(round(mean(PA\_pbc\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_pbc\_T1, na.rm = T), 1), ')'),  
 'PA self-efficacy' = paste0(round(mean(PA\_selfefficacy\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_selfefficacy\_T1, na.rm = T), 1), ')'),  
 'SB descriptive norm' = paste0(round(mean(SB\_dnorm\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_dnorm\_T1, na.rm = T), 1), ')'),  
 'SB injunctive norm' = paste0(round(mean(SB\_inorm\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_inorm\_T1, na.rm = T), 1), ')'),  
 'SB intention' = paste0(round(mean(SB\_intention\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_intention\_T1, na.rm = T), 1), ')'),  
 'SB outcome expectations' = paste0(round(mean(SB\_outcomeExpectations\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_outcomeExpectations\_T1, na.rm = T), 1), ')'),  
 'SB self-efficacy & perceived behavioural control' = paste0(round(mean(SB\_sePbc\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_sePbc\_T1, na.rm = T), 1), ')'),  
  
 "Mean daily MVPA minutes" = paste0(round(mean(paAccelerometer\_T1, na.rm = T), 1)," (",   
 round(sd(paAccelerometer\_T1, na.rm = T), 1), ")"),  
 "Mean daily minutes spent sitting or lying down" = paste0(round(mean(sitLieAccelerometer\_T1, na.rm = T), 1)," (",   
 round(sd(sitLieAccelerometer\_T1, na.rm = T), 1), ")"),  
 "Mean daily breaks in sitting" = paste0(round(mean(sitBreaks\_T1, na.rm = T), 1)," (",   
 round(sd(sitBreaks\_T1, na.rm = T), 1), ")"),  
 n = n()) %>%   
 filter(complete.cases(.)) %>%   
 arrange(desc(track)) %>%   
 filter(track != "Other")  
  
outtable\_intervention <- df %>%  
 select(paAccelerometer\_T1, sitLieAccelerometer\_T1, sitBreaks\_T1, # Main outcomes   
 PA\_actionplan\_T1,  
 PA\_copingplan\_T1,  
 PA\_agrbct\_T1,  
 PA\_frqbct\_T1,  
 PA\_amotivation\_T1,  
 PA\_autonomous\_T1,  
 PA\_controlled\_T1,  
 PA\_goal\_T1,  
 PA\_inorm\_T1,  
 PA\_dnorm\_T1,  
 PA\_intention\_T1,  
 PA\_outcomeExpectations\_T1,  
 PA\_opportunities\_T1,  
 PA\_pbc\_T1,  
 PA\_selfefficacy\_T1,  
 SB\_dnorm\_T1,  
 SB\_inorm\_T1,  
 SB\_intention\_T1,  
 SB\_outcomeExpectations\_T1,  
 SB\_sePbc\_T1,  
 track, girl, intervention) %>%   
 group\_by(intervention) %>%   
 summarise(  
 'PA action planning' = paste0(round(mean(PA\_actionplan\_T1, na.rm = TRUE), 1), " (", round(sd(PA\_actionplan\_T1, na.rm = T), 1), ")"),  
 'PA coping planning' = paste0(round(mean(PA\_copingplan\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_copingplan\_T1, na.rm = T), 1), ')'),  
 'PA agreement-BCTs' = paste0(round(mean(PA\_agrbct\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_agrbct\_T1, na.rm = T), 1), ')'),  
 'PA frequency-BCTs' = paste0(round(mean(PA\_frqbct\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_frqbct\_T1, na.rm = T), 1), ')'),  
 'PA amotivation' = paste0(round(mean(PA\_amotivation\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_amotivation\_T1, na.rm = T), 1), ')'),  
 'PA autonomous regulation' = paste0(round(mean(PA\_autonomous\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_autonomous\_T1, na.rm = T), 1), ')'),  
 'PA controlled regulation' = paste0(round(mean(PA\_controlled\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_controlled\_T1, na.rm = T), 1), ')'),  
 'PA injunctive norm' = paste0(round(mean(PA\_inorm\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_inorm\_T1, na.rm = T), 1), ')'),  
 'PA descriptive norm' = paste0(round(mean(PA\_dnorm\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_dnorm\_T1, na.rm = T), 1), ')'),  
 'PA intention' = paste0(round(mean(PA\_intention\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_intention\_T1, na.rm = T), 1), ')'),  
 'PA outcome expectations' = paste0(round(mean(PA\_outcomeExpectations\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_outcomeExpectations\_T1, na.rm = T), 1), ')'),  
 'PA opportunities' = paste0(round(mean(PA\_opportunities\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_opportunities\_T1, na.rm = T), 1), ')'),  
 'PA perceived behavioural control' = paste0(round(mean(PA\_pbc\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_pbc\_T1, na.rm = T), 1), ')'),  
 'PA self-efficacy' = paste0(round(mean(PA\_selfefficacy\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_selfefficacy\_T1, na.rm = T), 1), ')'),  
 'SB descriptive norm' = paste0(round(mean(SB\_dnorm\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_dnorm\_T1, na.rm = T), 1), ')'),  
 'SB injunctive norm' = paste0(round(mean(SB\_inorm\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_inorm\_T1, na.rm = T), 1), ')'),  
 'SB intention' = paste0(round(mean(SB\_intention\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_intention\_T1, na.rm = T), 1), ')'),  
 'SB outcome expectations' = paste0(round(mean(SB\_outcomeExpectations\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_outcomeExpectations\_T1, na.rm = T), 1), ')'),  
 'SB self-efficacy & perceived behavioural control' = paste0(round(mean(SB\_sePbc\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_sePbc\_T1, na.rm = T), 1), ')'),  
  
 "Mean daily MVPA minutes" = paste0(round(mean(paAccelerometer\_T1, na.rm = T), 1)," (",   
 round(sd(paAccelerometer\_T1, na.rm = T), 1), ")"),  
 "Mean daily minutes spent sitting or lying down" = paste0(round(mean(sitLieAccelerometer\_T1, na.rm = T), 1)," (",   
 round(sd(sitLieAccelerometer\_T1, na.rm = T), 1), ")"),  
 "Mean daily breaks in sitting" = paste0(round(mean(sitBreaks\_T1, na.rm = T), 1)," (",   
 round(sd(sitBreaks\_T1, na.rm = T), 1), ")"),  
 n = n()) %>%   
 filter(complete.cases(.))  
  
outtable\_girl <- df %>%  
 select(paAccelerometer\_T1, sitLieAccelerometer\_T1, sitBreaks\_T1, # Main outcomes   
 PA\_actionplan\_T1,  
 PA\_copingplan\_T1,  
 PA\_agrbct\_T1,  
 PA\_frqbct\_T1,  
 PA\_amotivation\_T1,  
 PA\_autonomous\_T1,  
 PA\_controlled\_T1,  
 PA\_goal\_T1,  
 PA\_inorm\_T1,  
 PA\_dnorm\_T1,  
 PA\_intention\_T1,  
 PA\_outcomeExpectations\_T1,  
 PA\_opportunities\_T1,  
 PA\_pbc\_T1,  
 PA\_selfefficacy\_T1,  
 SB\_dnorm\_T1,  
 SB\_inorm\_T1,  
 SB\_intention\_T1,  
 SB\_outcomeExpectations\_T1,  
 SB\_sePbc\_T1,  
 track, girl, intervention) %>%   
 group\_by(girl) %>%   
 summarise(  
 'PA action planning' = paste0(round(mean(PA\_actionplan\_T1, na.rm = TRUE), 1), " (", round(sd(PA\_actionplan\_T1, na.rm = T), 1), ")"),  
 'PA coping planning' = paste0(round(mean(PA\_copingplan\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_copingplan\_T1, na.rm = T), 1), ')'),  
 'PA agreement-BCTs' = paste0(round(mean(PA\_agrbct\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_agrbct\_T1, na.rm = T), 1), ')'),  
 'PA frequency-BCTs' = paste0(round(mean(PA\_frqbct\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_frqbct\_T1, na.rm = T), 1), ')'),  
 'PA amotivation' = paste0(round(mean(PA\_amotivation\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_amotivation\_T1, na.rm = T), 1), ')'),  
 'PA autonomous regulation' = paste0(round(mean(PA\_autonomous\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_autonomous\_T1, na.rm = T), 1), ')'),  
 'PA controlled regulation' = paste0(round(mean(PA\_controlled\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_controlled\_T1, na.rm = T), 1), ')'),  
 'PA injunctive norm' = paste0(round(mean(PA\_inorm\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_inorm\_T1, na.rm = T), 1), ')'),  
 'PA descriptive norm' = paste0(round(mean(PA\_dnorm\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_dnorm\_T1, na.rm = T), 1), ')'),  
 'PA intention' = paste0(round(mean(PA\_intention\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_intention\_T1, na.rm = T), 1), ')'),  
 'PA outcome expectations' = paste0(round(mean(PA\_outcomeExpectations\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_outcomeExpectations\_T1, na.rm = T), 1), ')'),  
 'PA opportunities' = paste0(round(mean(PA\_opportunities\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_opportunities\_T1, na.rm = T), 1), ')'),  
 'PA perceived behavioural control' = paste0(round(mean(PA\_pbc\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_pbc\_T1, na.rm = T), 1), ')'),  
 'PA self-efficacy' = paste0(round(mean(PA\_selfefficacy\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_selfefficacy\_T1, na.rm = T), 1), ')'),  
 'SB descriptive norm' = paste0(round(mean(SB\_dnorm\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_dnorm\_T1, na.rm = T), 1), ')'),  
 'SB injunctive norm' = paste0(round(mean(SB\_inorm\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_inorm\_T1, na.rm = T), 1), ')'),  
 'SB intention' = paste0(round(mean(SB\_intention\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_intention\_T1, na.rm = T), 1), ')'),  
 'SB outcome expectations' = paste0(round(mean(SB\_outcomeExpectations\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_outcomeExpectations\_T1, na.rm = T), 1), ')'),  
 'SB self-efficacy & perceived behavioural control' = paste0(round(mean(SB\_sePbc\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_sePbc\_T1, na.rm = T), 1), ')'),  
  
 "Mean daily MVPA minutes" = paste0(round(mean(paAccelerometer\_T1, na.rm = T), 1)," (",   
 round(sd(paAccelerometer\_T1, na.rm = T), 1), ")"),  
 "Mean daily minutes spent sitting or lying down" = paste0(round(mean(sitLieAccelerometer\_T1, na.rm = T), 1)," (",   
 round(sd(sitLieAccelerometer\_T1, na.rm = T), 1), ")"),  
 "Mean daily breaks in sitting" = paste0(round(mean(sitBreaks\_T1, na.rm = T), 1)," (",   
 round(sd(sitBreaks\_T1, na.rm = T), 1), ")"),  
 n = n()) %>%   
 filter(complete.cases(.))  
  
outtable\_total <- df %>%   
 summarise('girl' = "Full sample",  
 'PA action planning' = paste0(round(mean(PA\_actionplan\_T1, na.rm = TRUE), 1), " (", round(sd(PA\_actionplan\_T1, na.rm = T), 1), ")"),  
 'PA coping planning' = paste0(round(mean(PA\_copingplan\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_copingplan\_T1, na.rm = T), 1), ')'),  
 'PA agreement-BCTs' = paste0(round(mean(PA\_agrbct\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_agrbct\_T1, na.rm = T), 1), ')'),  
 'PA frequency-BCTs' = paste0(round(mean(PA\_frqbct\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_frqbct\_T1, na.rm = T), 1), ')'),  
 'PA amotivation' = paste0(round(mean(PA\_amotivation\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_amotivation\_T1, na.rm = T), 1), ')'),  
 'PA autonomous regulation' = paste0(round(mean(PA\_autonomous\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_autonomous\_T1, na.rm = T), 1), ')'),  
 'PA controlled regulation' = paste0(round(mean(PA\_controlled\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_controlled\_T1, na.rm = T), 1), ')'),  
 'PA injunctive norm' = paste0(round(mean(PA\_inorm\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_inorm\_T1, na.rm = T), 1), ')'),  
 'PA descriptive norm' = paste0(round(mean(PA\_dnorm\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_dnorm\_T1, na.rm = T), 1), ')'),  
 'PA intention' = paste0(round(mean(PA\_intention\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_intention\_T1, na.rm = T), 1), ')'),  
 'PA outcome expectations' = paste0(round(mean(PA\_outcomeExpectations\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_outcomeExpectations\_T1, na.rm = T), 1), ')'),  
 'PA opportunities' = paste0(round(mean(PA\_opportunities\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_opportunities\_T1, na.rm = T), 1), ')'),  
 'PA perceived behavioural control' = paste0(round(mean(PA\_pbc\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_pbc\_T1, na.rm = T), 1), ')'),  
 'PA self-efficacy' = paste0(round(mean(PA\_selfefficacy\_T1, na.rm = TRUE), 1), ' (', round(sd(PA\_selfefficacy\_T1, na.rm = T), 1), ')'),  
 'SB descriptive norm' = paste0(round(mean(SB\_dnorm\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_dnorm\_T1, na.rm = T), 1), ')'),  
 'SB injunctive norm' = paste0(round(mean(SB\_inorm\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_inorm\_T1, na.rm = T), 1), ')'),  
 'SB intention' = paste0(round(mean(SB\_intention\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_intention\_T1, na.rm = T), 1), ')'),  
 'SB outcome expectations' = paste0(round(mean(SB\_outcomeExpectations\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_outcomeExpectations\_T1, na.rm = T), 1), ')'),  
 'SB self-efficacy & perceived behavioural control' = paste0(round(mean(SB\_sePbc\_T1, na.rm = TRUE), 1), ' (', round(sd(SB\_sePbc\_T1, na.rm = T), 1), ')'),  
  
 "Mean daily MVPA minutes" = paste0(round(mean(paAccelerometer\_T1, na.rm = T), 1)," (",   
 round(sd(paAccelerometer\_T1, na.rm = T), 1), ")"),  
 "Mean daily minutes spent sitting or lying down" = paste0(round(mean(sitLieAccelerometer\_T1, na.rm = T), 1)," (",   
 round(sd(sitLieAccelerometer\_T1, na.rm = T), 1), ")"),  
 "Mean daily breaks in sitting" = paste0(round(mean(sitBreaks\_T1, na.rm = T), 1)," (",   
 round(sd(sitBreaks\_T1, na.rm = T), 1), ")"),  
 n = n())  
  
  
  
# Transpose each table  
outtable\_track <- outtable\_track %>% tidyr::gather(Variable, val, 2:ncol(.)) %>% tidyr::spread(track, val)  
  
outtable\_intervention <- outtable\_intervention %>% tidyr::gather(Variable, val, 2:ncol(.)) %>% tidyr::spread(intervention, val)  
names(outtable\_intervention) <- c("Variable", "Control", "Intervention")  
  
## In the last table, have a column for total before transposing  
outtable\_girl <- bind\_rows(outtable\_girl, outtable\_total)  
## Warning in bind\_rows\_(x, .id): binding factor and character vector,  
## coercing into character vector  
## Warning in bind\_rows\_(x, .id): binding character and factor vector,  
## coercing into character vector  
outtable\_girl <- outtable\_girl %>% tidyr::gather(Variable, val, 2:ncol(.)) %>% tidyr::spread(girl, val)  
names(outtable\_girl) <- c("Variable", "Boy", "Girl", "Full sample")  
  
# Are all variables the same?  
identical(outtable\_track$Variable, outtable\_girl$Variable)

[1] TRUE

identical(outtable\_track$Variable, outtable\_intervention$Variable)

[1] TRUE

# Create large table with all variables  
outtable\_mega <- bind\_cols(outtable\_track, outtable\_intervention, outtable\_girl) %>%   
 select(-Variable1, -Variable2)   
  
# Move n to be the first row  
out1 <- outtable\_mega %>% filter(Variable == "n")  
out2 <- outtable\_mega %>% filter(Variable != "n")  
outtable\_mega <- bind\_rows(out1, out2)  
  
# Fix names: leave first blank, have the rest as they were  
names(outtable\_mega) <- c("", names(outtable\_mega)[2:(length(names(outtable\_mega)))])  
  
papaja::apa\_table(outtable\_mega, caption = "Baseline demographics of educational tracks")

(#tab:outcome-table)

*Baseline demographics of educational tracks*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Business IT | Business Admin | HRC | Nursing | Control | Intervention | Boy | Girl | Full sample |
| n | 163 | 282 | 213 | 402 | 503 | 581 | 471 | 613 | 1165 |
| Mean daily breaks in sitting | 20 (9) | 27.5 (10) | 27.5 (9.6) | 31.7 (11.3) | 27.2 (10.4) | 29 (11.6) | 23.4 (9.5) | 31.3 (11) | 28.2 (11.1) |
| Mean daily minutes spent sitting or lying down | 652.8 (86.3) | 586.8 (93) | 557.8 (101.2) | 554.4 (82.1) | 568.5 (100.9) | 585.9 (92.7) | 612.5 (95.6) | 554.9 (90.9) | 578.2 (97.2) |
| Mean daily MVPA minutes | 141.3 (51.6) | 179.6 (48.4) | 183.1 (52.2) | 201.2 (49.2) | 179.5 (52) | 187.3 (55.1) | 173.5 (59.4) | 190.4 (48.6) | 183.8 (53.8) |
| PA action planning | 2.4 (1.1) | 2.9 (0.9) | 2.8 (0.9) | 2.8 (0.9) | 2.8 (1) | 2.7 (0.9) | 2.8 (1) | 2.7 (0.9) | 2.8 (0.9) |
| PA agreement-BCTs | 2.6 (1.3) | 3.5 (1.2) | 3.1 (1.3) | 3 (1.3) | 3.2 (1.3) | 3 (1.3) | 3.1 (1.4) | 3.1 (1.3) | 3.1 (1.3) |
| PA amotivation | 1.8 (1) | 1.5 (0.8) | 1.5 (0.8) | 1.4 (0.7) | 1.6 (0.8) | 1.5 (0.8) | 1.6 (0.9) | 1.5 (0.7) | 1.5 (0.8) |
| PA autonomous regulation | 2.9 (1.1) | 3.6 (1) | 3.4 (1.1) | 3.5 (1) | 3.5 (1) | 3.4 (1.1) | 3.5 (1.1) | 3.4 (1) | 3.4 (1.1) |
| PA controlled regulation | 1.9 (0.8) | 1.8 (0.8) | 1.8 (0.9) | 1.8 (0.8) | 1.9 (0.9) | 1.8 (0.8) | 1.8 (0.8) | 1.9 (0.8) | 1.8 (0.8) |
| PA coping planning | 2.2 (0.9) | 2.7 (0.8) | 2.5 (0.9) | 2.5 (0.9) | 2.5 (0.9) | 2.5 (0.9) | 2.6 (0.9) | 2.5 (0.8) | 2.5 (0.9) |
| PA descriptive norm | 3.8 (1.6) | 4.8 (1.5) | 4.5 (1.5) | 4.2 (1.6) | 4.5 (1.6) | 4.3 (1.6) | 4.5 (1.6) | 4.3 (1.6) | 4.4 (1.6) |
| PA frequency-BCTs | 2.2 (1.1) | 2.8 (1.2) | 2.5 (1.1) | 2.5 (1.1) | 2.6 (1.1) | 2.5 (1.1) | 2.6 (1.2) | 2.5 (1.1) | 2.5 (1.1) |
| PA injunctive norm | 4.7 (2) | 4.8 (2) | 4.7 (2) | 4.5 (2) | 4.8 (2) | 4.5 (2) | 4.8 (2) | 4.6 (2) | 4.6 (2) |
| PA intention | 4.3 (2.1) | 5.7 (1.7) | 5.5 (1.8) | 5.5 (1.7) | 5.4 (1.9) | 5.4 (1.8) | 5.3 (1.9) | 5.4 (1.8) | 5.4 (1.8) |
| PA opportunities | 5.1 (0.9) | 5.2 (1) | 5.1 (0.9) | 5.1 (1) | 5.2 (0.9) | 5.1 (1) | 5.2 (0.9) | 5.1 (1) | 5.1 (0.9) |
| PA outcome expectations | 4.3 (0.9) | 4.6 (0.9) | 4.6 (0.9) | 4.7 (0.9) | 4.6 (0.9) | 4.6 (0.9) | 4.5 (0.9) | 4.7 (0.9) | 4.6 (0.9) |
| PA perceived behavioural control | 5.3 (1.3) | 5.5 (1.2) | 5.2 (1.3) | 5.3 (1.3) | 5.3 (1.3) | 5.3 (1.3) | 5.5 (1.2) | 5.2 (1.3) | 5.3 (1.3) |
| PA self-efficacy | 5.2 (1.3) | 5.4 (1.4) | 5.1 (1.2) | 5.2 (1.3) | 5.3 (1.3) | 5.2 (1.3) | 5.3 (1.3) | 5.1 (1.3) | 5.2 (1.3) |
| SB descriptive norm | 2.8 (1.5) | 3.2 (1.4) | 3.4 (1.4) | 3.4 (1.5) | 3.3 (1.4) | 3.2 (1.5) | 3.3 (1.5) | 3.2 (1.4) | 3.2 (1.5) |
| SB injunctive norm | 4 (1.1) | 3.8 (1.3) | 4 (1.2) | 4.1 (1.3) | 4.1 (1.2) | 3.9 (1.3) | 4 (1.2) | 4 (1.3) | 4 (1.3) |
| SB intention | 3 (1.6) | 3.6 (1.5) | 3.7 (1.6) | 4 (1.6) | 3.7 (1.6) | 3.7 (1.6) | 3.5 (1.7) | 3.9 (1.6) | 3.7 (1.6) |
| SB outcome expectations | 4.1 (1.1) | 4.4 (1) | 4.4 (1) | 4.5 (1) | 4.4 (1) | 4.4 (1.1) | 4.3 (1.1) | 4.5 (1) | 4.4 (1) |
| SB self-efficacy & perceived behavioural control | 5 (1.3) | 4.9 (1.2) | 5.1 (1.2) | 4.9 (1.3) | 5.1 (1.2) | 4.9 (1.3) | 5 (1.3) | 4.9 (1.2) | 4.9 (1.3) |

## Mediator table with CIs

mediatortable\_intervention <- ci\_intervention %>%  
 dplyr::filter(grepl('PA\_actionplan\_T1|PA\_copingplan\_T1|PA\_agrbct\_T1|PA\_frqbct\_T1|PA\_amotivation\_T1|PA\_autonomous\_T1|PA\_controlled\_T1|PA\_goal\_T1|PA\_inorm\_T1|PA\_dnorm\_T1|PA\_intention\_T1|PA\_outcomeExpectations\_T1|PA\_opportunities\_T1|PA\_pbc\_T1|PA\_selfefficacy\_T1|SB\_dnorm\_T1|SB\_inorm\_T1|SB\_intention\_T1|SB\_outcomeExpectations\_T1', ci\_intervention$diamondlabels)) %>%   
 filter(diamondlabels != "PA\_goal\_T1") %>%   
 mutate("CI95" = paste(round(ciLo, 1), "-", round(ciHi, 1)),  
 mean = round(mean, 1))  
  
mediatortable\_intervention$diamondlabels <- gsub('PA\_actionplan\_T1', 'PA action planning', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_copingplan\_T1', 'PA coping planning', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_agrbct\_T1', 'PA agreement-BCTs', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_frqbct\_T1', 'PA frequency-BCTs', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_amotivation\_T1', 'PA amotivation', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_autonomous\_T1', 'PA autonomous regulation', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_controlled\_T1', 'PA controlled regulation', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_inorm\_T1', 'PA injunctive norm', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_dnorm\_T1', 'PA descriptive norm', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_intention\_T1', 'PA intention', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_outcomeExpectations\_T1', 'PA outcome expectations', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_opportunities\_T1', 'PA opportunities', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_pbc\_T1', 'PA perceived behavioural control', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('PA\_selfefficacy\_T1', 'PA self-efficacy', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('SB\_dnorm\_T1', 'SB descriptive norm', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('SB\_inorm\_T1', 'SB injunctive norm', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('SB\_intention\_T1', 'SB intention', mediatortable\_intervention$diamondlabels)  
mediatortable\_intervention$diamondlabels <- gsub('SB\_outcomeExpectations\_T1', 'SB outcome expectations', mediatortable\_intervention$diamondlabels)  
  
mediatortable\_control <- ci\_control %>%  
 dplyr::filter(grepl('PA\_actionplan\_T1|PA\_copingplan\_T1|PA\_agrbct\_T1|PA\_frqbct\_T1|PA\_amotivation\_T1|PA\_autonomous\_T1|PA\_controlled\_T1|PA\_goal\_T1|PA\_inorm\_T1|PA\_dnorm\_T1|PA\_intention\_T1|PA\_outcomeExpectations\_T1|PA\_opportunities\_T1|PA\_pbc\_T1|PA\_selfefficacy\_T1|SB\_dnorm\_T1|SB\_inorm\_T1|SB\_intention\_T1|SB\_outcomeExpectations\_T1', ci\_control$diamondlabels)) %>%   
 filter(diamondlabels != "PA\_goal\_T1") %>%   
 mutate("CI95" = paste(round(ciLo, 1), "-", round(ciHi, 1)),  
 mean = round(mean, 1))  
  
mediatortable\_control$diamondlabels <- gsub('PA\_actionplan\_T1', 'PA action planning', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_copingplan\_T1', 'PA coping planning', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_agrbct\_T1', 'PA agreement-BCTs', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_frqbct\_T1', 'PA frequency-BCTs', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_amotivation\_T1', 'PA amotivation', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_autonomous\_T1', 'PA autonomous regulation', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_controlled\_T1', 'PA controlled regulation', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_inorm\_T1', 'PA injunctive norm', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_dnorm\_T1', 'PA descriptive norm', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_intention\_T1', 'PA intention', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_outcomeExpectations\_T1', 'PA outcome expectations', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_opportunities\_T1', 'PA opportunities', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_pbc\_T1', 'PA perceived behavioural control', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('PA\_selfefficacy\_T1', 'PA self-efficacy', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('SB\_dnorm\_T1', 'SB descriptive norm', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('SB\_inorm\_T1', 'SB injunctive norm', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('SB\_intention\_T1', 'SB intention', mediatortable\_control$diamondlabels)  
mediatortable\_control$diamondlabels <- gsub('SB\_outcomeExpectations\_T1', 'SB outcome expectations', mediatortable\_control$diamondlabels)  
  
mediatortable\_girls <- ci\_girls %>%  
 dplyr::filter(grepl('PA\_actionplan\_T1|PA\_copingplan\_T1|PA\_agrbct\_T1|PA\_frqbct\_T1|PA\_amotivation\_T1|PA\_autonomous\_T1|PA\_controlled\_T1|PA\_goal\_T1|PA\_inorm\_T1|PA\_dnorm\_T1|PA\_intention\_T1|PA\_outcomeExpectations\_T1|PA\_opportunities\_T1|PA\_pbc\_T1|PA\_selfefficacy\_T1|SB\_dnorm\_T1|SB\_inorm\_T1|SB\_intention\_T1|SB\_outcomeExpectations\_T1', ci\_girls$diamondlabels)) %>%   
 filter(diamondlabels != "PA\_goal\_T1") %>%   
 mutate("CI95" = paste(round(ciLo, 1), "-", round(ciHi, 1)),  
 mean = round(mean, 1))  
  
mediatortable\_girls$diamondlabels <- gsub('PA\_actionplan\_T1', 'PA action planning', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_copingplan\_T1', 'PA coping planning', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_agrbct\_T1', 'PA agreement-BCTs', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_frqbct\_T1', 'PA frequency-BCTs', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_amotivation\_T1', 'PA amotivation', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_autonomous\_T1', 'PA autonomous regulation', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_controlled\_T1', 'PA controlled regulation', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_inorm\_T1', 'PA injunctive norm', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_dnorm\_T1', 'PA descriptive norm', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_intention\_T1', 'PA intention', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_outcomeExpectations\_T1', 'PA outcome expectations', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_opportunities\_T1', 'PA opportunities', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_pbc\_T1', 'PA perceived behavioural control', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('PA\_selfefficacy\_T1', 'PA self-efficacy', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('SB\_dnorm\_T1', 'SB descriptive norm', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('SB\_inorm\_T1', 'SB injunctive norm', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('SB\_intention\_T1', 'SB intention', mediatortable\_girls$diamondlabels)  
mediatortable\_girls$diamondlabels <- gsub('SB\_outcomeExpectations\_T1', 'SB outcome expectations', mediatortable\_girls$diamondlabels)  
  
mediatortable\_boys <- ci\_boys %>%  
 dplyr::filter(grepl('PA\_actionplan\_T1|PA\_copingplan\_T1|PA\_agrbct\_T1|PA\_frqbct\_T1|PA\_amotivation\_T1|PA\_autonomous\_T1|PA\_controlled\_T1|PA\_goal\_T1|PA\_inorm\_T1|PA\_dnorm\_T1|PA\_intention\_T1|PA\_outcomeExpectations\_T1|PA\_opportunities\_T1|PA\_pbc\_T1|PA\_selfefficacy\_T1|SB\_dnorm\_T1|SB\_inorm\_T1|SB\_intention\_T1|SB\_outcomeExpectations\_T1', ci\_boys$diamondlabels)) %>%   
 filter(diamondlabels != "PA\_goal\_T1") %>%   
 mutate("CI95" = paste(round(ciLo, 1), "-", round(ciHi, 1)),  
 mean = round(mean, 1))  
  
mediatortable\_boys$diamondlabels <- gsub('PA\_actionplan\_T1', 'PA action planning', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_copingplan\_T1', 'PA coping planning', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_agrbct\_T1', 'PA agreement-BCTs', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_frqbct\_T1', 'PA frequency-BCTs', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_amotivation\_T1', 'PA amotivation', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_autonomous\_T1', 'PA autonomous regulation', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_controlled\_T1', 'PA controlled regulation', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_inorm\_T1', 'PA injunctive norm', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_dnorm\_T1', 'PA descriptive norm', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_intention\_T1', 'PA intention', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_outcomeExpectations\_T1', 'PA outcome expectations', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_opportunities\_T1', 'PA opportunities', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_pbc\_T1', 'PA perceived behavioural control', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('PA\_selfefficacy\_T1', 'PA self-efficacy', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('SB\_dnorm\_T1', 'SB descriptive norm', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('SB\_inorm\_T1', 'SB injunctive norm', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('SB\_intention\_T1', 'SB intention', mediatortable\_boys$diamondlabels)  
mediatortable\_boys$diamondlabels <- gsub('SB\_outcomeExpectations\_T1', 'SB outcome expectations', mediatortable\_boys$diamondlabels)  
  
mediatortable\_total <- ci\_total %>%  
 dplyr::filter(grepl('PA\_actionplan\_T1|PA\_copingplan\_T1|PA\_agrbct\_T1|PA\_frqbct\_T1|PA\_amotivation\_T1|PA\_autonomous\_T1|PA\_controlled\_T1|PA\_goal\_T1|PA\_inorm\_T1|PA\_dnorm\_T1|PA\_intention\_T1|PA\_outcomeExpectations\_T1|PA\_opportunities\_T1|PA\_pbc\_T1|PA\_selfefficacy\_T1|SB\_dnorm\_T1|SB\_inorm\_T1|SB\_intention\_T1|SB\_outcomeExpectations\_T1', ci\_total$diamondlabels)) %>%   
 filter(diamondlabels != "PA\_goal\_T1") %>%   
 mutate("CI95" = paste(round(ciLo, 1), "-", round(ciHi, 1)),  
 mean = round(mean, 1))  
  
mediatortable\_total$diamondlabels <- gsub('PA\_actionplan\_T1', 'PA action planning', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_copingplan\_T1', 'PA coping planning', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_agrbct\_T1', 'PA agreement-BCTs', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_frqbct\_T1', 'PA frequency-BCTs', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_amotivation\_T1', 'PA amotivation', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_autonomous\_T1', 'PA autonomous regulation', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_controlled\_T1', 'PA controlled regulation', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_inorm\_T1', 'PA injunctive norm', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_dnorm\_T1', 'PA descriptive norm', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_intention\_T1', 'PA intention', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_outcomeExpectations\_T1', 'PA outcome expectations', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_opportunities\_T1', 'PA opportunities', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_pbc\_T1', 'PA perceived behavioural control', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('PA\_selfefficacy\_T1', 'PA self-efficacy', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('SB\_dnorm\_T1', 'SB descriptive norm', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('SB\_inorm\_T1', 'SB injunctive norm', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('SB\_intention\_T1', 'SB intention', mediatortable\_total$diamondlabels)  
mediatortable\_total$diamondlabels <- gsub('SB\_outcomeExpectations\_T1', 'SB outcome expectations', mediatortable\_total$diamondlabels)  
  
mega\_mediatortable <- data.frame(Variable = mediatortable\_girls$diamondlabels,   
 "Girls" = paste0(round(mediatortable\_girls$mean, 2), " (", mediatortable\_girls$CI95, ")"),  
 "Boys" = paste0(round(mediatortable\_boys$mean, 2), " (", mediatortable\_boys$CI95, ")"),  
 "Intervention" = paste0(round(mediatortable\_intervention$mean, 2), " (", mediatortable\_intervention$CI95, ")"),  
 "Control" = paste0(round(mediatortable\_control$mean, 2), " (", mediatortable\_control$CI95, ")"),  
 "Total" = paste0(round(mediatortable\_total$mean, 2), " (", mediatortable\_total$CI95, ")")) %>%   
 arrange(Variable)  
  
papaja::apa\_table(mega\_mediatortable, caption = "Main mediating variables of PA and SB")

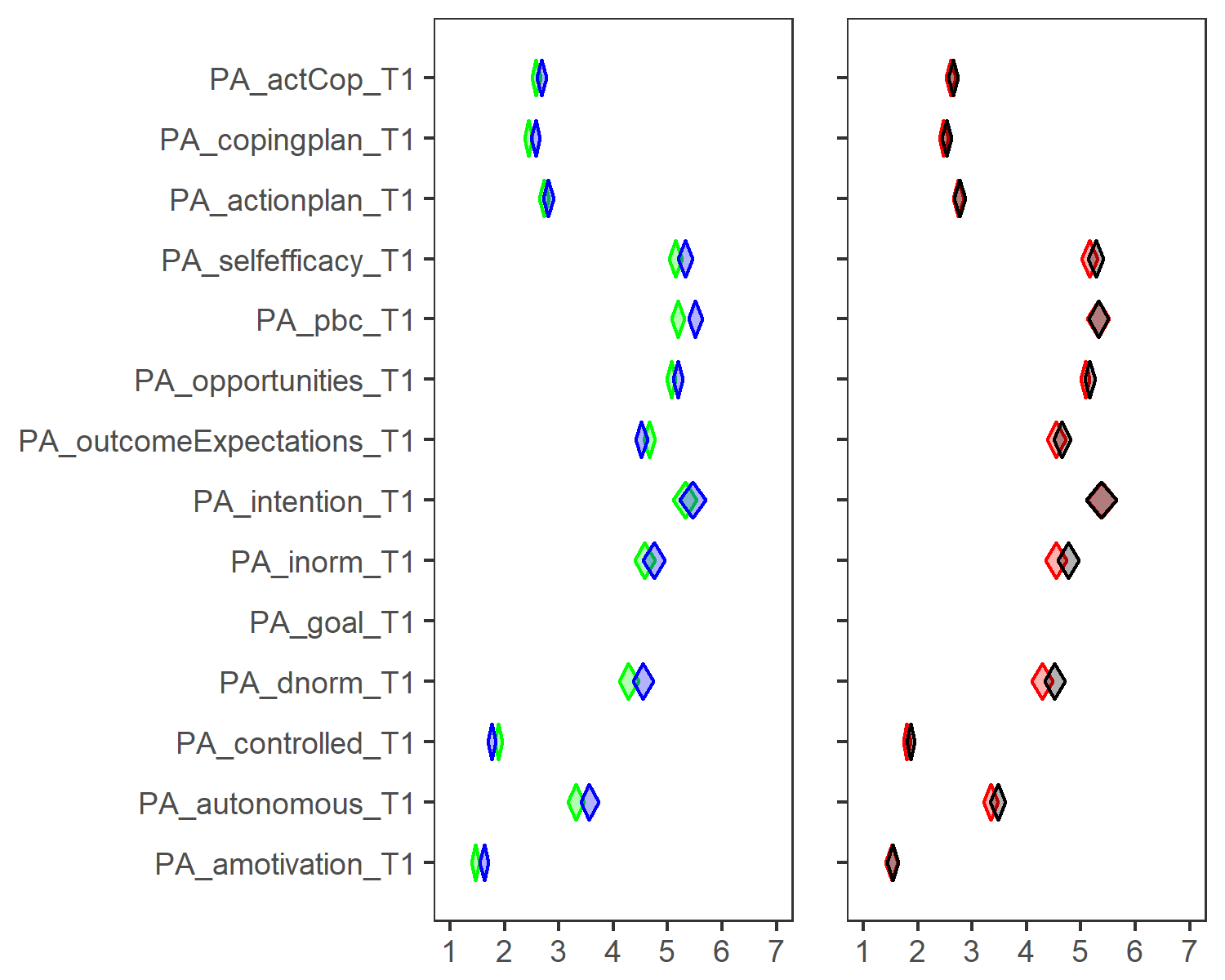
(#tab:outcome-table2)

*Main mediating variables of PA and SB*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Variable | Girls | Boys | Intervention | Control | Total |
| PA action planning | 2.7 (2.6 - 2.8) | 2.8 (2.7 - 2.9) | 2.7 (2.6 - 2.8) | 2.8 (2.7 - 2.9) | 2.8 (2.7 - 2.8) |
| PA agreement-BCTs | 3.1 (2.9 - 3.2) | 3.1 (3 - 3.3) | 3 (2.9 - 3.2) | 3.2 (3 - 3.4) | 3.1 (3 - 3.2) |
| PA amotivation | 1.5 (1.4 - 1.5) | 1.6 (1.5 - 1.7) | 1.5 (1.4 - 1.6) | 1.5 (1.4 - 1.7) | 1.5 (1.5 - 1.6) |
| PA autonomous regulation | 3.3 (3.2 - 3.5) | 3.6 (3.4 - 3.7) | 3.3 (3.2 - 3.5) | 3.5 (3.3 - 3.6) | 3.4 (3.3 - 3.5) |
| PA controlled regulation | 1.9 (1.8 - 2) | 1.8 (1.7 - 1.8) | 1.8 (1.7 - 1.9) | 1.9 (1.8 - 1.9) | 1.8 (1.8 - 1.9) |
| PA coping planning | 2.4 (2.4 - 2.5) | 2.6 (2.5 - 2.7) | 2.5 (2.4 - 2.6) | 2.5 (2.4 - 2.6) | 2.5 (2.4 - 2.6) |
| PA descriptive norm | 4.3 (4.1 - 4.5) | 4.5 (4.4 - 4.7) | 4.3 (4.1 - 4.5) | 4.5 (4.3 - 4.7) | 4.4 (4.2 - 4.6) |
| PA frequency-BCTs | 2.5 (2.4 - 2.6) | 2.6 (2.5 - 2.7) | 2.5 (2.4 - 2.6) | 2.6 (2.4 - 2.7) | 2.5 (2.4 - 2.6) |
| PA injunctive norm | 4.6 (4.4 - 4.8) | 4.8 (4.5 - 5) | 4.5 (4.3 - 4.7) | 4.8 (4.6 - 5) | 4.7 (4.5 - 4.8) |
| PA intention | 5.3 (5.1 - 5.5) | 5.5 (5.2 - 5.7) | 5.4 (5.1 - 5.7) | 5.4 (5.1 - 5.7) | 5.4 (5.2 - 5.6) |
| PA opportunities | 5.1 (5 - 5.1) | 5.2 (5.1 - 5.3) | 5.1 (5 - 5.2) | 5.2 (5.1 - 5.3) | 5.1 (5.1 - 5.2) |
| PA outcome expectations | 4.7 (4.6 - 4.8) | 4.5 (4.4 - 4.6) | 4.6 (4.4 - 4.7) | 4.7 (4.5 - 4.8) | 4.6 (4.5 - 4.7) |
| PA perceived behavioural control | 5.2 (5.1 - 5.3) | 5.5 (5.4 - 5.6) | 5.3 (5.1 - 5.5) | 5.3 (5.1 - 5.5) | 5.3 (5.2 - 5.5) |
| PA self-efficacy | 5.1 (5 - 5.3) | 5.3 (5.2 - 5.5) | 5.2 (5 - 5.3) | 5.3 (5.1 - 5.4) | 5.2 (5.1 - 5.4) |
| SB descriptive norm | 3.2 (3 - 3.4) | 3.4 (3.1 - 3.6) | 3.2 (3 - 3.4) | 3.3 (3.1 - 3.5) | 3.2 (3.1 - 3.4) |
| SB injunctive norm | 4 (3.8 - 4.1) | 4.1 (3.9 - 4.3) | 3.9 (3.8 - 4.1) | 4.1 (4 - 4.2) | 4 (3.9 - 4.1) |
| SB intention | 3.8 (3.5 - 4.1) | 3.6 (3.3 - 3.9) | 3.7 (3.2 - 4.2) | 3.7 (3.3 - 4.2) | 3.7 (3.4 - 4.1) |
| SB outcome expectations | 4.5 (4.4 - 4.6) | 4.3 (4.2 - 4.4) | 4.4 (4.2 - 4.6) | 4.4 (4.3 - 4.6) | 4.4 (4.3 - 4.5) |

# PA psychosocial determinants: self report scales T1

PA\_ci\_girls <- ci\_girls %>% filter(diamondlabels %in% names(scales\_T1) & grepl("PA\_", diamondlabels) & !grepl("bct", diamondlabels))  
  
PA\_ci\_boys <- ci\_boys %>% filter(diamondlabels %in% names(scales\_T1) & grepl("PA\_", diamondlabels) & !grepl("bct", diamondlabels))  
  
PA\_ci\_intervention <- ci\_intervention %>% filter(diamondlabels %in% names(scales\_T1) & grepl("PA\_", diamondlabels) & !grepl("bct", diamondlabels))  
  
PA\_ci\_control <- ci\_control %>% filter(diamondlabels %in% names(scales\_T1) & grepl("PA\_", diamondlabels) & !grepl("bct", diamondlabels))  
  
ICClabels <- vardatatable\_containing\_edutrack %>% filter(Variable %in% PA\_ci\_control$diamondlabels)  
  
library(ggplot2)  
plot1 <- userfriendlyscience::diamondPlot(PA\_ci\_girls, color = 'green', alpha=.3, yLabels = PA\_ci\_girls$diamondlabels, fixedSize = 0.3, xlab = NULL) +  
 userfriendlyscience::diamondPlot(PA\_ci\_boys, returnLayerOnly = TRUE, color='blue', alpha=.3, fixedSize = 0.3) +  
 scale\_x\_continuous(limits = c(1, 7), breaks = 1:7)  
  
plot2 <- userfriendlyscience::diamondPlot(PA\_ci\_intervention, color = 'red', alpha=.3, yLabels = c(rep("", length(PA\_ci\_girls$diamondlabels))), fixedSize = 0.3, xlab = NULL, ylab = NULL) +  
 userfriendlyscience::diamondPlot(PA\_ci\_control, returnLayerOnly = TRUE, color='black', alpha=.3, fixedSize = 0.3) +  
 scale\_x\_continuous(limits = c(1, 7), breaks = 1:7)  
  
grid::grid.newpage()  
grid::grid.draw(cbind(ggplotGrob(plot1), ggplotGrob(plot2), size = "first"))



# dat2 <- data.frame(ciLo = c(3, 2), mean = c(4, 2.5), ciHi = c(6, 3));  
# userfriendlyscience::diamondPlot(dat1, color = 'blue', alpha=.3, yLabels = dat1$diamondlabels) +  
# userfriendlyscience::diamondPlot(dat2, returnLayerOnly = TRUE, color='red', alpha=.3)

Note: action and coping planning on a scale from 1 to 4.

# Visualising random effects using full posterior (UNDER CONSTRUCTION)

## Using rstanarm

Plot below shows credible intervals deviations

library(rstanarm)  
# grep("PA\_", names(df), value = TRUE)  
  
# Centre the variable:  
df <- df %>% mutate(paAccelerometer\_T1\_centred = paAccelerometer\_T1 - mean(paAccelerometer\_T1, na.rm = TRUE))  
  
m\_1 <- stan\_glmer(paAccelerometer\_T1\_centred ~ (1 | group), data = df, chains = 2, iter = 2000)   
##   
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).  
##   
## Gradient evaluation took 0.001 seconds  
## 1000 transitions using 10 leapfrog steps per transition would take 10 seconds.  
## Adjust your expectations accordingly!  
##   
##   
## Iteration: 1 / 2000 [ 0%] (Warmup)  
## Iteration: 200 / 2000 [ 10%] (Warmup)  
## Iteration: 400 / 2000 [ 20%] (Warmup)  
## Iteration: 600 / 2000 [ 30%] (Warmup)  
## Iteration: 800 / 2000 [ 40%] (Warmup)  
## Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Iteration: 2000 / 2000 [100%] (Sampling)  
##   
## Elapsed Time: 1.902 seconds (Warm-up)  
## 1.001 seconds (Sampling)  
## 2.903 seconds (Total)  
##   
##   
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).  
##   
## Gradient evaluation took 0 seconds  
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Adjust your expectations accordingly!  
##   
##   
## Iteration: 1 / 2000 [ 0%] (Warmup)  
## Iteration: 200 / 2000 [ 10%] (Warmup)  
## Iteration: 400 / 2000 [ 20%] (Warmup)  
## Iteration: 600 / 2000 [ 30%] (Warmup)  
## Iteration: 800 / 2000 [ 40%] (Warmup)  
## Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Iteration: 2000 / 2000 [100%] (Sampling)  
##   
## Elapsed Time: 1.836 seconds (Warm-up)  
## 1.121 seconds (Sampling)  
## 2.957 seconds (Total)  
pt <- ranef(m\_1)   
posteriors <- data.frame(posterior\_interval(m\_1, prob = 0.90))   
school\_int <- posteriors[-c(1, nrow(posteriors)-1, nrow(posteriors)),] #remove intercept and sigma parameters, leaving only class intervals   
  
# This not needed any more:  
# groupmeans <- df %>% select(paAccelerometer\_T1, group) %>% group\_by(group) %>%   
# summarise(mean = mean(paAccelerometer\_T1, na.rm = T)) %>% filter(complete.cases(.))  
  
dat\_plt <- data.frame(pt$group, school\_int)   
colnames(dat\_plt) <- c("pt", "low", "up")   
dat\_plt <- dat\_plt %>% arrange(pt)   
dat\_plt$index <- 1:nrow(dat\_plt)   
dat\_plt %>% ggplot(aes(x = index, y = pt)) + geom\_errorbar(aes(ymin = low, ymax = up)) + geom\_point(size = 4, shape = 18) + geom\_point(size = 2.5, shape = 18, color = "white") + theme\_bw()



## Using brms and diamondPlot

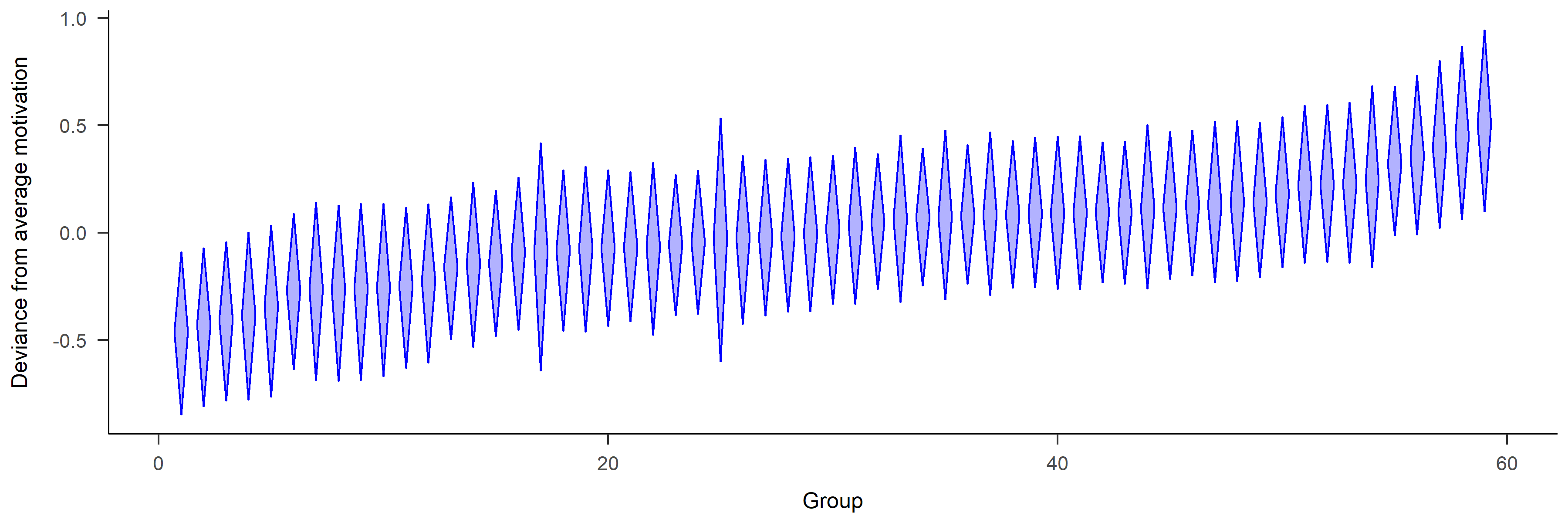
Set Rtools folder, if not in C:/Rtools/:

# For some reason, having BINPREF in .Renviron doesn't work and I keep needing this:  
Sys.setenv("BINPREF" = "C:/HYApp/Rtools3.4/mingw\_$(WIN)/bin/")

library(brms)  
  
# Centre the variable:  
df <- df %>% mutate(paAccelerometer\_T1\_centred =   
 paAccelerometer\_T1 - mean(paAccelerometer\_T1, na.rm = TRUE))  
  
m\_1 <- brms::bf(paAccelerometer\_T1\_centred ~ (1 | group))   
get\_prior(m\_1, data = df)  
  
fit\_1 <- brms::brm(m\_1, df)  
  
pt <- brms::ranef(fit\_1, robust = TRUE) %>%   
 data.frame %>%   
 select(credintLow = group.2.5.ile.Intercept,  
 intercept = group.Estimate.Intercept,   
 credintHigh = group.97.5.ile.Intercept) %>%   
 arrange(intercept)  
  
  
plot1 <- userfriendlyscience::diamondPlot(pt, color = 'green', alpha=.3, yLabels = 1:58, fixedSize = 0.3, xlab = NULL) +  
 coord\_flip(expand = TRUE) +  
 labs(x = "Deviance from average MVPA", y = "Group")  
  
  
df %>% ggplot(aes(x = group, y = paAccelerometer\_T1\_centred)) +  
 userfriendlyscience::diamondPlot(pt, color = 'green', alpha=.3, yLabels = 1:58, fixedSize = 0.3, xlab = NULL, returnLayerOnly = TRUE) +  
 coord\_flip(expand = TRUE) +  
 labs(x = "Deviance from average MVPA", y = "Group")  
   
forest(fit\_1)

## Random effects in autonomous motivation

# Centre the variable:  
df <- df %>% mutate(PA\_autonomous\_T1\_centred = PA\_autonomous\_T1 - mean(PA\_autonomous\_T1, na.rm = TRUE))  
  
m\_1 <- brms::bf(PA\_autonomous\_T1\_centred ~ (1 | group))   
get\_prior(m\_1, data = df)  
## Error in get\_prior(m\_1, data = df): could not find function "get\_prior"  
  
fit\_1 <- brms::brm(m\_1, df)  
## Warning: Rows containing NAs were excluded from the model  
## Compiling the C++ model  
## Start sampling  
##   
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 1).  
##   
## Gradient evaluation took 0 seconds  
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Adjust your expectations accordingly!  
##   
##   
## Iteration: 1 / 2000 [ 0%] (Warmup)  
## Iteration: 200 / 2000 [ 10%] (Warmup)  
## Iteration: 400 / 2000 [ 20%] (Warmup)  
## Iteration: 600 / 2000 [ 30%] (Warmup)  
## Iteration: 800 / 2000 [ 40%] (Warmup)  
## Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Iteration: 2000 / 2000 [100%] (Sampling)  
##   
## Elapsed Time: 1.341 seconds (Warm-up)  
## 0.971 seconds (Sampling)  
## 2.312 seconds (Total)  
##   
##   
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 2).  
##   
## Gradient evaluation took 0 seconds  
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Adjust your expectations accordingly!  
##   
##   
## Iteration: 1 / 2000 [ 0%] (Warmup)  
## Iteration: 200 / 2000 [ 10%] (Warmup)  
## Iteration: 400 / 2000 [ 20%] (Warmup)  
## Iteration: 600 / 2000 [ 30%] (Warmup)  
## Iteration: 800 / 2000 [ 40%] (Warmup)  
## Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Iteration: 2000 / 2000 [100%] (Sampling)  
##   
## Elapsed Time: 1.351 seconds (Warm-up)  
## 1.321 seconds (Sampling)  
## 2.672 seconds (Total)  
##   
##   
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 3).  
##   
## Gradient evaluation took 0 seconds  
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Adjust your expectations accordingly!  
##   
##   
## Iteration: 1 / 2000 [ 0%] (Warmup)  
## Iteration: 200 / 2000 [ 10%] (Warmup)  
## Iteration: 400 / 2000 [ 20%] (Warmup)  
## Iteration: 600 / 2000 [ 30%] (Warmup)  
## Iteration: 800 / 2000 [ 40%] (Warmup)  
## Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Iteration: 2000 / 2000 [100%] (Sampling)  
##   
## Elapsed Time: 1.473 seconds (Warm-up)  
## 1.35 seconds (Sampling)  
## 2.823 seconds (Total)  
##   
##   
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 4).  
##   
## Gradient evaluation took 0 seconds  
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Adjust your expectations accordingly!  
##   
##   
## Iteration: 1 / 2000 [ 0%] (Warmup)  
## Iteration: 200 / 2000 [ 10%] (Warmup)  
## Iteration: 400 / 2000 [ 20%] (Warmup)  
## Iteration: 600 / 2000 [ 30%] (Warmup)  
## Iteration: 800 / 2000 [ 40%] (Warmup)  
## Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Iteration: 2000 / 2000 [100%] (Sampling)  
##   
## Elapsed Time: 1.491 seconds (Warm-up)  
## 1.309 seconds (Sampling)  
## 2.8 seconds (Total)  
  
pt <- brms::ranef(fit\_1, robust = TRUE) %>%   
 data.frame %>%   
 select(credintLow = group.2.5.ile.Intercept,  
 intercept = group.Estimate.Intercept,   
 credintHigh = group.97.5.ile.Intercept) %>%   
 arrange(intercept)  
  
  
plot1 <- userfriendlyscience::diamondPlot(pt, color = 'blue', alpha=.3, yLabels = 1:59, fixedSize = 0.3, xlab = NULL) +  
 coord\_flip(expand = TRUE) +  
 labs(x = "Deviance from average motivation", y = "Group")  
  
  
df %>% ggplot(aes(x = group, y = PA\_autonomous\_T1\_centred)) +  
 userfriendlyscience::diamondPlot(pt, color = 'blue', alpha=.3, yLabels = 1:59, fixedSize = 0.3, xlab = NULL, returnLayerOnly = TRUE) +  
 coord\_flip(expand = TRUE) +  
 labs(x = "Deviance from average motivation", y = "Group")



# Ridge plots

## PA motivation

plot1 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 'PA autonomous \nmotivation' = PA\_autonomous\_T1,  
'PA controlled \nmotivation' = PA\_controlled\_T1,  
'PA amotivation' = PA\_amotivation\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%   
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, fill = paste(Variable, girl)),   
 alpha = .6, color = "black", from = 1, to = 7, scale = 1) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0)) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("PA amotivation Boy", "PA amotivation Girl"),  
 labels = c( 'PA amotivation Girl' = "Boy", 'PA amotivation Boy' = "Girl"),  
 values = c("#3bc600", "#0000ff", "#9cc68b", "#8080ff"),  
 name = "", guide = "legend") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom")  
  
plot2 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 'PA autonomous \nmotivation' = PA\_autonomous\_T1,  
'PA controlled \nmotivation' = PA\_controlled\_T1,  
'PA amotivation' = PA\_amotivation\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%   
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, fill = paste(Variable, intervention)),   
 alpha = .6, color = "black", from = 1, to = 7, scale = 1) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), labels = NULL) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("PA amotivation 1", "PA amotivation 0"),  
 labels = c('PA amotivation 1' = "Intervention", 'PA amotivation 0' = "Control"),  
 values = c("#ff0000", "#0000ff", "#ff8080", "#8080ff"),  
 name = "", guide = "legend") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom")  
  
# grid.arrange(plot1, plot2, ncol = 2)  
  
grid::grid.newpage()  
grid::grid.draw(cbind(ggplotGrob(plot1), ggplotGrob(plot2), size = "first"))  
## Picking joint bandwidth of 0.216  
## Warning: Removed 272 rows containing non-finite values  
## (stat\_density\_ridges).  
## Picking joint bandwidth of 0.216  
## Warning: Removed 272 rows containing non-finite values  
## (stat\_density\_ridges).  
## Error: nrow(x) == nrow(y) is not TRUE

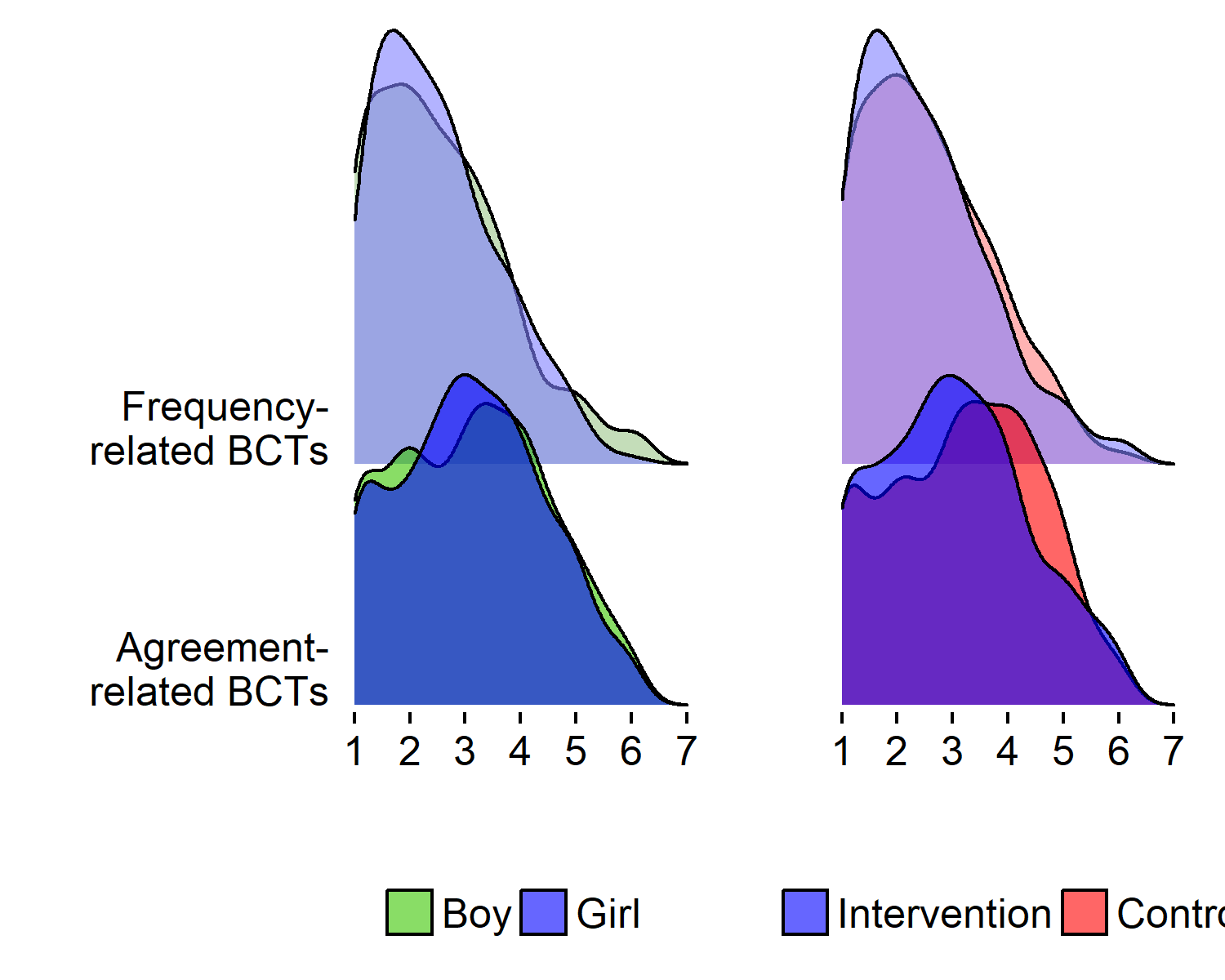
## Big 5 personality traits

plot1 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
'Agreeableness' = big5\_agreeableness\_T1,  
'Conscientiousness' = big5\_conscientiousness\_T1,  
'Extraversion' = big5\_extraversion\_T1,  
'Neuroticism' = big5\_neuroticism\_T1,  
'Openness' = big5\_openness\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%   
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, fill = paste(Variable, girl)),   
 alpha = .6, color = "black", from = 1, to = 7) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0)) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("Agreeableness 0", "Agreeableness 1"),  
 labels = c( 'Agreeableness 0' = "Boy", 'Agreeableness 1' = "Girl"),  
 values = c("#3bc600", "#0000ff", "#9cc68b", "#8080ff"),  
 name = "", guide = "legend") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom")  
## Error in overscope\_eval\_next(overscope, expr): object 'big5\_agreeableness\_T1' not found  
  
plot2 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
'Agreeableness' = big5\_agreeableness\_T1,  
'Conscientiousness' = big5\_conscientiousness\_T1,  
'Extraversion' = big5\_extraversion\_T1,  
'Neuroticism' = big5\_neuroticism\_T1,  
'Openness' = big5\_openness\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%   
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, fill = paste(Variable, intervention)),   
 alpha = .6, color = "black", from = 1, to = 7) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), labels = NULL) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("Agreeableness 1", "Agreeableness 0"),  
 labels = c('Agreeableness 1' = "Intervention", 'Agreeableness 0' = "Control"),  
 values = c("#ff0000", "#0000ff", "#ff8080", "#8080ff"),  
 name = "", guide = "legend") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom")  
## Error in overscope\_eval\_next(overscope, expr): object 'big5\_agreeableness\_T1' not found  
  
# grid.arrange(plot1, plot2, ncol = 2)  
  
grid::grid.newpage()  
grid::grid.draw(cbind(ggplotGrob(plot1), ggplotGrob(plot2), size = "first"))  
## Picking joint bandwidth of 0.216  
## Warning: Removed 272 rows containing non-finite values  
## (stat\_density\_ridges).  
## Picking joint bandwidth of 0.216  
## Warning: Removed 272 rows containing non-finite values  
## (stat\_density\_ridges).  
## Error: nrow(x) == nrow(y) is not TRUE

## BCT use

### Ridge plots for means

plot1 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
'Frequency-\nrelated BCTs' = PA\_frqbct\_T1,  
'Agreement-\nrelated BCTs' = PA\_agrbct\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%   
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, fill = paste(Variable, girl)),   
 alpha = .6, color = "black", from = 1, to = 7) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0)) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("Agreement-\nrelated BCTs 0", "Agreement-\nrelated BCTs 1"),  
 labels = c( 'Agreement-\nrelated BCTs 0' = "Boy", 'Agreement-\nrelated BCTs 1' = "Girl"),  
 values = c("#3bc600", "#0000ff", "#9cc68b", "#8080ff"),  
 name = "", guide = "legend") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom")  
  
plot2 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
'Frequency-\nrelated BCTs' = PA\_frqbct\_T1,  
'Agreement-\nrelated BCTs' = PA\_agrbct\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%   
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, fill = paste(Variable, intervention)),   
 alpha = .6, color = "black", from = 1, to = 7) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), labels = NULL) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("Agreement-\nrelated BCTs 1", "Agreement-\nrelated BCTs 0"),  
 labels = c('Agreement-\nrelated BCTs 1' = "Intervention", 'Agreement-\nrelated BCTs 0' = "Control"),  
 values = c("#ff0000", "#0000ff", "#ff8080", "#8080ff"),  
 name = "", guide = "legend") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom")  
  
grid::grid.newpage()  
grid::grid.draw(cbind(ggplotGrob(plot1), ggplotGrob(plot2), size = "first"))  
## Picking joint bandwidth of 0.319  
## Warning: Removed 189 rows containing non-finite values  
## (stat\_density\_ridges).  
## Picking joint bandwidth of 0.318  
## Warning: Removed 189 rows containing non-finite values  
## (stat\_density\_ridges).

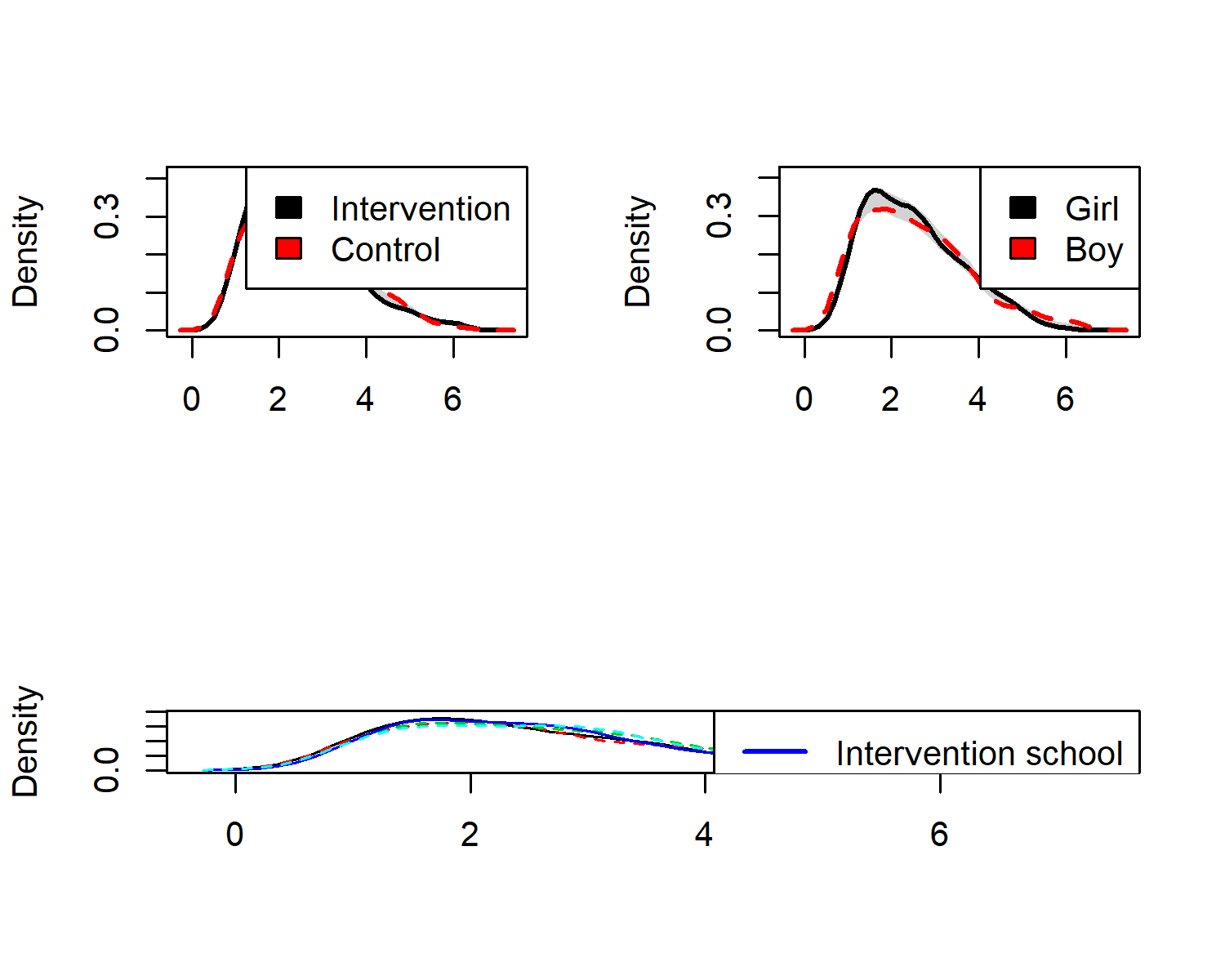


# grid.arrange(plot1, plot2, ncol = 2)

### sm kernel density plots for means

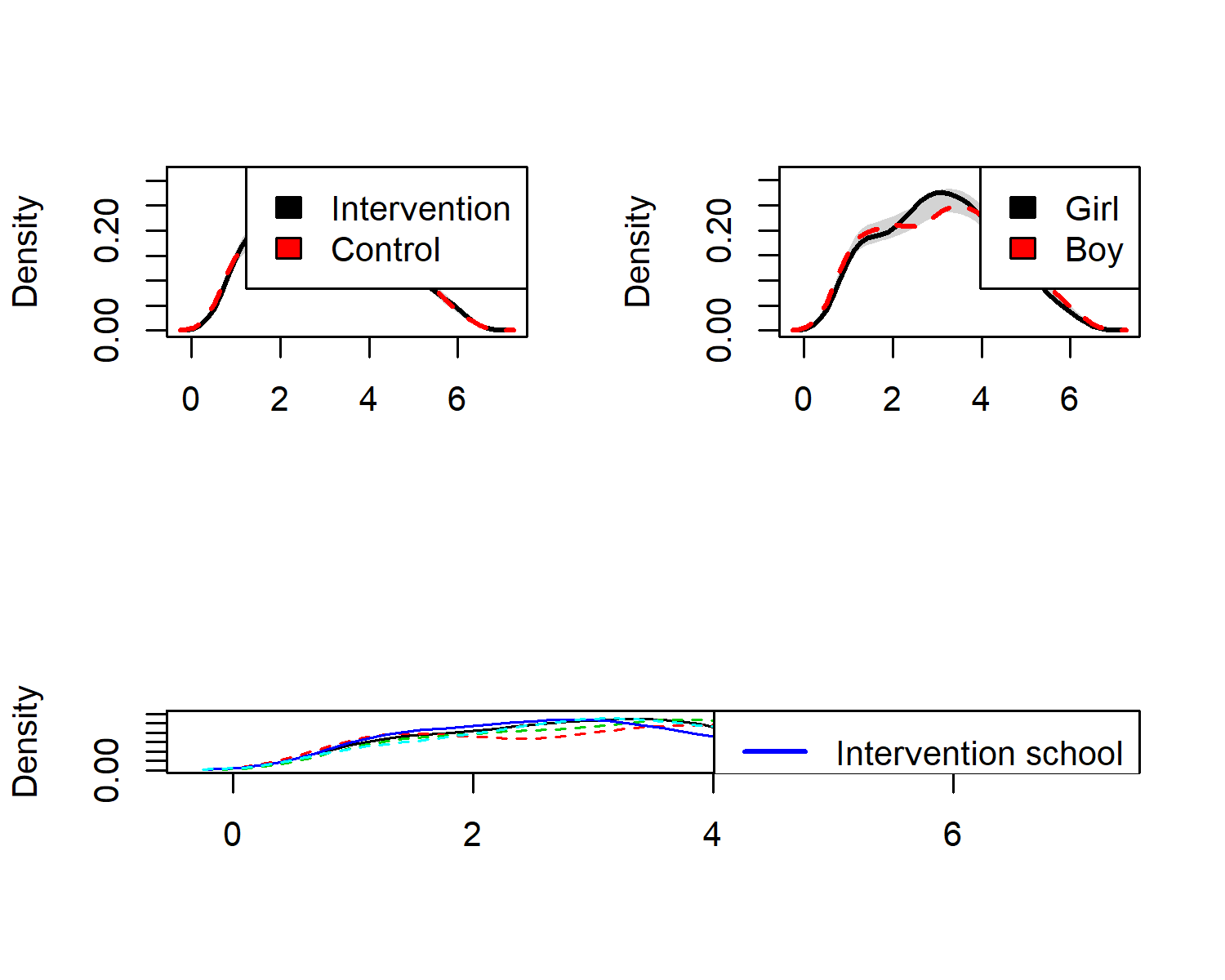
#### Frequency-measured BCTs

colfill <- c(1,2,3,4,5)  
  
# Create data frame  
densplot <- df  
levels(densplot$intervention) <- list("Intervention" = "1", "Control" = "0")  
levels(densplot$girl) <- list("Girl" = "1", "Boy" = "0")  
  
# This gives side-by-side plots  
 layout(matrix(c(1,2,3,3), nrow = 2, ncol = 2, byrow = TRUE), heights=c(2.5, 2))  
  
## Intervention vs. control  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(PA\_frqbct\_T1, intervention) %>%   
 na.omit(.)  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
# Make plot  
sm.PA\_frqbct\_T1\_1 <- sm.density.compare2(as.numeric(dens$PA\_frqbct\_T1), as.factor(dens$intervention), xlab="", col=colfill, lty=c(1,2), bandcol='LightGray', model="equal", lwd=(c(2,2)))  
##   
## Test of equal densities: p-value = 0.31  
legend("topright", levels(dens$intervention), fill=c(1, 2))  
  
## Girls vs. boys  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(PA\_frqbct\_T1, girl) %>%   
 na.omit(.)  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
# Make plot  
sm.PA\_frqbct\_T1\_2 <- sm.density.compare2(as.numeric(dens$PA\_frqbct\_T1), as.factor(dens$girl), xlab="", col=colfill, lty=c(1,2), bandcol='LightGray', model="equal", lwd=(c(2,2)))  
##   
## Test of equal densities: p-value = 0.11  
legend("topright", levels(dens$girl), fill=c(1, 2))  
  
## Differences in schools  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(PA\_frqbct\_T1, school) %>%   
 na.omit(.)  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
  
# Intervention groups with solid thicker lines  
linetypes <- c(1,2,2,1,2)  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
# Make plot  
sm.PA\_frqbct\_T1\_3 <- sm::sm.density.compare(as.numeric(dens$PA\_frqbct\_T1), dens$school, xlab="", col=colfill, lty=linetypes, model="none")  
legend("topright", c("Intervention school", "Control school"), col = c(4, 3), lty = c(1, 2), lwd = c(2, 2))



#### Agreement-measured BCTs

colfill <- c(1,2,3,4,5)  
  
# Create data frame  
densplot <- df  
levels(densplot$intervention) <- list("Intervention" = "1", "Control" = "0")  
levels(densplot$girl) <- list("Girl" = "1", "Boy" = "0")  
  
# This gives side-by-side plots  
 layout(matrix(c(1,2,3,3), nrow = 2, ncol = 2, byrow = TRUE), heights=c(2.5, 2))  
  
## Intervention vs. control  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(PA\_agrbct\_T1, intervention) %>%   
 na.omit(.)  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
# Make plot  
sm.PA\_agrbct\_T1\_1 <- sm.density.compare2(as.numeric(dens$PA\_agrbct\_T1), as.factor(dens$intervention), xlab="", col=colfill, lty=c(1,2), bandcol='LightGray', model="equal", lwd=(c(2,2)))  
##   
## Test of equal densities: p-value = 0  
legend("topright", levels(dens$intervention), fill=c(1, 2))  
  
## Girls vs. boys  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(PA\_agrbct\_T1, girl) %>%   
 na.omit(.)  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
# Make plot  
sm.PA\_agrbct\_T1\_2 <- sm.density.compare2(as.numeric(dens$PA\_agrbct\_T1), as.factor(dens$girl), xlab="", col=colfill, lty=c(1,2), bandcol='LightGray', model="equal", lwd=(c(2,2)))  
##   
## Test of equal densities: p-value = 0.27  
legend("topright", levels(dens$girl), fill=c(1, 2))  
  
## Differences in schools  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(PA\_agrbct\_T1, school) %>%   
 na.omit(.)  
  
# Intervention groups with solid thicker lines  
linetypes <- c(1,2,2,1,2)  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
# Make plot  
sm.PA\_agrbct\_T1\_3 <- sm::sm.density.compare(as.numeric(dens$PA\_agrbct\_T1), dens$school, xlab="", col=colfill, lty=linetypes, model="none")  
legend("topright", c("Intervention school", "Control school"), col = c(4, 3), lty = c(1, 2), lwd = c(2, 2))



### Histograms for individual items

#### Frequency-measured BCTs

These questions were asked with the lead “Have you done the following during the last three weeks?”.

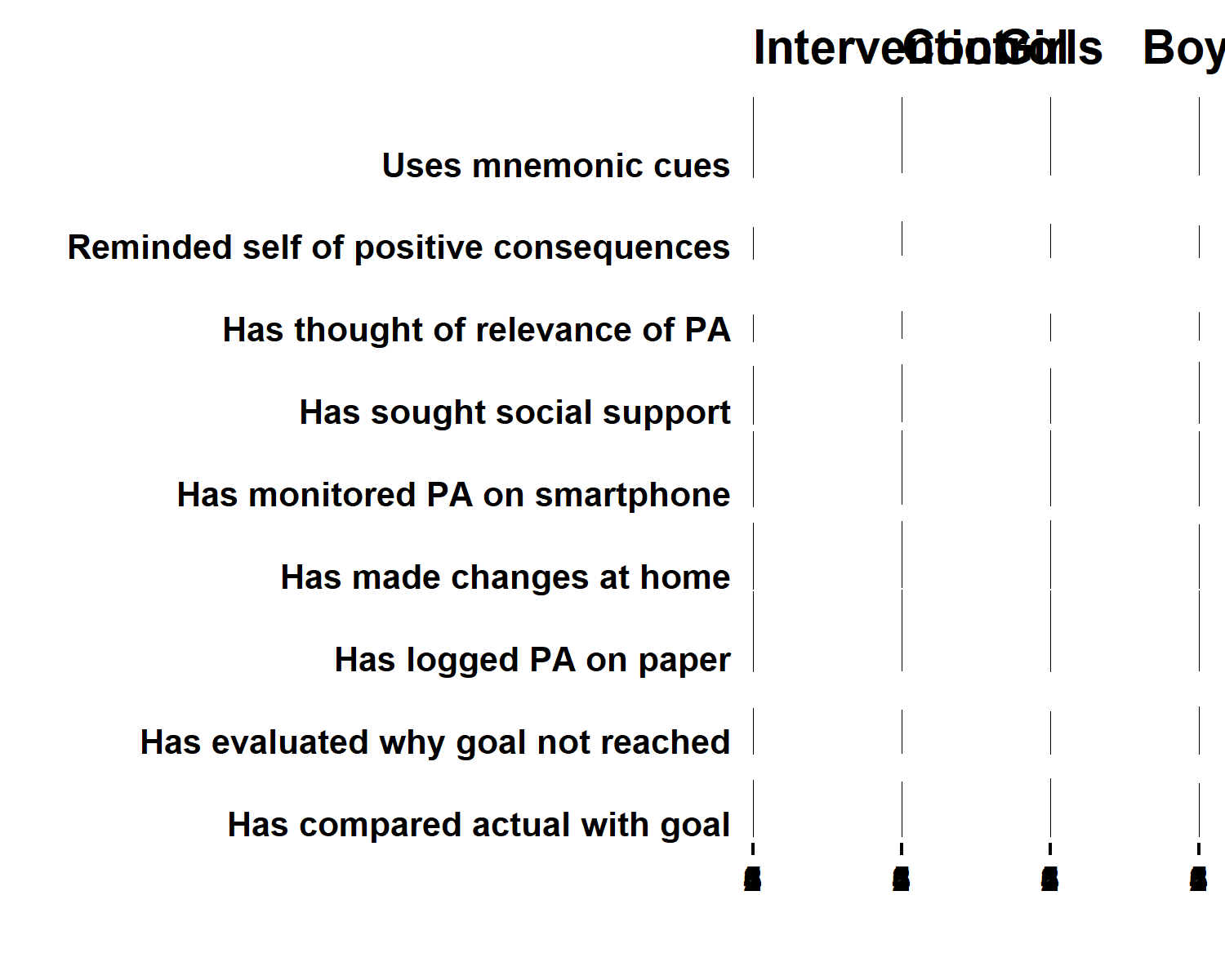
The answer scale was as follows:

0 = not once 1 = once 2 = twice 3 = weekly 4 = about every second day 5 = daily

Items are as follows:

1. I have reminded myself even in my spare time, what kind of positive consequences frequent PA would have in my life.
2. I have monitored my PA by marking the PA occasions on an exercise log on paper.
3. I have monitored my PA by using a smart phone, e.g. the Moves-app.
4. I use mnemonic cues with which I remember to implement my PA intention.
5. I have compared my actualized PA with the PA goal I have set.
6. I have thought about which reasons to do PA are important to me personally.
7. I have made changes in my home (e.g. my room or my computer), so that starting PA would be easier.
8. I have asked my friends or family for support to reach my PA goals.
9. If I haven’t reached my PA goal, I have evaluated, what went wrong.

bctGirls <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
'Reminded self of positive consequences' = PA\_frqbct\_01\_T1,  
'Has logged PA on paper' = PA\_frqbct\_02\_T1,  
'Has monitored PA on smartphone' = PA\_frqbct\_03\_T1,  
'Uses mnemonic cues' = PA\_frqbct\_04\_T1,  
'Has compared actual with goal' = PA\_frqbct\_05\_T1,  
'Has thought of relevance of PA' = PA\_frqbct\_06\_T1,  
'Has made changes at home' = PA\_frqbct\_07\_T1,  
'Has sought social support' = PA\_frqbct\_08\_T1,  
'Has evaluated why goal not reached' = PA\_frqbct\_09\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(girl == "1") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:6), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "", labels = NULL) +  
 ggridges::scale\_fill\_cyclical(values = c("darkolivegreen2", "darkolivegreen4")) +  
 labs(title = "Girls") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 plot.title = element\_text(hjust = 0.5),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 6.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
bctBoys <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
'Reminded self of positive consequences' = PA\_frqbct\_01\_T1,  
'Has logged PA on paper' = PA\_frqbct\_02\_T1,  
'Has monitored PA on smartphone' = PA\_frqbct\_03\_T1,  
'Uses mnemonic cues' = PA\_frqbct\_04\_T1,  
'Has compared actual with goal' = PA\_frqbct\_05\_T1,  
'Has thought of relevance of PA' = PA\_frqbct\_06\_T1,  
'Has made changes at home' = PA\_frqbct\_07\_T1,  
'Has sought social support' = PA\_frqbct\_08\_T1,  
'Has evaluated why goal not reached' = PA\_frqbct\_09\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(girl == "0") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:6), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "", labels = NULL) +  
 ggridges::scale\_fill\_cyclical(values = c("darkolivegreen2", "darkolivegreen4")) +  
 labs(title = "Boys") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 plot.title = element\_text(hjust = 0.5),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 6.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
bctInt <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
'Reminded self of positive consequences' = PA\_frqbct\_01\_T1,  
'Has logged PA on paper' = PA\_frqbct\_02\_T1,  
'Has monitored PA on smartphone' = PA\_frqbct\_03\_T1,  
'Uses mnemonic cues' = PA\_frqbct\_04\_T1,  
'Has compared actual with goal' = PA\_frqbct\_05\_T1,  
'Has thought of relevance of PA' = PA\_frqbct\_06\_T1,  
'Has made changes at home' = PA\_frqbct\_07\_T1,  
'Has sought social support' = PA\_frqbct\_08\_T1,  
'Has evaluated why goal not reached' = PA\_frqbct\_09\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(intervention == "1") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:6), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "") +  
 ggridges::scale\_fill\_cyclical(values = c("deepskyblue", "deepskyblue4")) +  
 labs(title = "Intervention") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 6.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
bctCont <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
'Reminded self of positive consequences' = PA\_frqbct\_01\_T1,  
'Has logged PA on paper' = PA\_frqbct\_02\_T1,  
'Has monitored PA on smartphone' = PA\_frqbct\_03\_T1,  
'Uses mnemonic cues' = PA\_frqbct\_04\_T1,  
'Has compared actual with goal' = PA\_frqbct\_05\_T1,  
'Has thought of relevance of PA' = PA\_frqbct\_06\_T1,  
'Has made changes at home' = PA\_frqbct\_07\_T1,  
'Has sought social support' = PA\_frqbct\_08\_T1,  
'Has evaluated why goal not reached' = PA\_frqbct\_09\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(intervention == "0") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:6), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "", labels = NULL) +  
 ggridges::scale\_fill\_cyclical(values = c("deepskyblue", "deepskyblue4")) +  
 labs(title = "Control") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 6.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
#grid.arrange(bctInt, bctGirls, bctCont, bctBoys, ncol = 2)  
  
# ("Seldom or never", "About once a month", "About once a week", "Almost daily")  
  
# This draws all histograms next to each other:  
grid::grid.newpage()  
grid::grid.draw(cbind(ggplotGrob(bctInt), ggplotGrob(bctCont), ggplotGrob(bctGirls), ggplotGrob(bctBoys), size = "last"))  
## Warning: Removed 54 rows containing non-finite values (stat\_binline).  
## Warning: Removed 72 rows containing non-finite values (stat\_binline).  
## Warning: Removed 54 rows containing non-finite values (stat\_binline).  
## Warning: Removed 72 rows containing non-finite values (stat\_binline).



# This draws 2 histograms per row:  
# grid.newpage()  
# grid.draw(rbind(cbind(ggplotGrob(bctInt), ggplotGrob(bctCont), size = "last"), cbind(ggplotGrob(bctGirls), ggplotGrob(bctBoys), size = "last")))

#### Agreement-measured BCTs

These questions were asked with the lead “Have you done the following during the last three weeks?”.

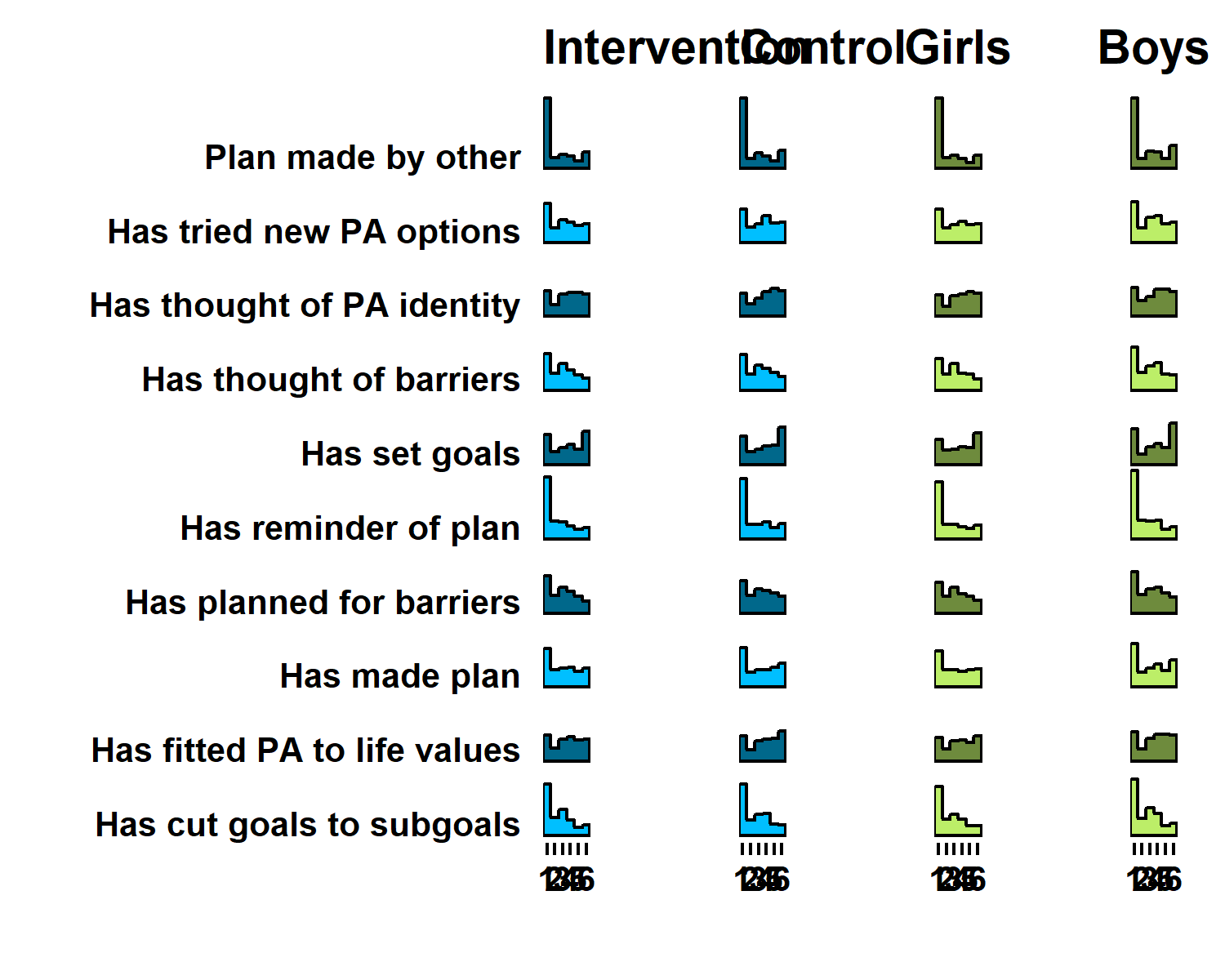
The answer scale was as follows:

0 = not at all true 1 … 4 [unlabeled] 5 = completely true

Items are as follows:

1. I have set PA goals for myself.
2. I have personally made a specific plan (“what, where, how”) to implement my PA.
3. I have a PA plan, which has been made by someone else, e.g. my sports club (e.g. a workout schedule).
4. I have a way by which I remind myself of my PA plan, e.g. I write it down in the calendar.
5. I have cut larger PA goals to smaller subgoals.
6. I have tried out new ways for me to be physically active.
7. I have pondered, what kind of difficult situations or barriers prevent me from implementing my PA plan.
8. I have planned for ways to overcome barriers to doing PA.
9. I have thought about how PA fits my identity (self concept).
10. I have attempted to find ways to exercise so, that it won’t obstruct but instead helps actualise my other life values.

bctGirls <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 'Has set goals' = PA\_agrbct\_01\_T1,  
 'Has made plan' = PA\_agrbct\_02\_T1,  
 'Plan made by other' = PA\_agrbct\_03\_T1,  
 'Has reminder of plan' = PA\_agrbct\_04\_T1,  
 'Has cut goals to subgoals' = PA\_agrbct\_05\_T1,  
 'Has tried new PA options' = PA\_agrbct\_06\_T1,  
 'Has thought of barriers' = PA\_agrbct\_07\_T1,  
 'Has planned for barriers' = PA\_agrbct\_08\_T1,  
 'Has thought of PA identity' = PA\_agrbct\_09\_T1,  
 'Has fitted PA to life values' = PA\_agrbct\_10\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(girl == "1") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:6), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "", labels = NULL) +  
 ggridges::scale\_fill\_cyclical(values = c("darkolivegreen2", "darkolivegreen4")) +  
 labs(title = "Girls") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 plot.title = element\_text(hjust = 0.5),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 6.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
bctBoys <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 'Has set goals' = PA\_agrbct\_01\_T1,  
 'Has made plan' = PA\_agrbct\_02\_T1,  
 'Plan made by other' = PA\_agrbct\_03\_T1,  
 'Has reminder of plan' = PA\_agrbct\_04\_T1,  
 'Has cut goals to subgoals' = PA\_agrbct\_05\_T1,  
 'Has tried new PA options' = PA\_agrbct\_06\_T1,  
 'Has thought of barriers' = PA\_agrbct\_07\_T1,  
 'Has planned for barriers' = PA\_agrbct\_08\_T1,  
 'Has thought of PA identity' = PA\_agrbct\_09\_T1,  
 'Has fitted PA to life values' = PA\_agrbct\_10\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(girl == "0") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:6), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "", labels = NULL) +  
 ggridges::scale\_fill\_cyclical(values = c("darkolivegreen2", "darkolivegreen4")) +  
 labs(title = "Boys") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 plot.title = element\_text(hjust = 0.5),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 6.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
bctInt <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 'Has set goals' = PA\_agrbct\_01\_T1,  
 'Has made plan' = PA\_agrbct\_02\_T1,  
 'Plan made by other' = PA\_agrbct\_03\_T1,  
 'Has reminder of plan' = PA\_agrbct\_04\_T1,  
 'Has cut goals to subgoals' = PA\_agrbct\_05\_T1,  
 'Has tried new PA options' = PA\_agrbct\_06\_T1,  
 'Has thought of barriers' = PA\_agrbct\_07\_T1,  
 'Has planned for barriers' = PA\_agrbct\_08\_T1,  
 'Has thought of PA identity' = PA\_agrbct\_09\_T1,  
 'Has fitted PA to life values' = PA\_agrbct\_10\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(intervention == "1") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:6), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "") +  
 ggridges::scale\_fill\_cyclical(values = c("deepskyblue", "deepskyblue4")) +  
 labs(title = "Intervention") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 6.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
bctCont <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 'Has set goals' = PA\_agrbct\_01\_T1,  
 'Has made plan' = PA\_agrbct\_02\_T1,  
 'Plan made by other' = PA\_agrbct\_03\_T1,  
 'Has reminder of plan' = PA\_agrbct\_04\_T1,  
 'Has cut goals to subgoals' = PA\_agrbct\_05\_T1,  
 'Has tried new PA options' = PA\_agrbct\_06\_T1,  
 'Has thought of barriers' = PA\_agrbct\_07\_T1,  
 'Has planned for barriers' = PA\_agrbct\_08\_T1,  
 'Has thought of PA identity' = PA\_agrbct\_09\_T1,  
 'Has fitted PA to life values' = PA\_agrbct\_10\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(intervention == "0") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:6), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "", labels = NULL) +  
 ggridges::scale\_fill\_cyclical(values = c("deepskyblue", "deepskyblue4")) +  
 labs(title = "Control") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 6.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
#grid.arrange(bctInt, bctGirls, bctCont, bctBoys, ncol = 2)  
  
# ("Seldom or never", "About once a month", "About once a week", "Almost daily")  
  
# This draws all histograms next to each other:  
grid::grid.newpage()  
grid::grid.draw(cbind(ggplotGrob(bctInt), ggplotGrob(bctCont), ggplotGrob(bctGirls), ggplotGrob(bctBoys), size = "last"))  
## Warning: Removed 60 rows containing non-finite values (stat\_binline).  
## Warning: Removed 70 rows containing non-finite values (stat\_binline).  
## Warning: Removed 60 rows containing non-finite values (stat\_binline).  
## Warning: Removed 70 rows containing non-finite values (stat\_binline).

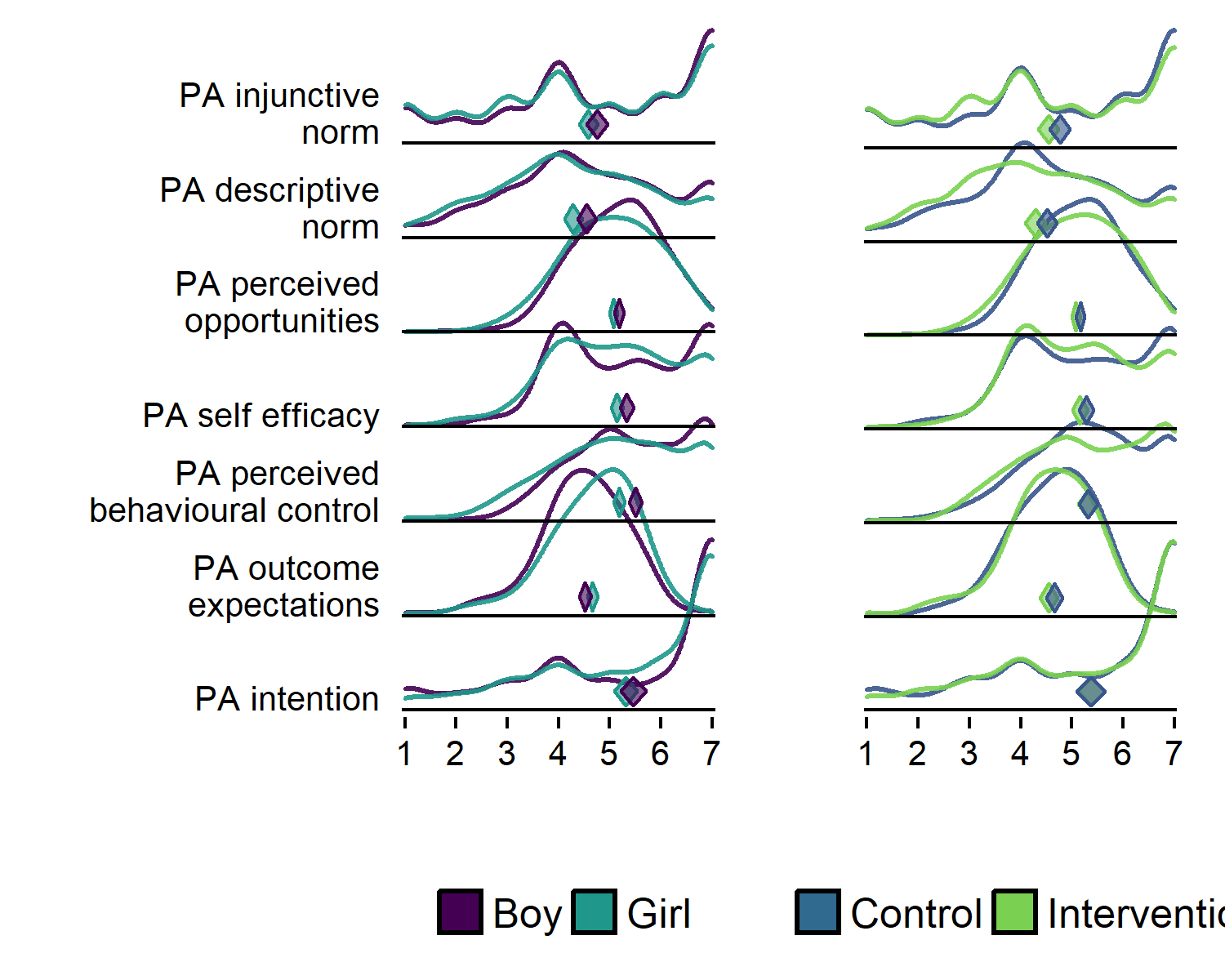


# This draws 2 histograms per row:  
# grid.newpage()  
# grid.draw(rbind(cbind(ggplotGrob(bctInt), ggplotGrob(bctCont), size = "last"), cbind(ggplotGrob(bctGirls), ggplotGrob(bctBoys), size = "last")))  
  
# bctBoys <- df %>% dplyr::select(id,  
# intervention,  
# group,  
# school,  
# girl,  
# 'Has set goals' = PA\_agrbct\_01\_T1,  
# 'Has made plan' = PA\_agrbct\_02\_T1,  
# 'Plan made by other' = PA\_agrbct\_03\_T1,  
# 'Has reminder of plan' = PA\_agrbct\_04\_T1,  
# 'Has cut goals to subgoals' = PA\_agrbct\_05\_T1,  
# 'Has tried new PA options' = PA\_agrbct\_06\_T1,  
# 'Has thought of barriers' = PA\_agrbct\_07\_T1,  
# 'Has planned for barriers' = PA\_agrbct\_08\_T1,  
# 'Has thought of PA identity' = PA\_agrbct\_09\_T1,  
# 'Has fitted PA to life values' = PA\_agrbct\_10\_T1)  
#   
# mean(bctBoys$`Has set goals`, na.rm = T)  
# mean(bctBoys$`Has made plan`, na.rm = T)  
# mean(bctBoys$`Plan made by other`, na.rm = T)  
# mean(bctBoys$`Has reminder of plan`, na.rm = T)  
# mean(bctBoys$`Has cut goals to subgoals`, na.rm = T)  
# mean(bctBoys$`Has tried new PA options`, na.rm = T)  
# mean(bctBoys$`Has thought of barriers`, na.rm = T)  
# mean(bctBoys$`Has planned for barriers`, na.rm = T)  
# mean(bctBoys$`Has thought of PA identity`, na.rm = T)  
# mean(bctBoys$`Has fitted PA to life values`, na.rm = T)

## PA determinants

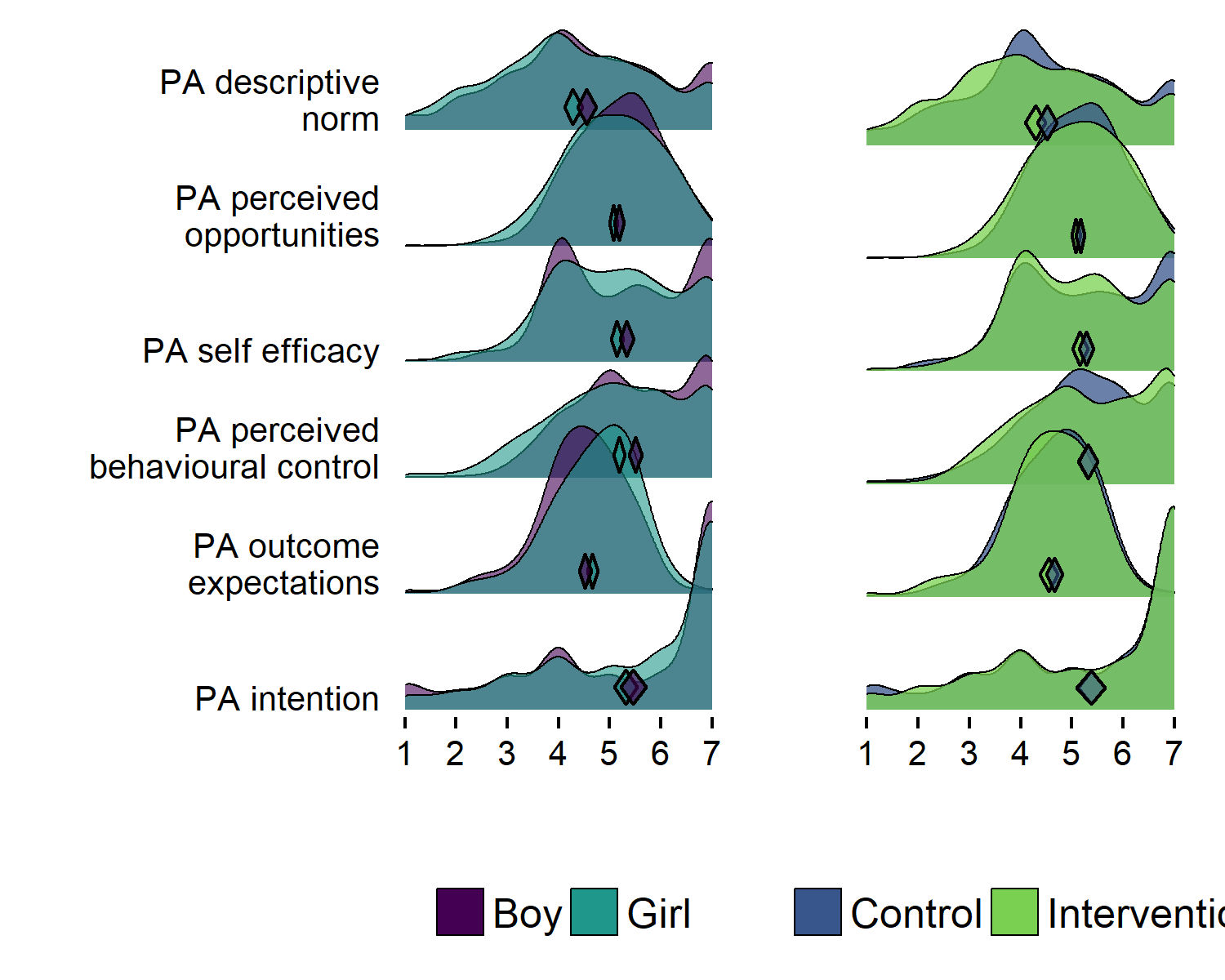
### Plot without fill

plot1 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 'PA intention' = PA\_intention\_T1,  
 'PA outcome\nexpectations' = PA\_outcomeExpectations\_T1,  
 'PA perceived\nbehavioural control' = PA\_pbc\_T1,  
 'PA self efficacy' = PA\_selfefficacy\_T1,  
 'PA perceived\nopportunities' = PA\_opportunities\_T1,  
 'PA descriptive\nnorm' = PA\_dnorm\_T1,  
 'PA injunctive\nnorm' = PA\_inorm\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.), factor\_key = TRUE) %>%  
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, colour = paste(Variable, girl),   
 fill = paste(Variable, girl)),   
 alpha = 0, size = 0.75, from = 1, to = 7) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0)) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("PA injunctive\nnorm 0", "PA injunctive\nnorm 1"),  
 labels = c( 'PA injunctive\nnorm 0' = "Boy", 'PA injunctive\nnorm 1' = "Girl"),  
 values = viridis::viridis(4, end = 0.8)[c(1, 3)], #c("#3bc600", "#0000ff", "#9cc68b", "#8080ff"),  
 name = "", guide = guide\_legend(override.aes = list(alpha = 1))) +  
 ggridges::scale\_colour\_cyclical(values = viridis::viridis(4, end = 0.8, alpha = 0.9)[c(1, 3)]) +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom", axis.text=element\_text(size=10)) +  
 geom\_hline(yintercept = 1:7)  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
target = c(  
"PA\_intention\_T1",  
"PA\_outcomeExpectations\_T1",  
"PA\_pbc\_T1",  
"PA\_selfefficacy\_T1",  
"PA\_opportunities\_T1",  
"PA\_dnorm\_T1",  
"PA\_inorm\_T1")  
  
PA\_ci\_girls <- ci\_girls %>% filter(diamondlabels %in% names(scales\_T1) &   
 grepl("PA\_intention\_T1", diamondlabels) |   
 grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
 grepl("PA\_pbc\_T1", diamondlabels) |   
 grepl("PA\_selfefficacy\_T1", diamondlabels) |   
 grepl("PA\_opportunities\_T1", diamondlabels) |   
 grepl("PA\_dnorm\_T1", diamondlabels) |   
 grepl("PA\_inorm\_T1", diamondlabels))  
  
PA\_ci\_girls <- PA\_ci\_girls[match(target, PA\_ci\_girls$diamondlabels), ]  
  
PA\_ci\_boys <- ci\_boys %>% filter(diamondlabels %in% names(scales\_T1) &   
 grepl("PA\_intention\_T1", diamondlabels) |   
 grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
 grepl("PA\_pbc\_T1", diamondlabels) |   
 grepl("PA\_selfefficacy\_T1", diamondlabels) |   
 grepl("PA\_opportunities\_T1", diamondlabels) |   
 grepl("PA\_dnorm\_T1", diamondlabels) |   
 grepl("PA\_inorm\_T1", diamondlabels))  
  
PA\_ci\_boys <- PA\_ci\_boys[match(target, PA\_ci\_boys$diamondlabels), ]  
  
plot1 <- plot1 + userfriendlyscience::diamondPlot(PA\_ci\_girls, returnLayerOnly = TRUE, color=viridis::viridis(4, end = 0.8)[c(3)],   
 alpha=.6, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2)) +  
 userfriendlyscience::diamondPlot(PA\_ci\_boys, returnLayerOnly = TRUE, color=viridis::viridis(4, end = 0.8)[c(1)],   
 alpha=.6, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2))  
  
plot2 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 'PA intention' = PA\_intention\_T1,  
 'PA outcome\nexpectations' = PA\_outcomeExpectations\_T1,  
 'PA perceived\nbehavioural control' = PA\_pbc\_T1,  
 'PA self efficacy' = PA\_selfefficacy\_T1,  
 'PA perceived\nopportunities' = PA\_opportunities\_T1,  
 'PA descriptive\nnorm' = PA\_dnorm\_T1,  
 'PA injunctive\nnorm' = PA\_inorm\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.), factor\_key = TRUE) %>%  
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, colour = paste(Variable, intervention),   
 fill = paste(Variable, intervention)),   
 alpha = 0, size = 0.75, from = 1, to = 7) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), labels = NULL) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("PA injunctive\nnorm 0", "PA injunctive\nnorm 1"),  
 labels = c( 'PA injunctive\nnorm 0' = "Control", 'PA injunctive\nnorm 1' = "Intervention"),  
 values = c(viridis::viridis(4, end = 0.8)[c(2, 4)], viridis::viridis(8, end = 0.8)[c(4, 8)]), # get 2 x 2 colors, close to each other MUUTTAAKO MITÄÄN?  
 name = "", guide = guide\_legend(override.aes = list(alpha = 1))) +  
 ggridges::scale\_colour\_cyclical(values = viridis::viridis(4, end = 0.8, alpha = 0.9)[c(2, 4)]) +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom", axis.text=element\_text(size=10)) +  
 geom\_hline(yintercept = 1:7)  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
PA\_ci\_intervention <- ci\_intervention %>% filter(diamondlabels %in% names(scales\_T1) &   
 grepl("PA\_intention\_T1", diamondlabels) |   
 grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
 grepl("PA\_pbc\_T1", diamondlabels) |   
 grepl("PA\_selfefficacy\_T1", diamondlabels) |   
 grepl("PA\_opportunities\_T1", diamondlabels) |   
 grepl("PA\_dnorm\_T1", diamondlabels) |   
 grepl("PA\_inorm\_T1", diamondlabels))  
  
PA\_ci\_intervention <- PA\_ci\_intervention[match(target, PA\_ci\_intervention$diamondlabels), ]  
  
PA\_ci\_control <- ci\_control %>% filter(diamondlabels %in% names(scales\_T1) &   
 grepl("PA\_intention\_T1", diamondlabels) |   
 grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
 grepl("PA\_pbc\_T1", diamondlabels) |   
 grepl("PA\_selfefficacy\_T1", diamondlabels) |   
 grepl("PA\_opportunities\_T1", diamondlabels) |   
 grepl("PA\_dnorm\_T1", diamondlabels) |   
 grepl("PA\_inorm\_T1", diamondlabels))  
  
PA\_ci\_control <- PA\_ci\_control[match(target, PA\_ci\_control$diamondlabels), ]  
  
plot2 <- plot2 +   
 userfriendlyscience::diamondPlot(PA\_ci\_intervention, returnLayerOnly = TRUE, color=viridis::viridis(4, end = 0.8)[c(4)],  
 alpha=.6, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2)) +  
 userfriendlyscience::diamondPlot(PA\_ci\_control, returnLayerOnly = TRUE, color=viridis::viridis(4, end = 0.8)[c(2)],   
 alpha=.6, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2))  
  
# grid.arrange(plot1, plot2, ncol = 2)  
grid::grid.newpage()  
grid::grid.draw(cbind(ggplotGrob(plot1), ggplotGrob(plot2), size = "first"))  
## Picking joint bandwidth of 0.355  
## Warning: Removed 641 rows containing non-finite values  
## (stat\_density\_ridges).  
## Picking joint bandwidth of 0.359  
## Warning: Removed 641 rows containing non-finite values  
## (stat\_density\_ridges).



### Pretty plot with fill

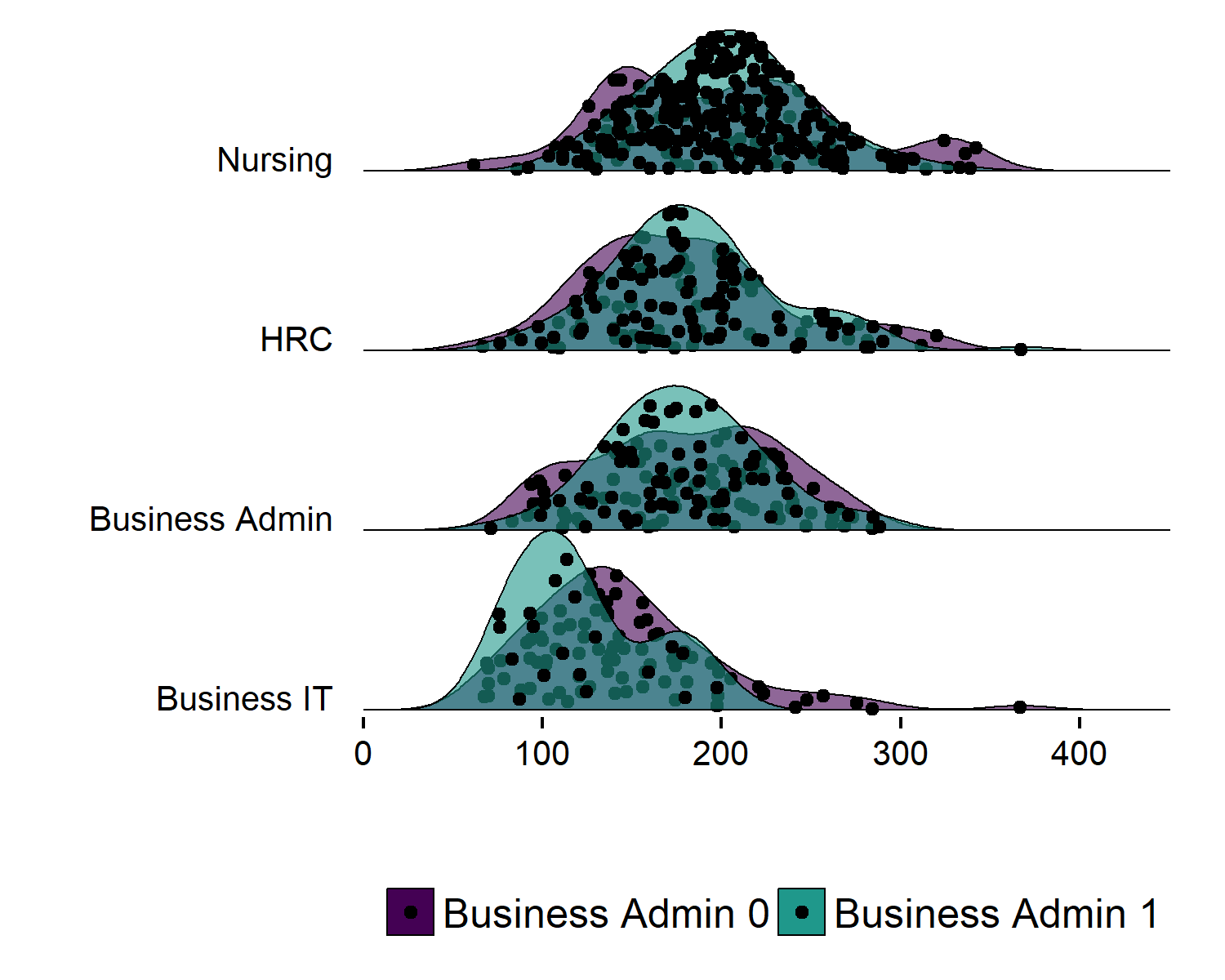
plot1 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 'PA intention' = PA\_intention\_T1,  
 'PA outcome\nexpectations' = PA\_outcomeExpectations\_T1,  
 'PA perceived\nbehavioural control' = PA\_pbc\_T1,  
 'PA self efficacy' = PA\_selfefficacy\_T1,  
 'PA perceived\nopportunities' = PA\_opportunities\_T1,  
 'PA descriptive\nnorm' = PA\_dnorm\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.), factor\_key = TRUE) %>%  
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, colour = "black",   
 fill = paste(Variable, girl)),   
 alpha = 0.6, size = 0.25, from = 1, to = 7) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0)) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("PA descriptive\nnorm 0", "PA descriptive\nnorm 1"),  
 labels = c( 'PA descriptive\nnorm 0' = "Boy", 'PA descriptive\nnorm 1' = "Girl"),  
 values = viridis::viridis(4, end = 0.8)[c(1, 3)],  
 name = "", guide = guide\_legend(override.aes = list(alpha = 1))) +  
 ggridges::scale\_colour\_cyclical(values = "black") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom", axis.text=element\_text(size=10))  
  
target = c(  
"PA\_intention\_T1",  
"PA\_outcomeExpectations\_T1",  
"PA\_pbc\_T1",  
"PA\_selfefficacy\_T1",  
"PA\_opportunities\_T1",  
"PA\_dnorm\_T1")  
  
PA\_ci\_girls <- ci\_girls %>% filter(diamondlabels %in% names(scales\_T1) &   
 grepl("PA\_intention\_T1", diamondlabels) |   
 grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
 grepl("PA\_pbc\_T1", diamondlabels) |   
 grepl("PA\_selfefficacy\_T1", diamondlabels) |   
 grepl("PA\_opportunities\_T1", diamondlabels) |   
 grepl("PA\_dnorm\_T1", diamondlabels))  
  
PA\_ci\_girls <- PA\_ci\_girls[match(target, PA\_ci\_girls$diamondlabels), ]  
  
PA\_ci\_boys <- ci\_boys %>% filter(diamondlabels %in% names(scales\_T1) &   
 grepl("PA\_intention\_T1", diamondlabels) |   
 grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
 grepl("PA\_pbc\_T1", diamondlabels) |   
 grepl("PA\_selfefficacy\_T1", diamondlabels) |   
 grepl("PA\_opportunities\_T1", diamondlabels) |   
 grepl("PA\_dnorm\_T1", diamondlabels))  
  
PA\_ci\_boys <- PA\_ci\_boys[match(target, PA\_ci\_boys$diamondlabels), ]  
  
plot1 <- plot1 + userfriendlyscience::diamondPlot(PA\_ci\_girls, returnLayerOnly = TRUE, lineColor = "black", color=viridis::viridis(4, end = 0.8)[3],   
 alpha=.6, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2)) +  
 userfriendlyscience::diamondPlot(PA\_ci\_boys, returnLayerOnly = TRUE, lineColor = "black", linetype = "solid", color=viridis::viridis(4, end = 0.8)[1],   
 alpha=.6, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2))  
  
plot2 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 'PA intention' = PA\_intention\_T1,  
 'PA outcome\nexpectations' = PA\_outcomeExpectations\_T1,  
 'PA perceived\nbehavioural control' = PA\_pbc\_T1,  
 'PA self efficacy' = PA\_selfefficacy\_T1,  
 'PA perceived\nopportunities' = PA\_opportunities\_T1,  
 'PA descriptive\nnorm' = PA\_dnorm\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.), factor\_key = TRUE) %>%  
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, colour = "black",   
 fill = paste(Variable, intervention)),   
 alpha = 0.75, size = 0.25, from = 1, to = 7) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), labels = NULL) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("PA descriptive\nnorm 0", "PA descriptive\nnorm 1"),  
 labels = c( 'PA descriptive\nnorm 0' = "Control", 'PA descriptive\nnorm 1' = "Intervention"),  
 values = viridis::viridis(4, end = 0.8)[c(2, 4)],  
 name = "", guide = guide\_legend(override.aes = list(alpha = 1))) +  
 ggridges::scale\_colour\_cyclical(values = "black") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom", axis.text=element\_text(size=10))  
  
PA\_ci\_intervention <- ci\_intervention %>% filter(diamondlabels %in% names(scales\_T1) &   
 grepl("PA\_intention\_T1", diamondlabels) |   
 grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
 grepl("PA\_pbc\_T1", diamondlabels) |   
 grepl("PA\_selfefficacy\_T1", diamondlabels) |   
 grepl("PA\_opportunities\_T1", diamondlabels) |   
 grepl("PA\_dnorm\_T1", diamondlabels))  
  
PA\_ci\_intervention <- PA\_ci\_intervention[match(target, PA\_ci\_intervention$diamondlabels), ]  
  
PA\_ci\_control <- ci\_control %>% filter(diamondlabels %in% names(scales\_T1) &   
 grepl("PA\_intention\_T1", diamondlabels) |   
 grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
 grepl("PA\_pbc\_T1", diamondlabels) |   
 grepl("PA\_selfefficacy\_T1", diamondlabels) |   
 grepl("PA\_opportunities\_T1", diamondlabels) |   
 grepl("PA\_dnorm\_T1", diamondlabels))  
  
PA\_ci\_control <- PA\_ci\_control[match(target, PA\_ci\_control$diamondlabels), ]  
  
plot2 <- plot2 +   
 userfriendlyscience::diamondPlot(PA\_ci\_intervention, returnLayerOnly = TRUE, lineColor = "black", color=viridis::viridis(4, end = 0.8)[4],  
 alpha=.6, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2)) +  
 userfriendlyscience::diamondPlot(PA\_ci\_control, returnLayerOnly = TRUE, lineColor = "black", color=viridis::viridis(4, end = 0.8)[2],   
 alpha=.6, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2))  
  
# grid.arrange(plot1, plot2, ncol = 2)  
grid::grid.newpage()  
grid::grid.draw(cbind(ggplotGrob(plot1), ggplotGrob(plot2), size = "first"))  
## Picking joint bandwidth of 0.327  
## Warning: Removed 549 rows containing non-finite values  
## (stat\_density\_ridges).  
## Picking joint bandwidth of 0.332  
## Warning: Removed 549 rows containing non-finite values  
## (stat\_density\_ridges).



# Test that the legend is right; descriptive norms are smaller for girls  
df %>% group\_by(girl) %>% summarise(mean = mean(PA\_dnorm\_T1, na.rm = T)) %>% filter(girl == "1") %>% select(mean) < df %>% group\_by(girl) %>% summarise(mean = mean(PA\_dnorm\_T1, na.rm = T)) %>% filter(girl == "0") %>% select(mean)  
## mean  
## [1,] TRUE  
  
# KESKEN: iv/ctrl-legend  
df %>% group\_by(intervention) %>% summarise(mean = mean(PA\_dnorm\_T1, na.rm = TRUE)) %>% filter(intervention == "1") %>% select(mean) < df %>% group\_by(intervention) %>% summarise(mean = mean(PA\_dnorm\_T1, na.rm = T)) %>% filter(intervention == "0") %>% select(mean)  
## mean  
## [1,] TRUE

### MVPA, girls and boys in different educational tracks

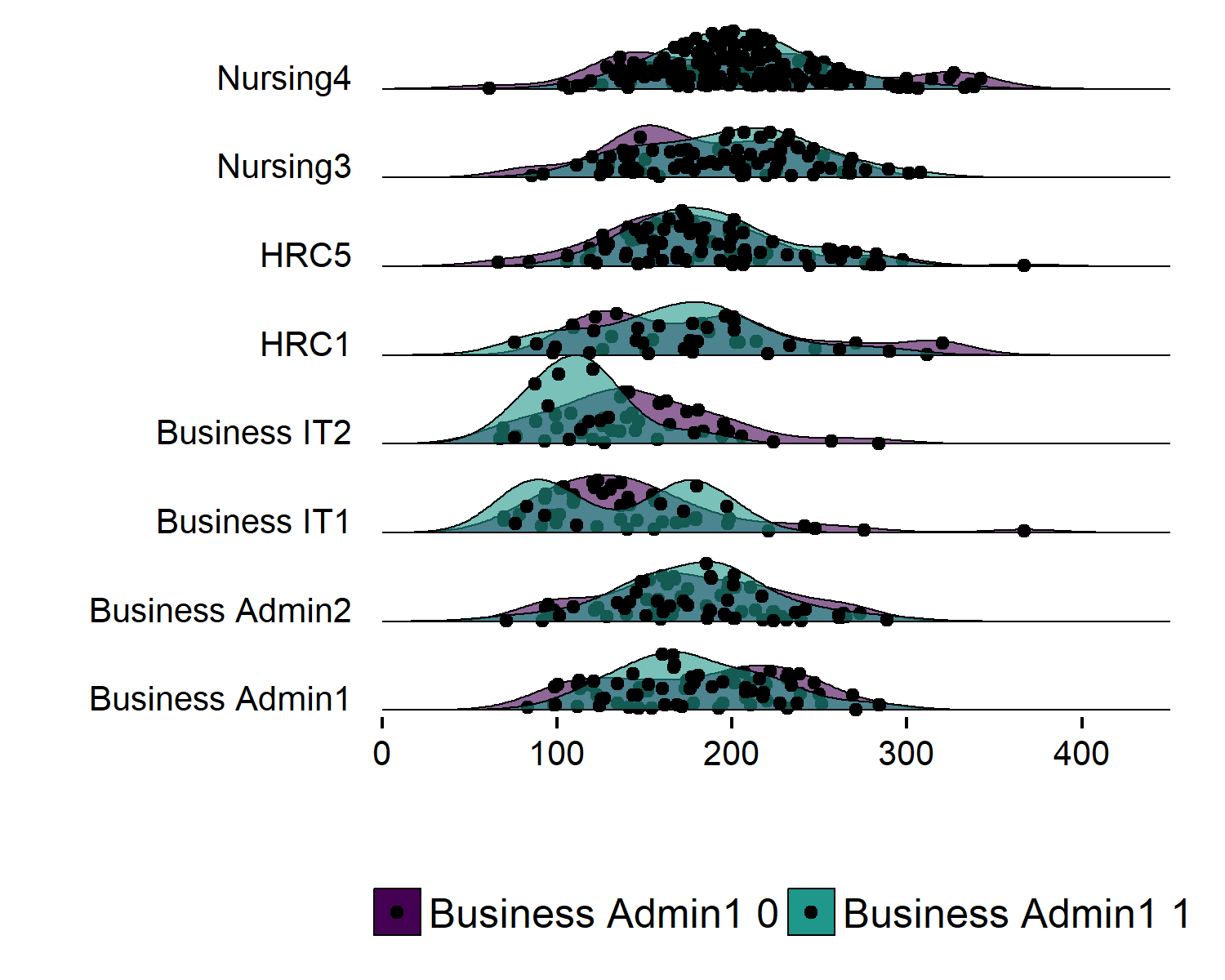
plot1 <- df %>% dplyr::select(id,  
 track = track,  
 girl,  
 PA = paAccelerometer\_T1) %>%  
 filter(!is.na(track), track != "Other") %>% # Drop category "Other"  
 ggplot(aes(y = track)) +  
 ggridges::geom\_density\_ridges2(aes(x = PA, colour = "black",   
 fill = paste(track, girl)),  
 scale = 1,  
 alpha = 0.6, size = 0.25,  
 from = 0, to = 450,  
 jittered\_points=TRUE, point\_shape=21,  
 point\_fill="black") +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0)) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(  
 labels = c('Business Admin boy' = "Boy", 'Business Admin girl' = "Girl"),  
 values = viridis::viridis(4, end = 0.8)[c(1, 3)],  
 name = "", guide = guide\_legend(override.aes = list(alpha = 1))) +  
 ggridges::scale\_colour\_cyclical(values = "black") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom", axis.text=element\_text(size=10))  
plot1  
## Picking joint bandwidth of 18.1  
## Warning: Removed 370 rows containing non-finite values  
## (stat\_density\_ridges).



df %>% select(paAccelerometer\_T1, girl, track) %>%   
 group\_by(girl, track) %>%   
 summarise(mean = mean(paAccelerometer\_T1, na.rm = TRUE),  
 median = median(paAccelerometer\_T1, na.rm = TRUE),  
 max = max(paAccelerometer\_T1, na.rm = TRUE),  
 min = min(paAccelerometer\_T1, na.rm = TRUE),  
 sd = sd(paAccelerometer\_T1, na.rm = TRUE),  
 n = n())  
## # A tibble: 11 x 8  
## # Groups: girl [?]  
## girl track mean median max min sd n  
## <fctr> <fctr> <dbl> <dbl> <dbl> <dbl> <dbl> <int>  
## 1 1 Other 188 198 264 102 48.6 17  
## 2 1 Business IT 122 114 197 75.7 37.7 26  
## 3 1 Business Admin 179 176 288 71.2 43.5 110  
## 4 1 HRC 185 178 367 76.0 49.8 129  
## 5 1 Nursing 201 201 339 92.1 44.9 331  
## 6 0 Other 182 217 262 73.4 75.9 7  
## 7 0 Business IT 146 137 367 67.6 53.5 137  
## 8 0 Business Admin 180 176 284 83.2 52.2 172  
## 9 0 HRC 179 177 320 66.5 56.3 84  
## 10 0 Nursing 201 203 342 61.5 65.9 71  
## 11 <NA> <NA> 194 162 291 138 65.9 81  
  
df %>% select(paAccelerometer\_T1, girl, track) %>%   
 group\_by(girl) %>%   
 summarise(mean = mean(paAccelerometer\_T1, na.rm = TRUE),  
 median = median(paAccelerometer\_T1, na.rm = TRUE),  
 max = max(paAccelerometer\_T1, na.rm = TRUE),  
 min = min(paAccelerometer\_T1, na.rm = TRUE),  
 sd = sd(paAccelerometer\_T1, na.rm = TRUE),  
 n = n())  
## # A tibble: 3 x 7  
## girl mean median max min sd n  
## <fctr> <dbl> <dbl> <dbl> <dbl> <dbl> <int>  
## 1 1 190 189 367 71.2 48.6 613  
## 2 0 174 165 367 61.5 59.4 471  
## 3 <NA> 194 162 291 138 65.9 81  
  
# target = c(  
# "PA\_intention\_T1",  
# "PA\_outcomeExpectations\_T1",  
# "PA\_pbc\_T1",  
# "PA\_selfefficacy\_T1",  
# "PA\_opportunities\_T1",  
# "PA\_dnorm\_T1")  
#   
# PA\_ci\_girls <- ci\_girls %>% filter(paAccelerometer)  
#   
# PA\_ci\_girls <- PA\_ci\_girls[match(target, PA\_ci\_girls$diamondlabels), ]  
#   
# PA\_ci\_boys <- ci\_boys %>% filter(diamondlabels %in% names(scales\_T1) &   
# grepl("PA\_intention\_T1", diamondlabels) |   
# grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
# grepl("PA\_pbc\_T1", diamondlabels) |   
# grepl("PA\_selfefficacy\_T1", diamondlabels) |   
# grepl("PA\_opportunities\_T1", diamondlabels) |   
# grepl("PA\_dnorm\_T1", diamondlabels))  
#   
# PA\_ci\_boys <- PA\_ci\_boys[match(target, PA\_ci\_boys$diamondlabels), ]  
#   
# plot1 <- plot1 + userfriendlyscience::diamondPlot(PA\_ci\_girls, returnLayerOnly = TRUE, lineColor = "black", color=viridis::viridis(4, end = 0.8)[3],   
# alpha=.6, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2)) +  
# userfriendlyscience::diamondPlot(PA\_ci\_boys, returnLayerOnly = TRUE, lineColor = "black", linetype = "solid", color=viridis::viridis(4, end = 0.8)[1],   
# alpha=.6, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2))  
#

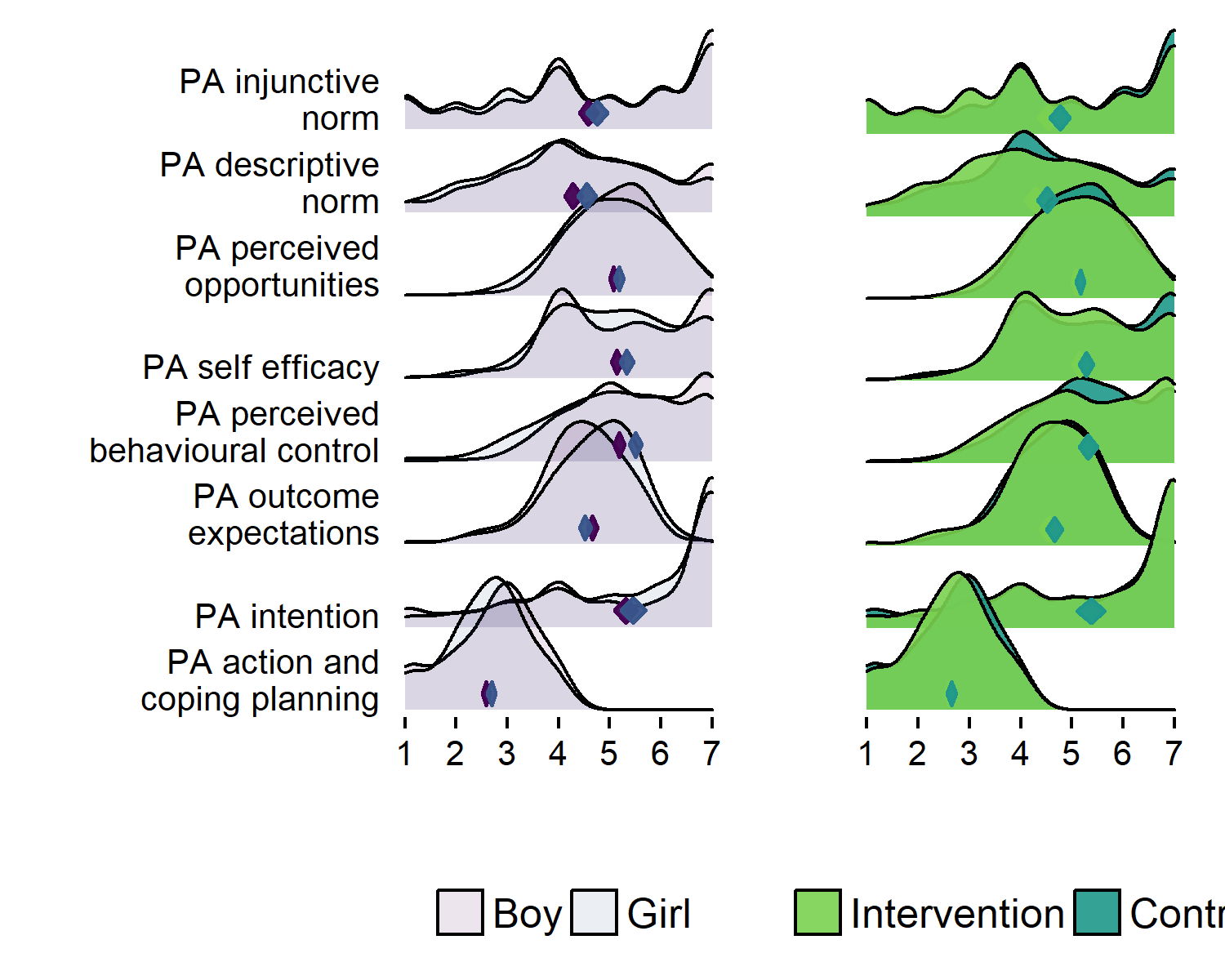
### MVPA, girls and boys in different educational tracks and schools

plot1 <- df %>% dplyr::select(id,  
 trackSchool = trackSchool,  
 girl,  
 PA = paAccelerometer\_T1) %>%  
 filter(!is.na(trackSchool), !grepl('Other|NANA|Business Admin4|HRC2|Nursing2', trackSchool)) %>% # Drop categories with just few participants  
 ggplot(aes(y = trackSchool)) +  
 ggridges::geom\_density\_ridges2(aes(x = PA, colour = "black",   
 fill = paste(trackSchool, girl)),  
 scale = 1,  
 alpha = 0.6, size = 0.25,  
 from = 0, to = 450,  
 jittered\_points=TRUE, point\_shape=21,  
 point\_fill="black") +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0)) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(  
 labels = c('Business Admin1 boy' = "Boy", 'Business Admin1 girl' = "Girl"),  
 values = viridis::viridis(4, end = 0.8)[c(1, 3)],  
 name = "", guide = guide\_legend(override.aes = list(alpha = 1))) +  
 ggridges::scale\_colour\_cyclical(values = "black") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom", axis.text=element\_text(size=10))  
plot1  
## Picking joint bandwidth of 20.3  
## Warning: Removed 367 rows containing non-finite values  
## (stat\_density\_ridges).

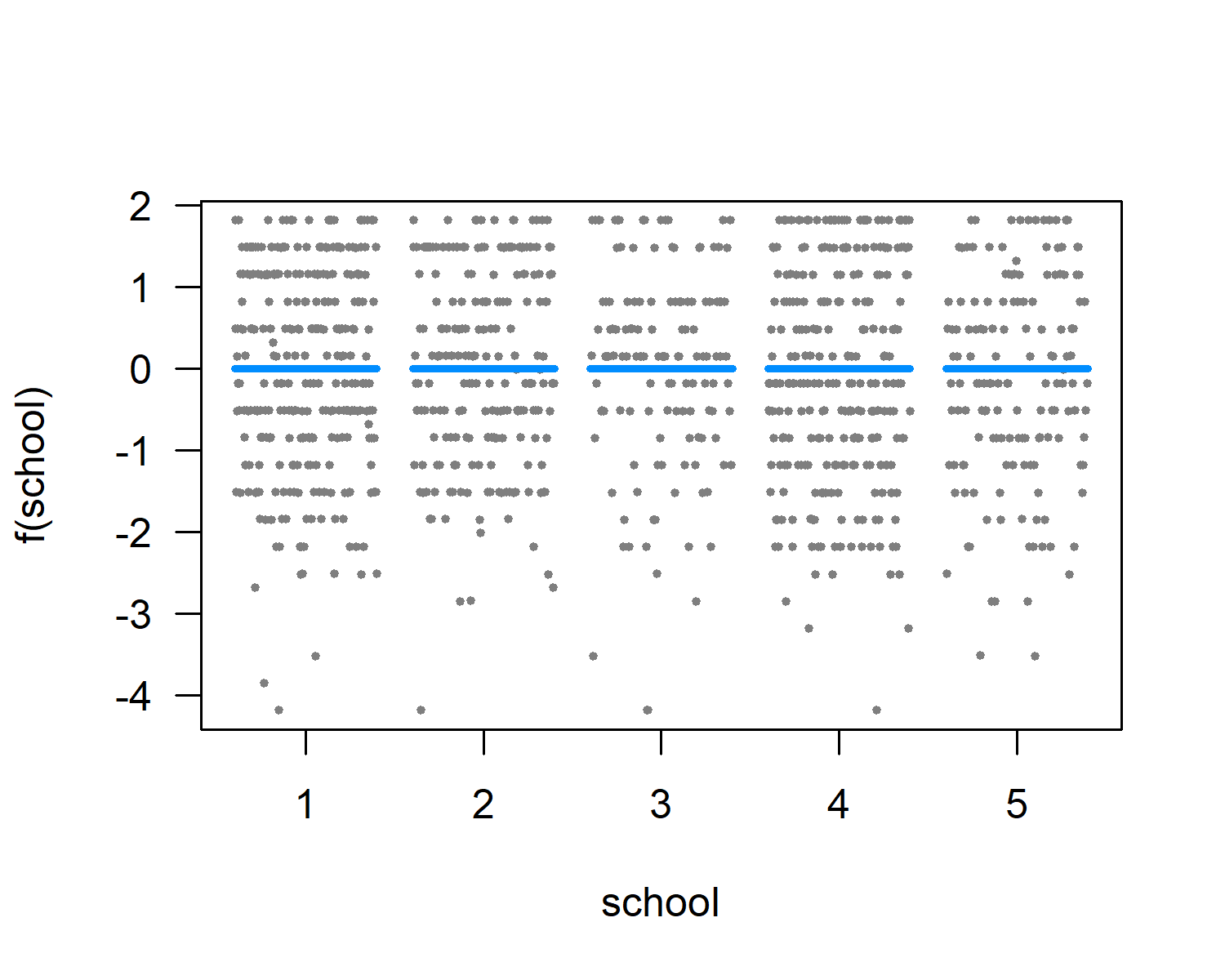


df %>% select(paAccelerometer\_T1, girl, trackSchool) %>%   
 group\_by(girl, trackSchool) %>%   
 summarise(mean = mean(paAccelerometer\_T1, na.rm = TRUE),  
 median = median(paAccelerometer\_T1, na.rm = TRUE),  
 max = max(paAccelerometer\_T1, na.rm = TRUE),  
 min = min(paAccelerometer\_T1, na.rm = TRUE),  
 sd = sd(paAccelerometer\_T1, na.rm = TRUE),  
 n = n())  
## # A tibble: 29 x 8  
## # Groups: girl [?]  
## girl trackSchool mean median max min sd n  
## <fctr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <int>  
## 1 1 Business Admin1 179 172 284 99.1 42.7 56  
## 2 1 Business Admin2 179 186 288 71.2 45.0 53  
## 3 1 Business Admin4 NaN NA -Inf Inf NaN 1  
## 4 1 Business IT1 134 135 197 76.2 48.4 12  
## 5 1 Business IT2 114 114 178 75.7 27.0 14  
## 6 1 HRC1 169 176 290 76.0 52.6 39  
## 7 1 HRC5 191 182 367 106 47.8 90  
## 8 1 Nursing3 198 203 308 92.1 48.4 104  
## 9 1 Nursing4 203 201 339 104 43.3 227  
## 10 1 Other1 142 142 142 142 NaN 1  
## # ... with 19 more rows  
  
df %>% select(paAccelerometer\_T1, girl, trackSchool) %>%   
 group\_by(girl) %>%   
 summarise(mean = mean(paAccelerometer\_T1, na.rm = TRUE),  
 median = median(paAccelerometer\_T1, na.rm = TRUE),  
 max = max(paAccelerometer\_T1, na.rm = TRUE),  
 min = min(paAccelerometer\_T1, na.rm = TRUE),  
 sd = sd(paAccelerometer\_T1, na.rm = TRUE),  
 n = n())  
## # A tibble: 3 x 7  
## girl mean median max min sd n  
## <fctr> <dbl> <dbl> <dbl> <dbl> <dbl> <int>  
## 1 1 190 189 367 71.2 48.6 613  
## 2 0 174 165 367 61.5 59.4 471  
## 3 <NA> 194 162 291 138 65.9 81

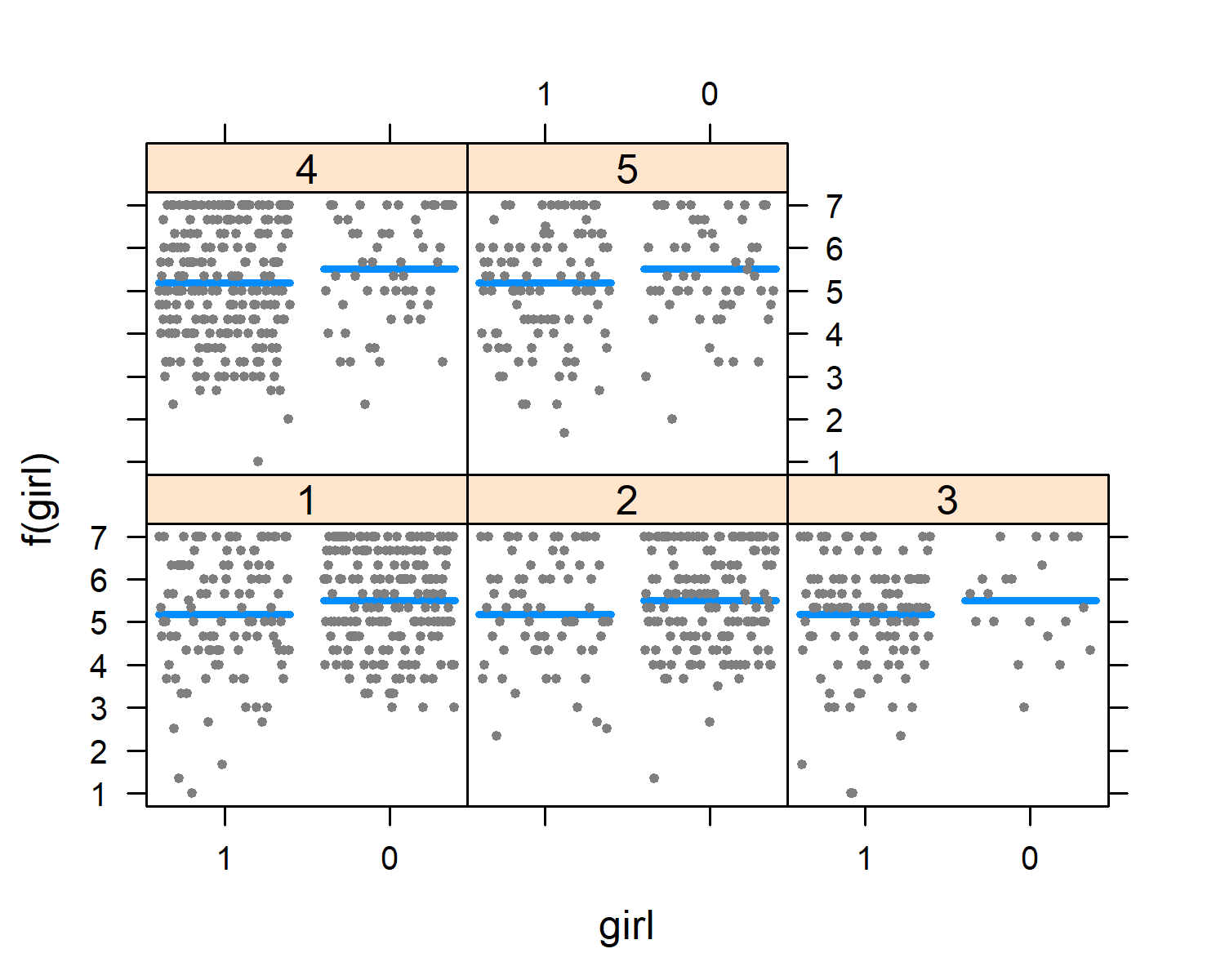
library(viridis::viridis)  
## Error in library(viridis::viridis): 'package' must be of length 1  
plot1 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
'PA action and\ncoping planning' = PA\_actCop\_T1,  
'PA intention' = PA\_intention\_T1,  
'PA outcome\nexpectations' = PA\_outcomeExpectations\_T1,  
'PA perceived\nbehavioural control' = PA\_pbc\_T1,  
'PA self efficacy' = PA\_selfefficacy\_T1,  
'PA perceived\nopportunities' = PA\_opportunities\_T1,  
'PA descriptive\nnorm' = PA\_dnorm\_T1,  
'PA injunctive\nnorm' = PA\_inorm\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.), factor\_key = TRUE) %>%   
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, fill = paste(Variable, girl)),   
 alpha = .1, color = "black", from = 1, to = 7) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0)) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("PA injunctive\nnorm 0", "PA injunctive\nnorm 1"),  
 labels = c( 'PA injunctive\nnorm 0' = "Boy", 'PA injunctive\nnorm 1' = "Girl"),  
 values = viridis::viridis(4, end = 0.8)[1:2], #c("#3bc600", "#0000ff", "#9cc68b", "#8080ff"),  
 name = "", guide = "legend") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom", axis.text=element\_text(size=10))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
plot2 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
'PA action and\ncoping planning' = PA\_actCop\_T1,  
'PA intention' = PA\_intention\_T1,  
'PA outcome\nexpectations' = PA\_outcomeExpectations\_T1,  
'PA perceived\nbehavioural control' = PA\_pbc\_T1,  
'PA self efficacy' = PA\_selfefficacy\_T1,  
'PA perceived\nopportunities' = PA\_opportunities\_T1,  
'PA descriptive\nnorm' = PA\_dnorm\_T1,  
'PA injunctive\nnorm' = PA\_inorm\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.), factor\_key = TRUE) %>%   
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, fill = paste(Variable, intervention)),   
 alpha = .9, color = "black", from = 1, to = 7) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), labels = NULL) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("PA injunctive\nnorm 1", "PA injunctive\nnorm 0"),  
 labels = c('PA injunctive\nnorm 1' = "Intervention", 'PA injunctive\nnorm 0' = "Control"),  
 values = viridis::viridis(4, end = 0.8)[3:4], #c("#ff0000", "#0000ff", "#ff8080", "#8080ff"),  
 name = "", guide = "legend") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom", axis.text=element\_text(size=10))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
target = c("PA\_actCop\_T1",  
"PA\_intention\_T1",  
"PA\_outcomeExpectations\_T1",  
"PA\_pbc\_T1",  
"PA\_selfefficacy\_T1",  
"PA\_opportunities\_T1",  
"PA\_dnorm\_T1",  
"PA\_inorm\_T1")  
  
PA\_ci\_girls <- ci\_girls %>% filter(diamondlabels %in% names(scales\_T1) &   
 grepl("PA\_actCop\_T1", diamondlabels) |   
 grepl("PA\_intention\_T1", diamondlabels) |   
 grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
 grepl("PA\_pbc\_T1", diamondlabels) |   
 grepl("PA\_selfefficacy\_T1", diamondlabels) |   
 grepl("PA\_opportunities\_T1", diamondlabels) |   
 grepl("PA\_dnorm\_T1", diamondlabels) |   
 grepl("PA\_inorm\_T1", diamondlabels))  
  
PA\_ci\_girls <- PA\_ci\_girls[match(target, PA\_ci\_girls$diamondlabels), ]  
  
PA\_ci\_boys <- ci\_boys %>% filter(diamondlabels %in% names(scales\_T1) &   
 grepl("PA\_actCop\_T1", diamondlabels) |   
 grepl("PA\_intention\_T1", diamondlabels) |   
 grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
 grepl("PA\_pbc\_T1", diamondlabels) |   
 grepl("PA\_selfefficacy\_T1", diamondlabels) |   
 grepl("PA\_opportunities\_T1", diamondlabels) |   
 grepl("PA\_dnorm\_T1", diamondlabels) |   
 grepl("PA\_inorm\_T1", diamondlabels))  
  
PA\_ci\_boys <- PA\_ci\_boys[match(target, PA\_ci\_boys$diamondlabels), ]  
  
plot1 <- plot1 +   
 userfriendlyscience::diamondPlot(PA\_ci\_girls, returnLayerOnly = TRUE, color=viridis::viridis(4, end = 0.8)[1],# 'blue',   
 alpha=.95, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2)) +  
 userfriendlyscience::diamondPlot(PA\_ci\_boys, returnLayerOnly = TRUE, color=viridis::viridis(4, end = 0.8)[2], #'green',   
 alpha=.95, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2))  
  
PA\_ci\_intervention <- ci\_intervention %>% filter(diamondlabels %in% names(scales\_T1) &   
 grepl("PA\_actCop\_T1", diamondlabels) |   
 grepl("PA\_intention\_T1", diamondlabels) |   
 grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
 grepl("PA\_pbc\_T1", diamondlabels) |   
 grepl("PA\_selfefficacy\_T1", diamondlabels) |   
 grepl("PA\_opportunities\_T1", diamondlabels) |   
 grepl("PA\_dnorm\_T1", diamondlabels) |   
 grepl("PA\_inorm\_T1", diamondlabels))  
  
PA\_ci\_intervention <- PA\_ci\_intervention[match(target, PA\_ci\_intervention$diamondlabels), ]  
  
PA\_ci\_control <- ci\_control %>% filter(diamondlabels %in% names(scales\_T1) &   
 grepl("PA\_actCop\_T1", diamondlabels) |   
 grepl("PA\_intention\_T1", diamondlabels) |   
 grepl("PA\_outcomeExpectations\_T1", diamondlabels) |   
 grepl("PA\_pbc\_T1", diamondlabels) |   
 grepl("PA\_selfefficacy\_T1", diamondlabels) |   
 grepl("PA\_opportunities\_T1", diamondlabels) |   
 grepl("PA\_dnorm\_T1", diamondlabels) |   
 grepl("PA\_inorm\_T1", diamondlabels))  
  
PA\_ci\_control <- PA\_ci\_control[match(target, PA\_ci\_control$diamondlabels), ]  
  
plot2 <- plot2 +   
 userfriendlyscience::diamondPlot(PA\_ci\_intervention, returnLayerOnly = TRUE, color=viridis::viridis(4, end = 0.8)[4],# 'blue',  
 alpha=.95, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2)) +  
 userfriendlyscience::diamondPlot(PA\_ci\_control, returnLayerOnly = TRUE, color=viridis::viridis(4, end = 0.8)[3], #'red',   
 alpha=.95, fixedSize = 0.15, otherAxisCol = (1:length(target) + .2))  
  
# grid.arrange(plot1, plot2, ncol = 2)  
grid::grid.newpage()  
grid::grid.draw(cbind(ggplotGrob(plot1), ggplotGrob(plot2), size = "first"))  
## Picking joint bandwidth of 0.337  
## Warning: Removed 733 rows containing non-finite values  
## (stat\_density\_ridges).  
## Picking joint bandwidth of 0.341  
## Warning: Removed 733 rows containing non-finite values  
## (stat\_density\_ridges).



library(visreg)  
library(broom)  
  
m\_pbcgirl <- lme4::lmer(PA\_pbc\_T1 ~ girl + (1 | school), data = df)  
summary(m\_pbcgirl)  
## Linear mixed model fit by REML ['lmerMod']  
## Formula: PA\_pbc\_T1 ~ girl + (1 | school)  
## Data: df  
##   
## REML criterion at convergence: 3561.2  
##   
## Scaled residuals:   
## Min 1Q Median 3Q Max   
## -3.2903 -0.6705 0.1155 0.9014 1.4253   
##   
## Random effects:  
## Groups Name Variance Std.Dev.   
## school (Intercept) 1.393e-14 1.180e-07  
## Residual 1.619e+00 1.272e+00  
## Number of obs: 1071, groups: school, 5  
##   
## Fixed effects:  
## Estimate Std. Error t value  
## (Intercept) 5.18644 0.05164 100.43  
## girl0 0.32578 0.07846 4.15  
##   
## Correlation of Fixed Effects:  
## (Intr)  
## girl0 -0.658  
  
visreg::visreg(m\_pbcgirl, "school", type = "contrast")



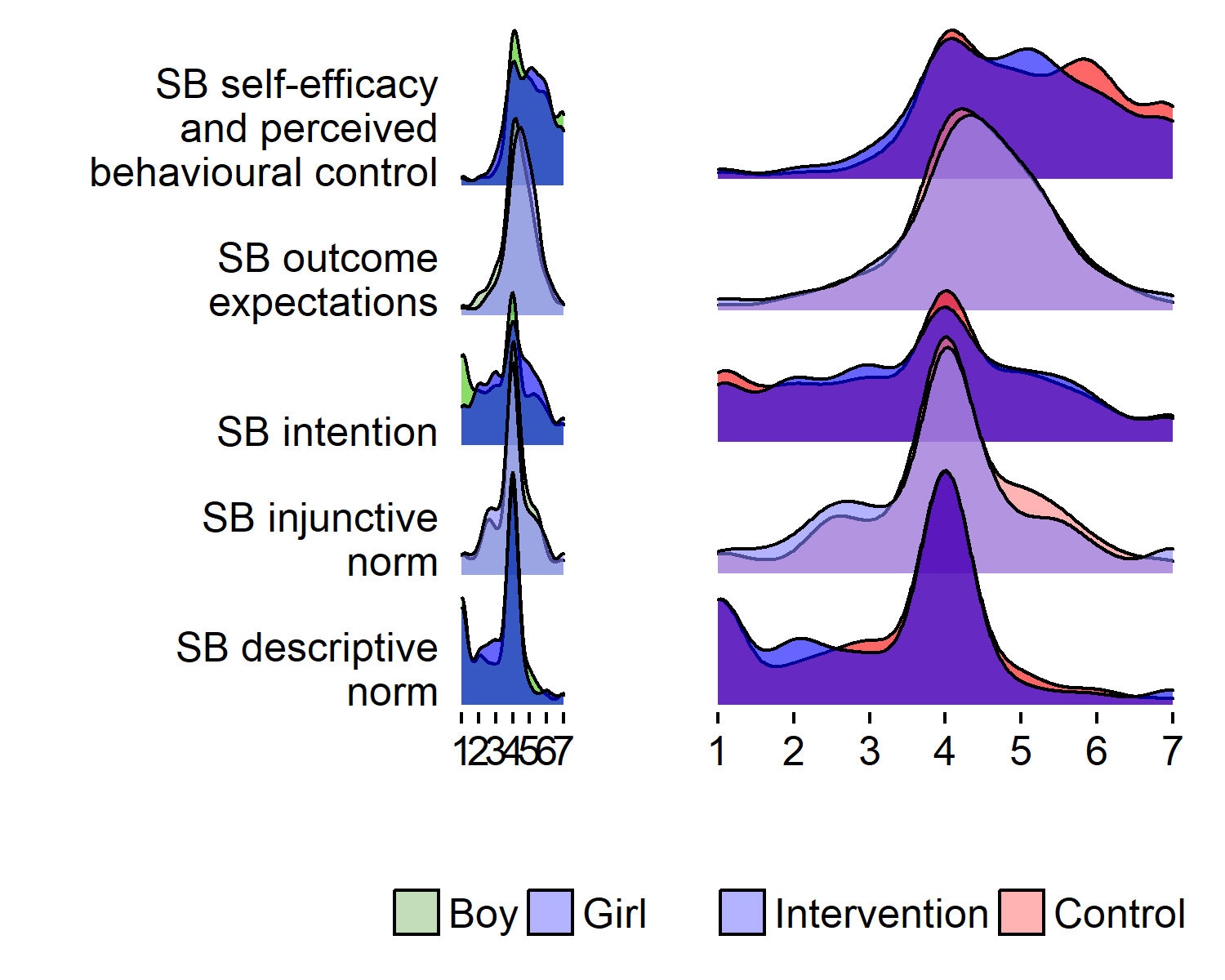
visreg(m\_pbcgirl, "girl", by="school", re.form=~(1|school))



glance(m\_pbcgirl)  
## sigma logLik AIC BIC deviance df.residual  
## 1 1.272372 -1780.624 3569.247 3589.153 3553.336 1067

## SB determinants

plot1 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
'SB intention' = SB\_intention\_T1,  
'SB outcome\nexpectations' = SB\_outcomeExpectations\_T1,  
'SB self-efficacy\nand perceived\nbehavioural control' = SB\_sePbc\_T1,  
'SB descriptive\nnorm' = SB\_dnorm\_T1,  
'SB injunctive\nnorm' = SB\_inorm\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%   
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, fill = paste(Variable, girl)),   
 alpha = .6, color = "black", from = 1, to = 7) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0)) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("SB injunctive\nnorm 0", "SB injunctive\nnorm 1"),  
 labels = c( 'SB injunctive\nnorm 0' = "Boy", 'SB injunctive\nnorm 1' = "Girl"),  
 values = c("#3bc600", "#0000ff", "#9cc68b", "#8080ff"),  
 name = "", guide = "legend") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom")  
  
plot2 <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
'SB intention' = SB\_intention\_T1,  
'SB outcome\nexpectations' = SB\_outcomeExpectations\_T1,  
'SB self-efficacy\nand perceived\nbehavioural control' = SB\_sePbc\_T1,  
'SB descriptive\nnorm' = SB\_dnorm\_T1,  
'SB injunctive\nnorm' = SB\_inorm\_T1) %>%  
 # select(noquote(order(colnames(.)))) %>% # Orders columns alphabetically  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%   
 ggplot(aes(y = Variable)) +  
 ggridges::geom\_density\_ridges(aes(x = Value, fill = paste(Variable, intervention)),   
 alpha = .6, color = "black", from = 1, to = 7) +  
 labs(x = "",  
 y = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), labels = NULL) +  
 scale\_x\_continuous(expand = c(0.01, 0)) +  
 ggridges::scale\_fill\_cyclical(breaks = c("SB injunctive\nnorm 1", "SB injunctive\nnorm 0"),  
 labels = c('SB injunctive\nnorm 1' = "Intervention", 'SB injunctive\nnorm 0' = "Control"),  
 values = c("#ff0000", "#0000ff", "#ff8080", "#8080ff"),  
 name = "", guide = "legend") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(legend.position="bottom")  
  
gridExtra::grid.arrange(plot1, plot2, ncol = 2)  
## Picking joint bandwidth of 0.311  
## Warning: Removed 507 rows containing non-finite values  
## (stat\_density\_ridges).  
## Picking joint bandwidth of 0.317  
## Warning: Removed 507 rows containing non-finite values  
## (stat\_density\_ridges).



# CIBER (not included for now)

#p\_install("userfriendlyscience")  
  
# cat(names(df), sep = "\",\n\"")  
  
# Make variables numeric, and d as data frame  
d\_num <- df %>% mutate\_all(as.numeric)   
d\_num <- as.data.frame(d\_num)  
  
userfriendlyscience::CIBER(data = d\_num,  
 determinants = c(  
 "actCop\_T1",  
 "agrbct\_T1",  
 "amotivation\_T1",  
 "autonomous\_T1",  
 "big5\_agreeableness\_T1.1",  
 "big5\_conscientiousness\_T1.1",  
 "big5\_extraversion\_T1.1",  
 "big5\_neuroticism\_T1.1",  
 "big5\_openness\_T1.1",  
 "controlled\_T1",  
 "dnorm\_T1",  
 "frqbct\_T1",  
 "goal\_T1",  
 "inorm\_T1",  
 "intention\_T1",  
 "outcomeExpectations\_T1",  
 "opportunities\_T1",  
 "pbc\_T1",  
 "selfefficacy\_T1"),  
 targets = c("paT1", "fatpct\_T1"),  
 conf.level = list(means = 0.9999,  
 associations = 0.99)  
 )  
  
userfriendlyscience::CIBER(data = d\_num,  
 determinants = c(  
 "actCop\_T1",  
 "agrbct\_T1",  
 "frqbct\_T1",  
 "amotivation\_T1",  
 "autonomous\_T1",  
 "controlled\_T1",  
 "dnorm\_T1",  
 "intention\_T1",  
 "outcomeExpectations\_T1",  
 "opportunities\_T1",  
 "sePbc\_T1"),  
 targets = c("paT1", "fatpct\_T1"),  
 conf.level = list(means = 0.9999,  
 associations = 0.99)  
 )  
  
# to create variable list for CIBER:  
# cat(names(d), sep = "\",\n\"")  
  
userfriendlyscience::CIBER(data = d\_num,  
 determinants = c(  
 "agrbct\_01\_T1",  
 "agrbct\_02\_T1",  
 "agrbct\_03\_T1",  
 "agrbct\_04\_T1",  
 "agrbct\_05\_T1",  
 "agrbct\_06\_T1",  
 "agrbct\_07\_T1",  
 "agrbct\_08\_T1",  
 "agrbct\_09\_T1",  
 "agrbct\_10\_T1",  
 "frqbct\_01\_T1",  
 "frqbct\_02\_T1",  
 "frqbct\_03\_T1",  
 "frqbct\_04\_T1",  
 "frqbct\_05\_T1",  
 "frqbct\_06\_T1",  
 "frqbct\_07\_T1",  
 "frqbct\_08\_T1",  
 "frqbct\_09\_T1"),  
 targets = c("MVPA", "SB"),  
 leftAnchors = rep("", 19),  
 rightAnchors = rep("", 19))  
  
userfriendlyscience::CIBER(data = d\_num,  
 determinants = c(  
 "autonomous\_01\_T1",  
 "autonomous\_02\_T1",  
 "autonomous\_03\_T1",  
 "autonomous\_04\_T1",  
 "autonomous\_05\_T1",  
 "autonomous\_06\_T1",  
 "autonomous\_07\_T1",  
 "autonomous\_08\_T1",  
 "autonomous\_09\_T1"),  
 targets = c("MVPA", "SB"),  
 leftAnchors = rep("", 9),  
 rightAnchors = rep("", 9))  
  
userfriendlyscience::CIBER(data = d\_num,  
 determinants = c(  
 "intention\_01\_T1",  
 "intention\_02\_T1",  
 "selfefficacy\_01\_T1",  
 "selfefficacy\_02\_T1",  
 "pbc\_01\_T1",  
 "pbc\_02\_T1",  
 "pbc\_03\_T1",  
 "norm\_01\_T1",  
 "norm\_02\_T1",  
 "outcomeExpectations\_01\_T1",  
 "outcomeExpectations\_02\_T1",  
 "outcomeExpectations\_03\_T1",  
 "outcomeExpectations\_04\_T1",  
 "outcomeExpectations\_05\_T1",  
 "outcomeExpectations\_06\_T1",  
 "outcomeExpectations\_07\_T1",  
 "outcomeExpectations\_08\_T1",  
 "outcomeExpectations\_09\_T1",  
 "outcomeExpectations\_10\_T1",  
 "outcomeExpectations\_11\_T1",  
 "outcomeExpectations\_12\_T1"),  
 targets = c("MVPA", "SB"),  
 leftAnchors = rep("", 21),  
 rightAnchors = rep("", 21))

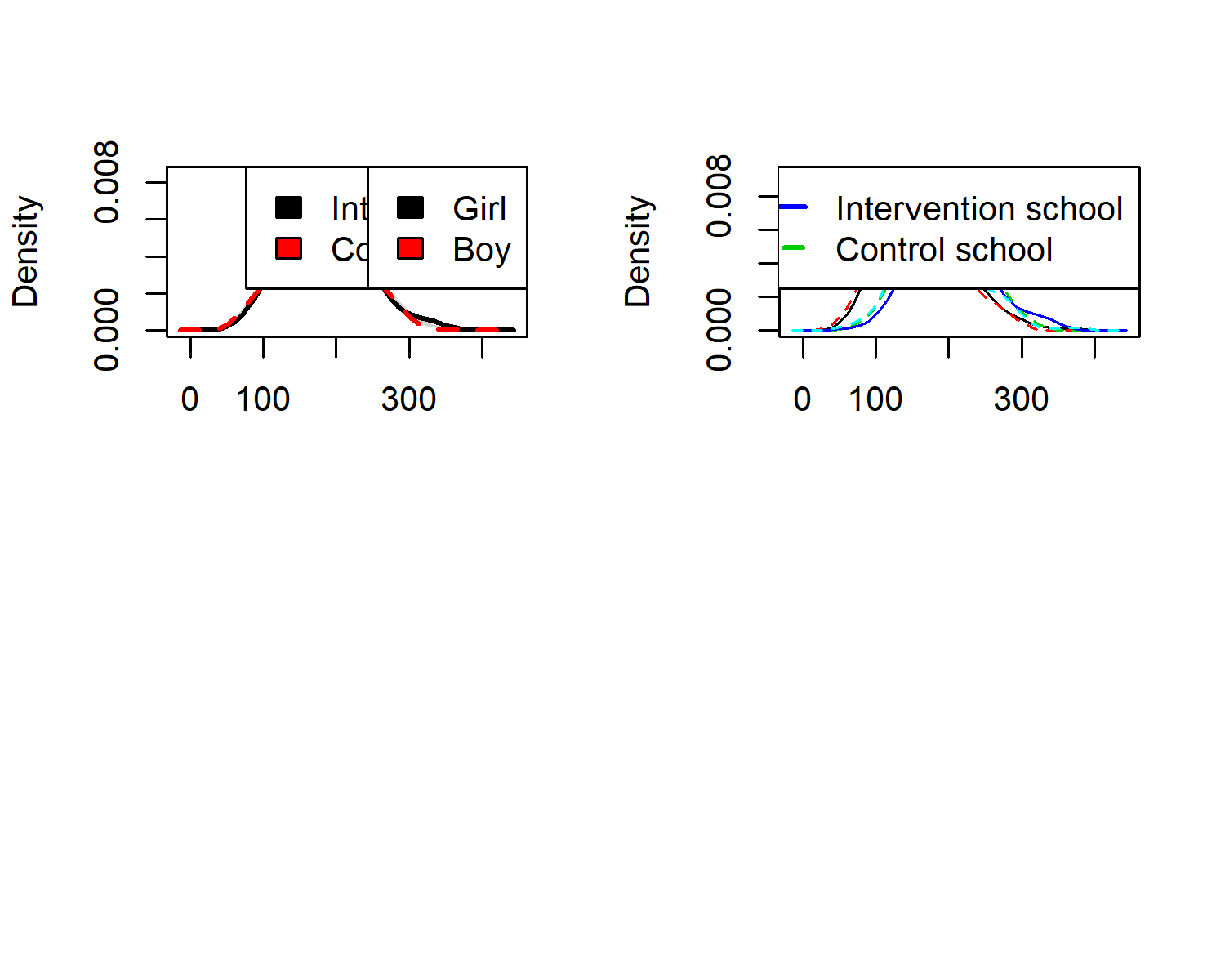
# Density plots

## Accelerometer-measured PA

Contrary to some other findings, in our sample girls were more active than boys.

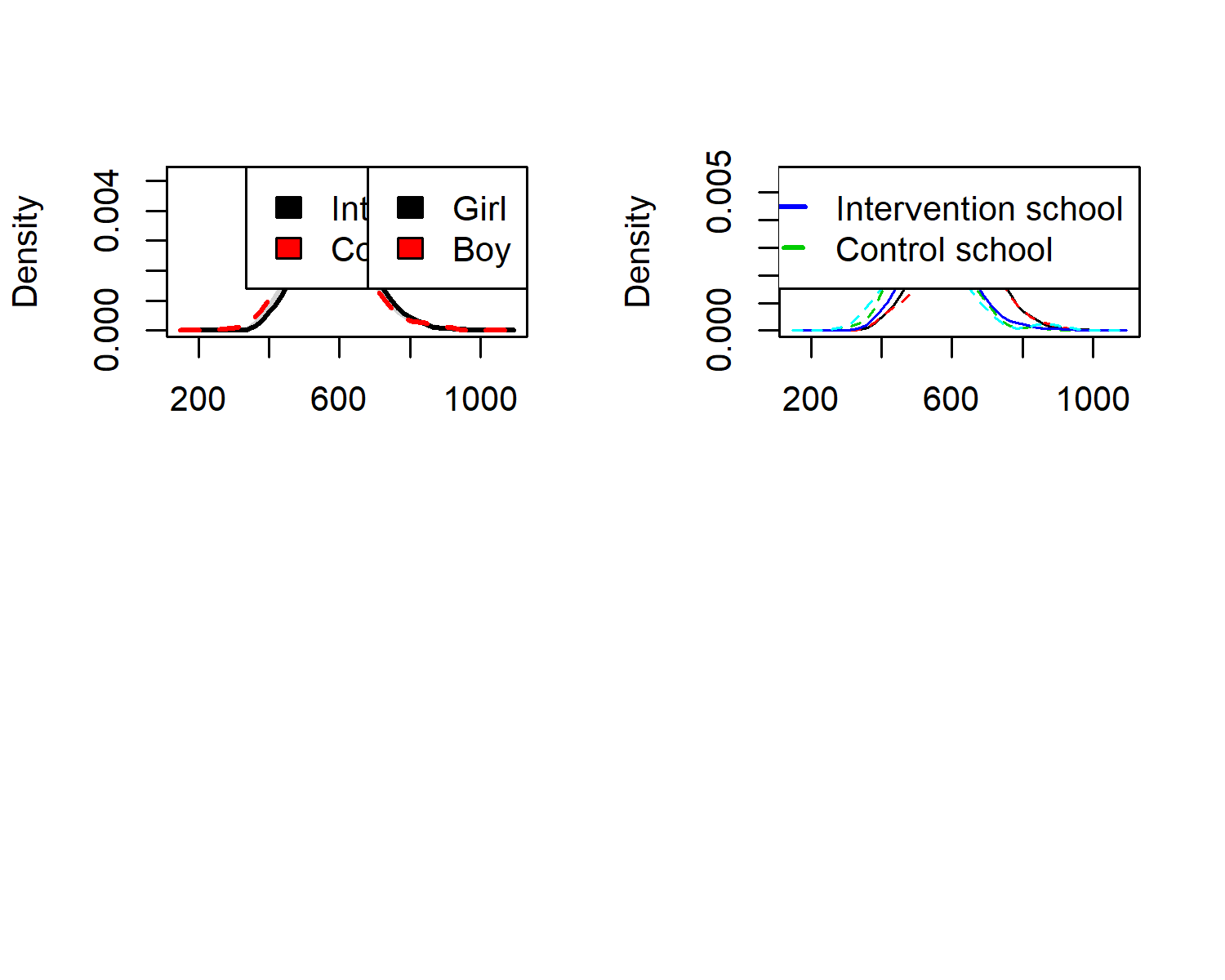
MVPAgirl\_df <- df %>% group\_by(girl) %>% select(girl, paAccelerometer\_T1) %>% summarise(mean = mean(paAccelerometer\_T1, na.rm = TRUE), median = median(paAccelerometer\_T1, na.rm = TRUE))  
MVPAgirl\_df[1, 2] - MVPAgirl\_df[2, 2]  
## mean  
## 1 16.81268  
MVPAgirl\_df[1, 3] - MVPAgirl\_df[2, 3]  
## median  
## 1 24.52857  
  
userfriendlyscience::meanDiff(df$girl, df$paAccelerometer\_T1)  
## Input variables:  
##   
## girl (grouping variable)  
## paAccelerometer\_T1 (dependent variable)  
## Mean 1 (1) = 190.36, sd = 48.6, n = 425  
## Mean 2 (0)= 173.55, sd = 59.39, n = 281  
##   
## Independent samples t-test (tested for equal variances, p < .001, so unequal variances)  
## (standard deviation used of largest sample, 48.6)  
##   
## 95% confidence intervals:  
## Absolute mean difference: [8.45, 25.17] (Absolute mean difference: 16.81)  
## Cohen's d for difference: [0.19, 0.5] (Cohen's d point estimate: 0.35)  
## Hedges g for difference: [0.19, 0.5] (Hedges g point estimate: 0.35)  
##   
## Achieved power for d=0.35: 0.9957 (for small: 0.756; medium: 1; large: 1)  
##   
## (secondary information (NHST): t[516] = 3.95, p < .001)  
##   
##   
## NOTE: because the t-test is based on unequal variances, the NHST p-value may be inconsistent with the confidence interval. Although this is not a problem, if you wish to ensure consistency, you can use parameter "var.equal = 'yes'" to force equal variances.

colfill <- c(1,2,3,4,5)  
  
# Create data frame  
densplot <- d  
levels(densplot$intervention) <- list("Intervention" = "1", "Control" = "0")  
levels(densplot$girl) <- list("Girl" = "1", "Boy" = "0")  
  
# This gives side-by-side plots  
 layout(matrix(c(1,2,3,3), nrow = 2, ncol = 2, byrow = TRUE), heights=c(2.5, 2))  
  
## Intervention vs. control  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(paAccelerometer\_T1, intervention) %>%   
 filter(complete.cases(.))  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
# Make plot  
sm.paAccelerometer\_T1\_1 <- sm.density.compare2(as.numeric(dens$paAccelerometer\_T1), as.factor(dens$intervention), xlab="", col=colfill, lty=c(1,2), bandcol='LightGray', model="equal", lwd=(c(2,2)))  
##   
## Test of equal densities: p-value = 0.25  
legend("topright", levels(dens$intervention), fill=c(1, 2))  
  
## Girls vs. boys  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(paAccelerometer\_T1, girl) %>%   
 filter(complete.cases(.))  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
# Make plot  
sm.paAccelerometer\_T1\_2 <- sm.density.compare2(as.numeric(dens$paAccelerometer\_T1), as.factor(dens$girl), xlab="", col=colfill, lty=c(1,2), bandcol='LightGray', model="equal", lwd=(c(2,2)))  
## Warning in min(x): no non-missing arguments to min; returning Inf  
  
## Warning in min(x): no non-missing arguments to min; returning Inf  
## Warning in max(x): no non-missing arguments to max; returning -Inf  
  
## Warning in max(x): no non-missing arguments to max; returning -Inf  
## Warning in min(x): no non-missing arguments to min; returning Inf  
## Warning in max(x): no non-missing arguments to max; returning -Inf  
## Error in seq.default(opt$xlim[1], opt$xlim[2], length = opt$ngrid): 'from' must be a finite number  
legend("topright", levels(dens$girl), fill=c(1, 2))  
  
## Differences in schools  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(paAccelerometer\_T1, school) %>%   
 filter(complete.cases(.))  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
  
# Intervention groups with solid thicker lines  
linetypes <- c(1,2,2,1,2)  
  
# Make plot  
sm.paAccelerometer\_T1\_3 <- sm::sm.density.compare(as.numeric(dens$paAccelerometer\_T1), dens$school, xlab="", col=colfill, lty=linetypes, model="none")  
legend("topright", c("Intervention school", "Control school"), col = c(4, 3), lty = c(1, 2), lwd = c(2, 2))



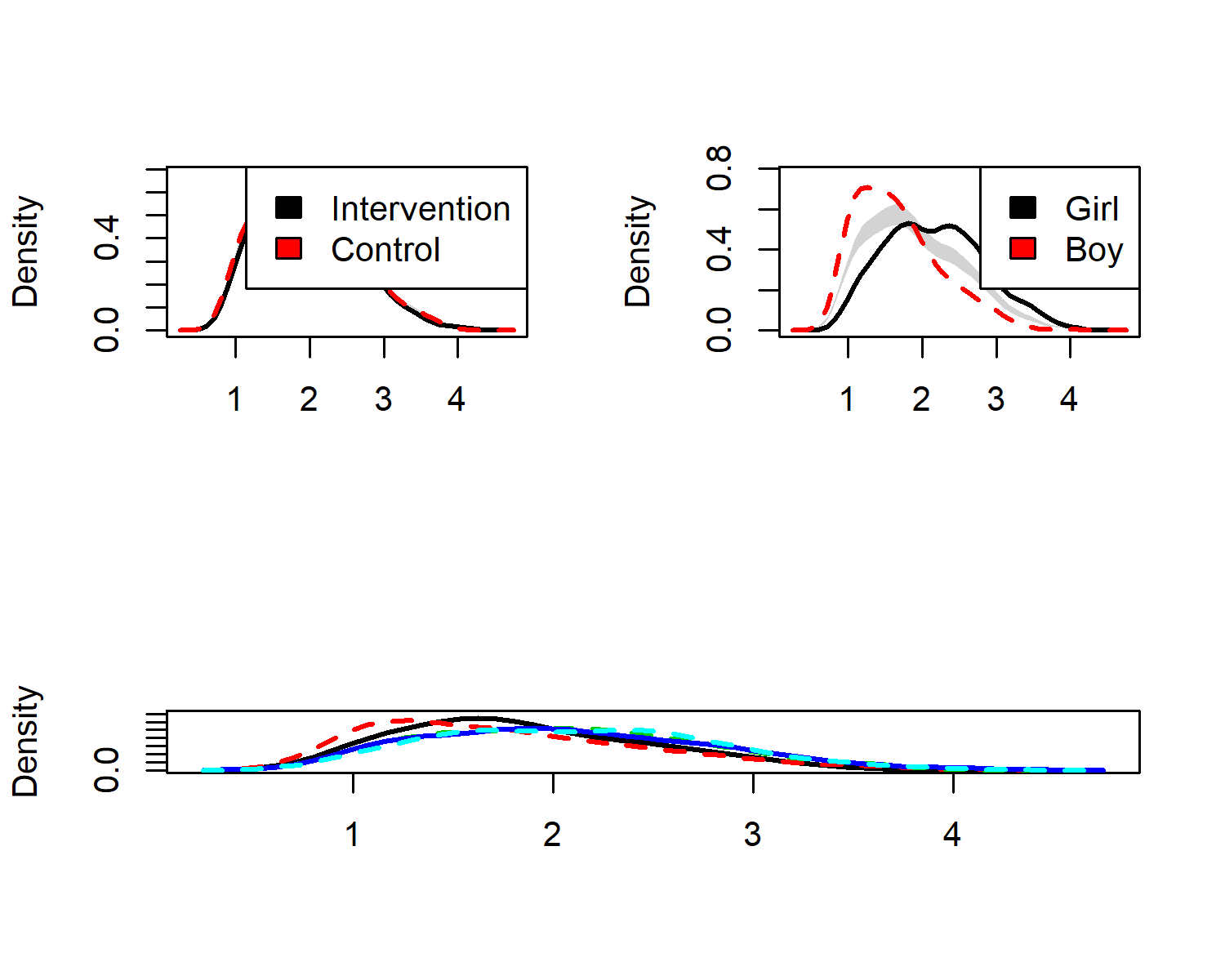
## Time spent sitting and lying down

colfill <- c(1,2,3,4,5)  
  
# Create data frame  
densplot <- d  
levels(densplot$intervention) <- list("Intervention" = "1", "Control" = "0")  
levels(densplot$girl) <- list("Girl" = "1", "Boy" = "0")  
  
# This gives side-by-side plots  
 layout(matrix(c(1,2,3,3), nrow = 2, ncol = 2, byrow = TRUE), heights=c(2.5, 2))  
  
## Intervention vs. control  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(sitLieAccelerometer\_T1, intervention) %>%   
 filter(complete.cases(.))  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
  
# Make plot  
sm.sitLieAccelerometer\_T1\_1 <- sm.density.compare2(as.numeric(dens$sitLieAccelerometer\_T1), as.factor(dens$intervention), xlab="", col=colfill, lty=c(1,2), bandcol='LightGray', model="equal", lwd=(c(2,2)))  
##   
## Test of equal densities: p-value = 0.09  
legend("topright", levels(dens$intervention), fill=c(1, 2))  
  
## Girls vs. boys  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(sitLieAccelerometer\_T1, girl) %>%   
 filter(complete.cases(.))  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
  
# Make plot  
sm.sitLieAccelerometer\_T1\_2 <- sm.density.compare2(as.numeric(dens$sitLieAccelerometer\_T1), as.factor(dens$girl), xlab="", col=colfill, lty=c(1,2), bandcol='LightGray', model="equal", lwd=(c(2,2)))  
## Warning in min(x): no non-missing arguments to min; returning Inf  
  
## Warning in min(x): no non-missing arguments to min; returning Inf  
## Warning in max(x): no non-missing arguments to max; returning -Inf  
  
## Warning in max(x): no non-missing arguments to max; returning -Inf  
## Warning in min(x): no non-missing arguments to min; returning Inf  
## Warning in max(x): no non-missing arguments to max; returning -Inf  
## Error in seq.default(opt$xlim[1], opt$xlim[2], length = opt$ngrid): 'from' must be a finite number  
legend("topright", levels(dens$girl), fill=c(1, 2))  
  
## Differences in schools  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(sitLieAccelerometer\_T1, school) %>%   
 filter(complete.cases(.))  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
  
# Intervention groups with solid thicker lines  
linetypes <- c(1,2,2,1,2)  
  
# Make plot  
sm.sitLieAccelerometer\_T1\_3 <- sm::sm.density.compare(as.numeric(dens$sitLieAccelerometer\_T1), dens$school, xlab="", col=colfill, lty=linetypes, model="none")  
legend("topright", c("Intervention school", "Control school"), col = c(4, 3), lty = c(1, 2), lwd = c(2, 2))



# Symptoms

colfill <- c(1,2,3,4,5)  
  
# Create data frame  
densplot <- df  
levels(densplot$intervention) <- list("Intervention" = "1", "Control" = "0")  
levels(densplot$girl) <- list("Girl" = "1", "Boy" = "0")  
  
# This gives side-by-side plots  
 layout(matrix(c(1,2,3,3), nrow = 2, ncol = 2, byrow = TRUE), heights=c(2.5, 2))  
  
## Intervention vs. control  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(symptom\_T1, intervention) %>% na.omit(densplot)  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
  
# Make plot  
sm.symptom\_T1\_1 <- sm.density.compare2(as.numeric(dens$symptom\_T1), as.factor(dens$intervention), xlab="", col=colfill, lty=c(1,2), bandcol='LightGray', model="equal", lwd=(c(2,2)))  
##   
## Test of equal densities: p-value = 0.59  
legend("topright", levels(dens$intervention), fill=c(1, 2))  
  
## Girls vs. boys  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select(symptom\_T1, girl) %>% na.omit(densplot)  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
  
# Make plot  
sm.symptom\_T1\_2 <- sm.density.compare2(as.numeric(dens$symptom\_T1), as.factor(dens$girl), xlab="", col=colfill, lty=c(1,2), bandcol='LightGray', model="equal", lwd=(c(2,2)))  
##   
## Test of equal densities: p-value = 0  
legend("topright", levels(dens$girl), fill=c(1, 2))  
  
## Differences in schools  
# Choose only the variables needed and drop NA  
dens <- densplot %>% select("Symptom sum score" = symptom\_T1, school) %>% na.omit(densplot)  
  
# Set random number generator for reproducibility of bootstrap test of equal densities  
set.seed(10)  
  
# Intervention groups with solid thicker lines  
linetypes <- c(1,2,2,1,2)  
linewidths <- c(2,1,1,2,1)  
  
# Make plot  
sm.symptom\_T1\_3 <- sm::sm.density.compare(dens$`Symptom sum score`, dens$school, xlab="", col=colfill, lty=linetypes, model="equal", lwd=linewidths)  
## Reference band available to compare two groups only.

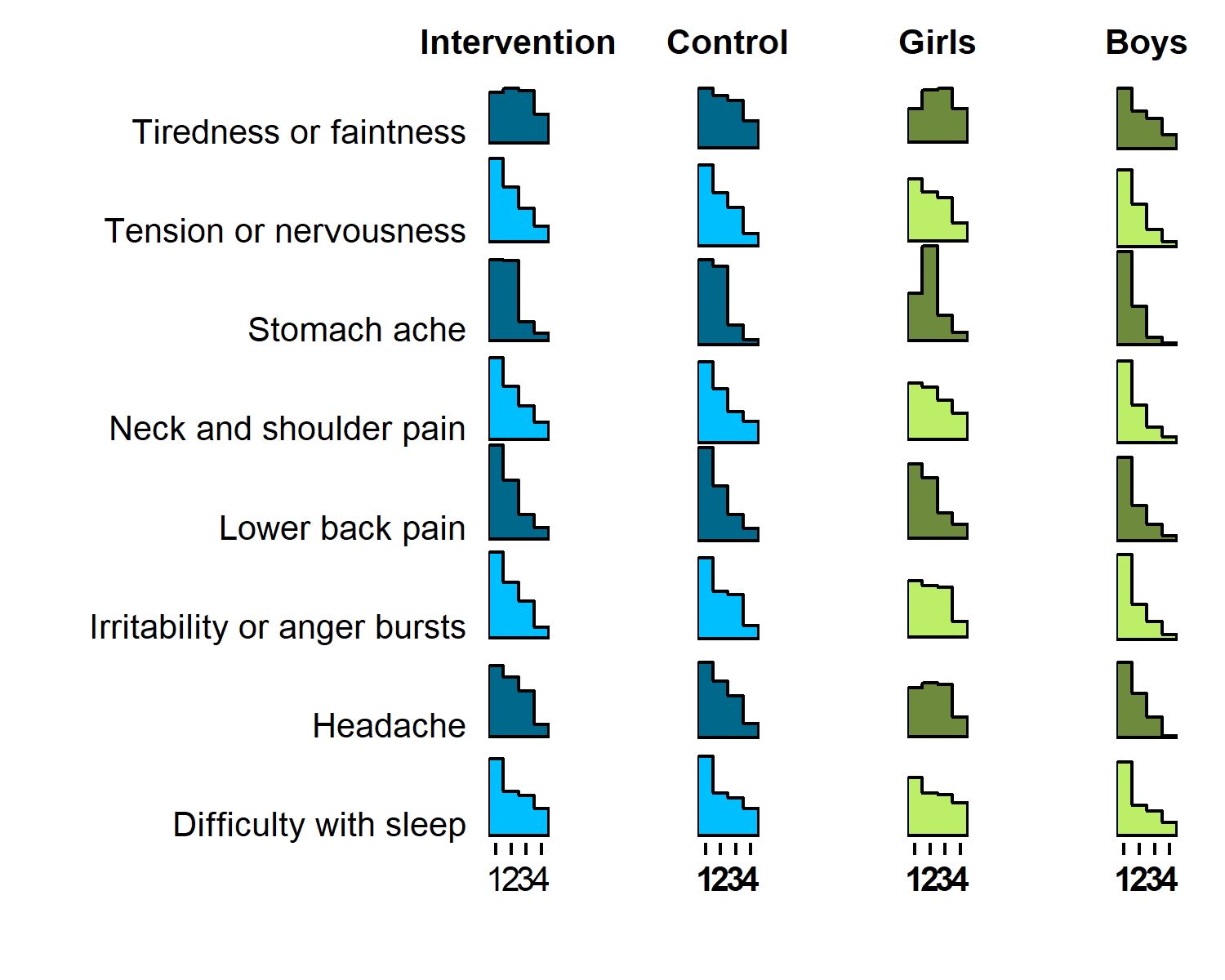


##   
## Test of equal densities: p-value = 0  
xlab = ""

## Symptom histograms

### intervention / gender

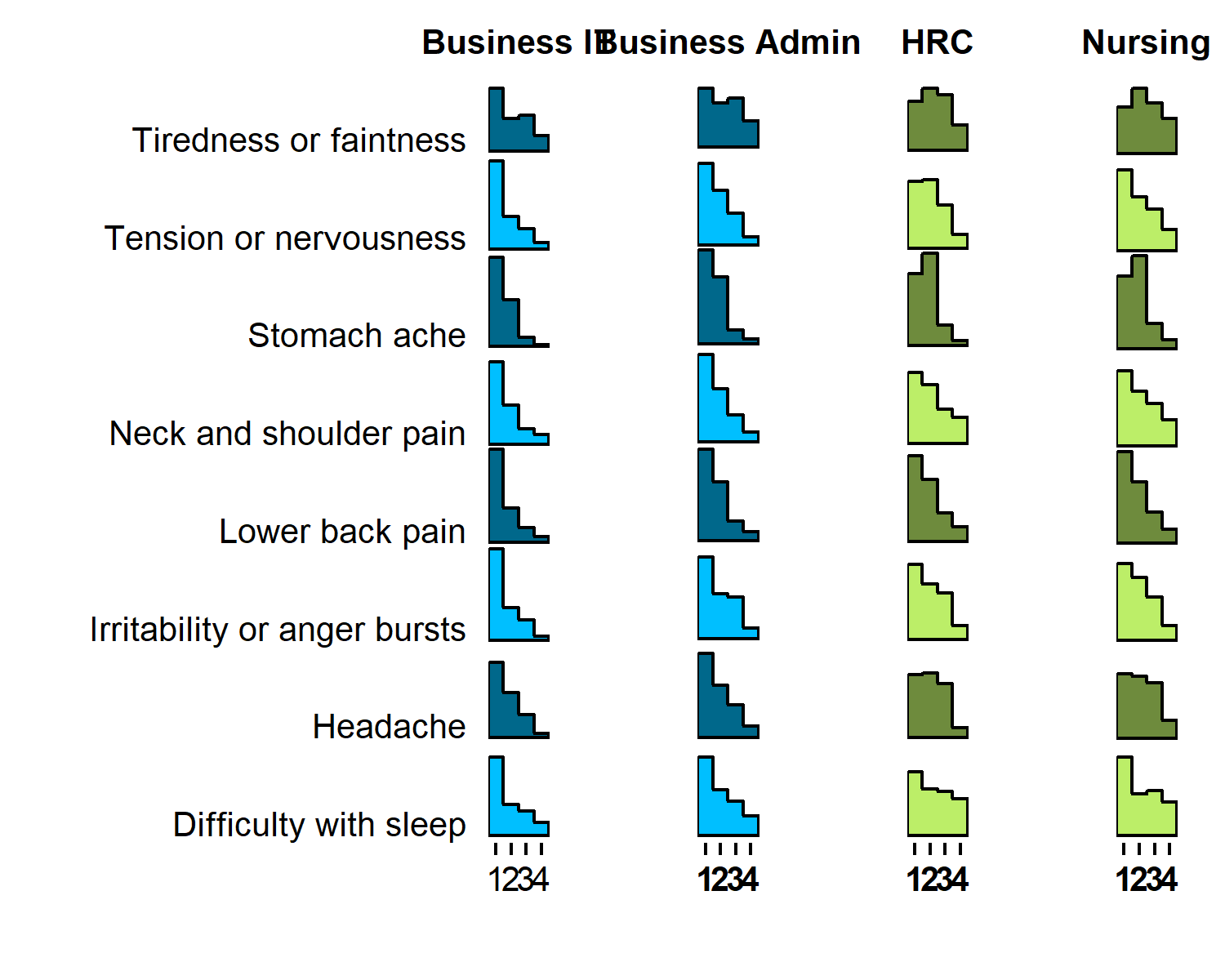
sympGirls <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 "Neck and shoulder pain" = symptom\_neckShoulderPain\_T1,  
 "Lower back pain" = symptom\_lowerBackPain\_T1,  
 "Stomach ache" = symptom\_stomachAche\_T1,  
 "Tension or nervousness" = symptom\_tensionNervousness\_T1,  
 "Irritability or anger bursts" = symptom\_irritabilityAngerbursts\_T1,  
 "Difficulty with sleep" = symptom\_sleepDifficulty\_T1,  
 "Headache" = symptom\_headAche\_T1,  
 "Tiredness or faintness" = symptom\_tirednessFaintness\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(girl == "1") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:4), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "", labels = NULL) +  
 ggridges::scale\_fill\_cyclical(values = c("darkolivegreen2", "darkolivegreen4")) +  
 labs(title = "Girls") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 plot.title = element\_text(hjust = 0.5, size = 10),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 4.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
sympBoys <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 "Neck and shoulder pain" = symptom\_neckShoulderPain\_T1,  
 "Lower back pain" = symptom\_lowerBackPain\_T1,  
 "Stomach ache" = symptom\_stomachAche\_T1,  
 "Tension or nervousness" = symptom\_tensionNervousness\_T1,  
 "Irritability or anger bursts" = symptom\_irritabilityAngerbursts\_T1,  
 "Difficulty with sleep" = symptom\_sleepDifficulty\_T1,  
 "Headache" = symptom\_headAche\_T1,  
 "Tiredness or faintness" = symptom\_tirednessFaintness\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(girl == "0") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:4), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "", labels = NULL) +  
 ggridges::scale\_fill\_cyclical(values = c("darkolivegreen2", "darkolivegreen4")) +  
 labs(title = "Boys") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 plot.title = element\_text(hjust = 0.5, size = 10),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 4.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
sympInt <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 "Neck and shoulder pain" = symptom\_neckShoulderPain\_T1,  
 "Lower back pain" = symptom\_lowerBackPain\_T1,  
 "Stomach ache" = symptom\_stomachAche\_T1,  
 "Tension or nervousness" = symptom\_tensionNervousness\_T1,  
 "Irritability or anger bursts" = symptom\_irritabilityAngerbursts\_T1,  
 "Difficulty with sleep" = symptom\_sleepDifficulty\_T1,  
 "Headache" = symptom\_headAche\_T1,  
 "Tiredness or faintness" = symptom\_tirednessFaintness\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(intervention == "1") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:4), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "") +  
 ggridges::scale\_fill\_cyclical(values = c("deepskyblue", "deepskyblue4")) +  
 labs(title = "Intervention") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 plot.title = element\_text(hjust = 0.5, size = 10),  
 axis.text=element\_text(size=10)) +  
coord\_cartesian(xlim = c(0.5, 4.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
sympCont <- df %>% dplyr::select(id,  
 intervention,  
 group,  
 school,  
 girl,  
 "Neck and shoulder pain" = symptom\_neckShoulderPain\_T1,  
 "Lower back pain" = symptom\_lowerBackPain\_T1,  
 "Stomach ache" = symptom\_stomachAche\_T1,  
 "Tension or nervousness" = symptom\_tensionNervousness\_T1,  
 "Irritability or anger bursts" = symptom\_irritabilityAngerbursts\_T1,  
 "Difficulty with sleep" = symptom\_sleepDifficulty\_T1,  
 "Headache" = symptom\_headAche\_T1,  
 "Tiredness or faintness" = symptom\_tirednessFaintness\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(intervention == "0") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:4), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "", labels = NULL) +  
 ggridges::scale\_fill\_cyclical(values = c("deepskyblue", "deepskyblue4")) +  
 labs(title = "Control") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 plot.title = element\_text(hjust = 0.5, size = 10),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 4.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
#grid.arrange(sympInt, sympGirls, sympCont, sympBoys, ncol = 2)  
  
# ("Seldom or never", "About once a month", "About once a week", "Almost daily")  
  
# This draws all histograms next to each other:  
grid::grid.newpage()  
grid::grid.draw(cbind(ggplotGrob(sympInt), ggplotGrob(sympCont), ggplotGrob(sympGirls), ggplotGrob(sympBoys), size = "last"))  
## Warning: Removed 198 rows containing non-finite values (stat\_binline).  
## Warning: Removed 93 rows containing non-finite values (stat\_binline).  
## Warning: Removed 197 rows containing non-finite values (stat\_binline).  
## Warning: Removed 94 rows containing non-finite values (stat\_binline).



# This draws 2 histograms per row:  
#grid.newpage()  
#grid.draw(rbind(cbind(ggplotGrob(sympInt), ggplotGrob(sympCont), size = "last"), cbind(ggplotGrob(sympGirls), ggplotGrob(sympBoys), size = "last")))

### educational track

sympHRC <- df %>% dplyr::select(id,  
 intervention,  
 track,  
 school,  
 girl,  
 "Neck and shoulder pain" = symptom\_neckShoulderPain\_T1,  
 "Lower back pain" = symptom\_lowerBackPain\_T1,  
 "Stomach ache" = symptom\_stomachAche\_T1,  
 "Tension or nervousness" = symptom\_tensionNervousness\_T1,  
 "Irritability or anger bursts" = symptom\_irritabilityAngerbursts\_T1,  
 "Difficulty with sleep" = symptom\_sleepDifficulty\_T1,  
 "Headache" = symptom\_headAche\_T1,  
 "Tiredness or faintness" = symptom\_tirednessFaintness\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(track == "HRC") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:4), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "", labels = NULL) +  
 ggridges::scale\_fill\_cyclical(values = c("darkolivegreen2", "darkolivegreen4")) +  
 labs(title = "HRC") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 plot.title = element\_text(hjust = 0.5, size = 10),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 4.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
sympNursing <- df %>% dplyr::select(id,  
 intervention,  
 track,  
 school,  
 girl,  
 "Neck and shoulder pain" = symptom\_neckShoulderPain\_T1,  
 "Lower back pain" = symptom\_lowerBackPain\_T1,  
 "Stomach ache" = symptom\_stomachAche\_T1,  
 "Tension or nervousness" = symptom\_tensionNervousness\_T1,  
 "Irritability or anger bursts" = symptom\_irritabilityAngerbursts\_T1,  
 "Difficulty with sleep" = symptom\_sleepDifficulty\_T1,  
 "Headache" = symptom\_headAche\_T1,  
 "Tiredness or faintness" = symptom\_tirednessFaintness\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(track == "Nursing") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:4), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "", labels = NULL) +  
 ggridges::scale\_fill\_cyclical(values = c("darkolivegreen2", "darkolivegreen4")) +  
 labs(title = "Nursing") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 plot.title = element\_text(hjust = 0.5, size = 10),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 4.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
sympIT <- df %>% dplyr::select(id,  
 intervention,  
 track,  
 school,  
 girl,  
 "Neck and shoulder pain" = symptom\_neckShoulderPain\_T1,  
 "Lower back pain" = symptom\_lowerBackPain\_T1,  
 "Stomach ache" = symptom\_stomachAche\_T1,  
 "Tension or nervousness" = symptom\_tensionNervousness\_T1,  
 "Irritability or anger bursts" = symptom\_irritabilityAngerbursts\_T1,  
 "Difficulty with sleep" = symptom\_sleepDifficulty\_T1,  
 "Headache" = symptom\_headAche\_T1,  
 "Tiredness or faintness" = symptom\_tirednessFaintness\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(track == "Business IT") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:4), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "") +  
 ggridges::scale\_fill\_cyclical(values = c("deepskyblue", "deepskyblue4")) +  
 labs(title = "Business IT") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 plot.title = element\_text(hjust = 0.5, size = 10),  
 axis.text=element\_text(size=10)) +  
coord\_cartesian(xlim = c(0.5, 4.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
sympBA <- df %>% dplyr::select(id,  
 intervention,  
 track,  
 school,  
 girl,  
 "Neck and shoulder pain" = symptom\_neckShoulderPain\_T1,  
 "Lower back pain" = symptom\_lowerBackPain\_T1,  
 "Stomach ache" = symptom\_stomachAche\_T1,  
 "Tension or nervousness" = symptom\_tensionNervousness\_T1,  
 "Irritability or anger bursts" = symptom\_irritabilityAngerbursts\_T1,  
 "Difficulty with sleep" = symptom\_sleepDifficulty\_T1,  
 "Headache" = symptom\_headAche\_T1,  
 "Tiredness or faintness" = symptom\_tirednessFaintness\_T1) %>%  
 tidyr::gather(key = Variable, value = Value, 6:ncol(.)) %>%  
 filter(track == "Business Admin") %>%   
 ggplot(aes(x = Value, y = Variable, group = Variable)) +  
 ggridges::geom\_density\_ridges2(aes(fill = Variable), stat = "binline", binwidth = 1, scale = 0.95) +  
 scale\_x\_continuous(breaks = c(1:4), expand = c(0, 0),  
 name = "") +  
 scale\_y\_discrete(expand = c(0.01, 0), name = "", labels = NULL) +  
 ggridges::scale\_fill\_cyclical(values = c("deepskyblue", "deepskyblue4")) +  
 labs(title = "Business Admin") +  
 guides(y = "none") +  
 ggridges::theme\_ridges(grid = FALSE) +  
 theme(axis.title.x = element\_text(hjust = 0.5),  
 axis.title.y = element\_text(hjust = 0.5),  
 plot.title = element\_text(hjust = 0.5, size = 10),  
 axis.text=element\_text(size=10, face="bold")) +  
coord\_cartesian(xlim = c(0.5, 4.5))  
## Warning: attributes are not identical across measure variables;  
## they will be dropped  
  
#grid.arrange(sympIT, sympHRC, sympBA, sympNursing, ncol = 2)  
  
# ("Seldom or never", "About once a month", "About once a week", "Almost daily")  
  
# This draws all histograms next to each other:  
grid::grid.newpage()  
grid::grid.draw(cbind(ggplotGrob(sympIT), ggplotGrob(sympBA), ggplotGrob(sympHRC), ggplotGrob(sympNursing), size = "last"))  
## Warning: Removed 26 rows containing non-finite values (stat\_binline).  
## Warning: Removed 49 rows containing non-finite values (stat\_binline).  
  
## Warning: Removed 49 rows containing non-finite values (stat\_binline).  
## Warning: Removed 141 rows containing non-finite values (stat\_binline).



# This draws 2 histograms per row:  
#grid.newpage()  
#grid.draw(rbind(cbind(ggplotGrob(sympIT), ggplotGrob(sympBA), size = "last"), cbind(ggplotGrob(sympHRC), ggplotGrob(sympNursing), size = "last")))