

# Back pain: four outcomes

Ty and Kaja

## Table of contents

<b>1</b>	<b>Approach</b>	<b>3</b>
<b>2</b>	<b>Set up</b>	<b>4</b>
2.1	Packages . . . . .	4
2.2	Constants . . . . .	4
<b>3</b>	<b>Data wrangling</b>	<b>7</b>
3.1	Read data . . . . .	7
3.2	Tidy data . . . . .	8
3.3	Impute missing values in compositions . . . . .	10
3.4	Compositions transformation to <i>ilrs</i> . . . . .	13
<b>4</b>	<b>Exploratory analysis</b>	<b>16</b>
4.1	Missing/NA value summaries . . . . .	16
4.2	Pairwise plots between <i>ilrs</i> and outcome variables . . . . .	18
<b>5</b>	<b>Statistical analysis</b>	<b>20</b>
5.1	Outcome 0: binary outcome of <code>Pain = "yes"</code> . . . . .	20
5.1.1	Model fit . . . . .	20
5.1.2	Model diagnostics . . . . .	22
5.1.3	Model predictions . . . . .	23
5.2	Note for outcomes 1 to 2 . . . . .	34
5.3	Outcome 1: <code>LBP_frequency_year</code> . . . . .	35
5.3.1	Model fit . . . . .	35
5.3.2	Model diagnostics . . . . .	38
5.3.3	Model predictions . . . . .	40
5.4	Outcome 2: <code>LBP_intensity_year</code> . . . . .	61
5.4.1	Model fit . . . . .	61
5.4.2	Model diagnostics . . . . .	73
5.4.3	Model predictions . . . . .	82



# 1 Approach

For each outcome, the simplest model that has appropriate fit will be sought. Models have been classified in rough ordering from “simplest” below using A, B, C, or D with A representing the common/easily understood models

<b>Outcome vari- able/Model</b>	<i>Multiple linear regression (A)</i>	<i>Ordinal logistic regression (B*)</i>	<i>Logistic regression (B)</i>	<i>Poisson/negative binomial regression (C)</i>	<i>Beta regression (D)</i>
<b>Binary</b>	-	-	+	-	-
<b>Ordinal</b>	-	+	+ (if outcome made binary)	-	-
<b>Values from 0 to 100</b>	+ (outcome potentially transformed)	- (if outcome made ordinal but bad option)	+ (if outcome made binary)	+	+

\*Probably “C” not “B” but is basically multiple logistic regressions performed with different dichotomisations of the order levels in the outcome

## 2 Set up

### 2.1 Packages

```
suppressPackageStartupMessages(suppressWarnings({

  library("dplyr") # tidyverse
  library("tidyr")
  library("readr")
  library("forcats")
  library("ggplot2")

  library("GGally") # additional ggplot-type plotting

  library("compositions")
  library("zCompositions") # this one for lr_EM

  library("performance") # model checking
  library("mice")          # missing data functions
  library("car")           # Anova() for comparing models
  library("knitr")         # kable() for pretty printing
  library("foreach")       # powerful looping

  library("boot")          # bootstrap confidence intervals
  library("tictoc")        # check time between tic() and toc()

}))
```

### 2.2 Constants

```
pred_comps <- c("Time_Sleep", "Time_Sedentary", "Time_LPA", "Time_MVPA")
(D <- length(pred_comps))
```

```
[1] 4
```

```
pred_covs <- c("age", "sex", "bmi", "stress", "smoking", "education", "ses")
outcs <-
  c(
    "LBP_frequency_year", "LBP_intensity_year",
```

```

    "LBP_intensity_month", "LBP_intensity_week"
  )

  # default RHS of model formulas
  # (rhs_formula <- paste(c(paste(pred_covs, collapse = " + "), "ilr"), collapse = " + "))
  (rhs_formula <- paste(pred_covs, collapse = " + "))

```

```
[1] "age + sex + bmi + stress + smoking + education + ses"
```

```

# this is the sequential binary partition matrix to be used for ilr creation
sbp1 <- matrix(
  c(
    1, 1, -1, -1,
    1, -1, 0, 0,
    0, 0, 1, -1
  ),
  ncol = 4, byrow = TRUE
)

# a way of creating ilr names automatically from SBP matrix
create_ilr_names <- function(sbp_matrix) {
  ilr_sbp_nms <- apply(sbp_matrix, 1, paste, collapse = "")
  ilr_sbp_nms <- gsub("-1", "-", ilr_sbp_nms)
  ilr_sbp_nms <- gsub("1", "+", ilr_sbp_nms)
  ilr_sbp_nms <- gsub("0", ".", ilr_sbp_nms)
  return(paste0("ilr(", ilr_sbp_nms, ")"))
}
create_ilr_names(sbp1)

```

```
[1] "ilr(++--)" "ilr(+-.)" "ilr(..+-)"
```

```

do_closure <- function(x, clo_val = 1) {
  return(clo_val * x / sum(x))
}
calc_comp_mean <- function(x, clo_val = 1) {
  unclosure_mean <- NULL
  if (is.null(dim(x))) {
    return(x)
  } else if (ncol(x) == 1) { # column matrix

```

```
    return(as.numeric(x))
  } else {
    uncloase_mean <- apply(x, 2, function(x) exp(mean(log(x))))
  }

  return(do_closure(uncloase_mean, clo_val = clo_val))
}
```

## 3 Data wrangling

### 3.1 Read data

```
bpd_col_spec <-  
  cols(  
    Time_Sleep = col_double(),  
    Time_Sedentary = col_double(),  
    Time_LPA = col_double(),  
    Time_MVPA = col_double(),  
    age = col_double(),  
    sex = col_character(),  
    bmi = col_double(),  
    stress = col_character(),  
    smoking = col_character(),  
    education = col_character(),  
    ses = col_character(),  
    LBP_sufferer = col_character(),  
    LBP_frequency_year = col_character(),  
    LBP_intensity_year = col_double(),  
    LBP_intensity_month = col_double(),  
    LBP_intensity_week = col_double()  
  )  
  
bpd <- read_csv("dat/bpd.csv", col_types = bpd_col_spec)  
# head(bpd)  
  
summary(bpd)
```

Time_Sleep	Time_Sedentary	Time_LPA	Time_MVPA
Min. : 87.14	Min. : 10.0	Min. : 9.857	Min. : 0.00
1st Qu.:400.00	1st Qu.: 305.4	1st Qu.: 363.286	1st Qu.: 12.00
Median :443.57	Median : 440.9	Median : 512.143	Median : 31.14
Mean :439.55	Mean : 451.9	Mean : 504.967	Mean : 43.55
3rd Qu.:480.71	3rd Qu.: 588.3	3rd Qu.: 642.286	3rd Qu.: 60.00
Max. :757.14	Max. :1100.1	Max. :1160.286	Max. :514.00

age	sex	bmi	stress
Min. :18.00	Length:2333	Min. :15.10	Length:2333
1st Qu.:38.00	Class :character	1st Qu.:21.95	Class :character

Median :49.00	Mode :character	Median :24.25	Mode :character
Mean :48.11		Mean :24.94	
3rd Qu.:58.00		3rd Qu.:27.15	
Max. :92.00		Max. :66.02	

smoking	education	ses	LBP_sufferer
Length:2333	Length:2333	Length:2333	Length:2333
Class :character	Class :character	Class :character	Class :character
Mode :character	Mode :character	Mode :character	Mode :character

LBP_frequency_year	LBP_intensity_year	LBP_intensity_month	LBP_intensity_week
Length:2333	Min. : 0.0	Min. : 0.00	Min. : 0.00
Class :character	1st Qu.: 19.0	1st Qu.: 9.00	1st Qu.: 0.00
Mode :character	Median : 30.0	Median : 25.00	Median : 10.00
	Mean : 33.8	Mean : 30.78	Mean : 20.36
	3rd Qu.: 49.0	3rd Qu.: 50.00	3rd Qu.: 31.00
	Max. :100.0	Max. :100.00	Max. :100.00
	NA's :673	NA's :673	NA's :673

### 3.2 Tidy data

```
# releval categories in LBP_frequency_year
y_lab <- "LBP_frequency_year"
sort(unique(bpd[[y_lab]]))
```

```
[1] "0days"           "1-7days"
[3] "31-90days"       "8-30days"
[5] "everyday"         "more_than90days_but_not_everyday"
```

```
bpd[[y_lab]] <-
  if_else(
    bpd[[y_lab]] == "more_than90days_but_not_everyday",
    "91+_not_evday",
    bpd[[y_lab]]
  )

# check
```



```
table(bpd[[y_lab]], useNA = "ifany")
```

	0days	1-7days	31-90days	8-30days	91+_not_evday
	673	760	146	451	203
everyday	100				

```
bpd[[y_lab]] <- factor(bpd[[y_lab]])
levels(bpd[[y_lab]])
```

```
[1] "0days"          "1-7days"         "31-90days"       "8-30days"
[5] "91+_not_evday"  "everyday"
```

```
lvls_ord <- c(1, 2, 4, 3, 5, 6)
# right order?
# levels(bpd[[y_lab]])[lvls_ord]
bpd[[y_lab]] <- lvls_reorder(bpd[[y_lab]], lvls_ord)
### right order?
levels(bpd[[y_lab]])
```

```
[1] "0days"          "1-7days"         "8-30days"        "31-90days"
[5] "91+_not_evday"  "everyday"
```

```
with(bpd, table(LBP_frequency_year, LBP_sufferer, useNA = "ifany"))
```

	LBP_sufferer	
LBP_frequency_year	no	yes
0days	673	0
1-7days	0	760
8-30days	0	451
31-90days	0	146
91+_not_evday	0	203
everyday	0	100

```
### comment these lines for sensitivity analysis
bpd$age <-
```

```

cut(
  bpd$age,
  breaks = c(17, 44, 64, 100),
  labels = c("1_younger", "2_middle", "3_older")
)
table(bpd$age)

```

1_younger	2_middle	3_older
896	1153	284

```

bpd$bmi <-
cut(
  bpd$bmi,
  breaks = c(15, 18.5, 25, 70),
  right = FALSE,
  labels = c("1_underweight", "2_normal", "3_overweight")
)
table(bpd$bmi)

```

1_underweight	2_normal	3_overweight
44	1309	980

### 3.3 Impute missing values in compositions

This code is thanks to Kaja!

Missing data is assumed to be below detectable threshold and imputed.

```

# Do I have zero values in my composition? (yes in MVPA)
### See: summary(bpd)

# We need to make compositions before we do the lrEM method. The most straightforward way
comp1 <- bpd[, pred_comps]

# How much participants have zero MVPA? 159 participants (6.8% of the sample)
missingSummary(comp1)

```

	missingType					
variable	NMV	BDL	MAR	MNAR	SZ	Err

```

Time_Sleep      2333    0    0    0    0    0
Time_Sedentary  2333    0    0    0    0    0
Time_LPA        2333    0    0    0    0    0
Time_MVPA       2174  159    0    0    0    0

```

```
sum(rowSums(is.na(comp1) | (comp1 < 0.1)), na.rm = FALSE)
```

```
[1] 159
```

```
sum(which_0 <- as.logical(rowSums(is.na(comp1) | (comp1 < 0.1))))
```

```
[1] 159
```

```

# these are 0 vals anywhere in composition (or NA)
bpd[which_0, pred_comps]

```

```
# A tibble: 159 x 4
```

```

  Time_Sleep Time_Sedentary Time_LPA Time_MVPA
    <dbl>         <dbl>    <dbl>    <dbl>
1     475         148.    817.        0
2     437.        826.    177.        0
3     479.        522.    440.        0
4     427.        429     584.        0
5     429.        468.    544.        0
6     424.        296.    720.        0
7     506.        727.    207.        0
8     456.        208     776.        0
9     475        607.    358.        0
10    272.        783.    384.        0

```

```
# i 149 more rows
```

```
# I have zeroes in MVPA - lrEM function will be applied
```

```
# ?lrEM
```

```
# what is the smallest time-use value above 0? [in minutes]
```

```
min(comp1[comp1 > 0])
```

```
[1] 0.1428571
```

```

thresh_detect <- 10 / 1440
# thresh_detect <- 0.01

comp1.a <- comp1 / 1440 # Create % based composition
dl <- c(rep(thresh_detect, times = D)) # threshold limit for the replacement

comp1.zr <- lrEM(comp1.a, label = 0, dl = dl) # conduct the lrEM Zero Replacement

```

No. iterations to converge: 6

```

comp1.zr <- as_tibble(comp1.zr * 1440)
# composition is larger than 1440 for those who have imputed MVPA
# (all behaviours will be proportionally downscaled to fit 1440 min when constructing the

# look at imputed values
comp1.zr[which_0, ]

```

# A tibble: 159 x 4

	Time_Sleep	Time_Sedentary	Time_LPA	Time_MVPA
	<dbl>	<dbl>	<dbl>	<dbl>
1	475	148.	817.	2.82
2	437.	826.	177.	3.21
3	479.	522.	440.	3.08
4	427.	429	584.	2.99
5	429.	468.	544.	3.01
6	424.	296.	720.	2.91
7	506.	727.	207.	3.21
8	456.	208	776.	2.87
9	475	607.	358.	3.12
10	272.	783.	384.	3.00

# i 149 more rows

```

# build dataset that contain imputed values for our 24-h composition

# add new compositions to other noncompositional data
# remove 24-h data from the dataset
# bpd <- subset(bpd, select = -c(id, Time_Sleep, Time_Sedentary, Time_LPA, Time_MVPA))
head(bpd[which_0, pred_comps])

```

```
# A tibble: 6 x 4
  Time_Sleep Time_Sedentary Time_LPA Time_MVPA
      <dbl>         <dbl>    <dbl>    <dbl>
1      475           148.    817.      0
2      437.          826.    177.      0
3      479.          522.    440.      0
4      427.          429     584.      0
5      429.          468.    544.      0
6      424.          296.    720.      0

bpd <- bpd[, !(colnames(bpd) %in% pred_comps)] # remove ori time-use cols
bpd <- bind_cols(comp1.zr, bpd) # add imputed 24-h data
head(bpd[which_0, pred_comps])
```

```
# A tibble: 6 x 4
  Time_Sleep Time_Sedentary Time_LPA Time_MVPA
      <dbl>         <dbl>    <dbl>    <dbl>
1      475           148.    817.    2.82
2      437.          826.    177.    3.21
3      479.          522.    440.    3.08
4      427.          429     584.    2.99
5      429.          468.    544.    3.01
6      424.          296.    720.    2.91
```

### 3.4 Compositions transformation to *ilrs*

The below function will allow us to automatically add *ilrs* to a dataset

```
add_ilrs_to_data <- function(dataset, comp_vars = pred_comps, sbp_matrix = sbp1) {

  # the time-use composition
  comp <- dataset[, comp_vars]
  comp <- acomp(comp) # designate it as a compositional variable

  # define sequential binary partition (SBP)
  psi1 <- gsi.buildilrBase(t(sbp_matrix)) # The orthonormal matrix

  # find the mean composition
  (m <- mean(comp)) # comp has been designated as acomp, therefore R knows it's a composition
  # cat(
  #   "\nThis is the compositional mean [in mins] of the columns ("
```

```

#   paste(comp_vars, collapse = ", "),
#   ")\n\n",
#   sep = ""
# )
# print(clo(m, total = 1440)) # to look at the mean in minutes/day.
# cat("\n\n")

# create isometric log ratios (ilr.1) using the above SBP and orthonormal basis V=psi1.
ilrs_from_comp <- ilr(comp, V = psi1)
colnames(ilrs_from_comp) <- create_ilr_names(sbp_matrix)
# colnames(ilrs_from_comp) <- paste0("coord", 1:(length(comp_vars) - 1))
dataset$ilr <- ilrs_from_comp

return(dataset)

}

# use function: creates the ilr columns nested in the single column "ilr"
bpd <- add_ilrs_to_data(bpd)
# check
bpd[, c("ilr", pred_comps)]

# A tibble: 2,333 x 5
  ilr[, "ilr(++--)" [, "ilr(+--.)" Time_Sleep Time_Sedentary Time_LPA Time_MVPA
    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1      1.04      0.355      435      263.      722.      19.7
2      2.29     -0.401      410      723.      297.      10.3
3      1.57     -0.336      426.      685.      285.      44.3
4      1.10     -0.0811      486.      545.      318.      91.3
5      0.801      0.220      484.      355.      536.      64.6
6      1.71     -0.134      494.      598.      318.      30.4
7      1.03     -0.0333      447.      469.      467.      57.6
8      1.29      0.189      546.      418.      435.      40
9      1.69     -0.243      452.      638      320.      30.4
10     1.36     -0.329      354.      565.      494.      26.7
# i 2,323 more rows
# i 1 more variable: ilr[3] <rmult>

# also create version of data without the nested ilrs
bpd_clean <- as.data.frame(bpd)
bpd_clean$ilr <- NULL # remove nested cols

```

```
bpd_clean <- cbind(bpd_clean, as.data.frame(bpd$ilr))
```

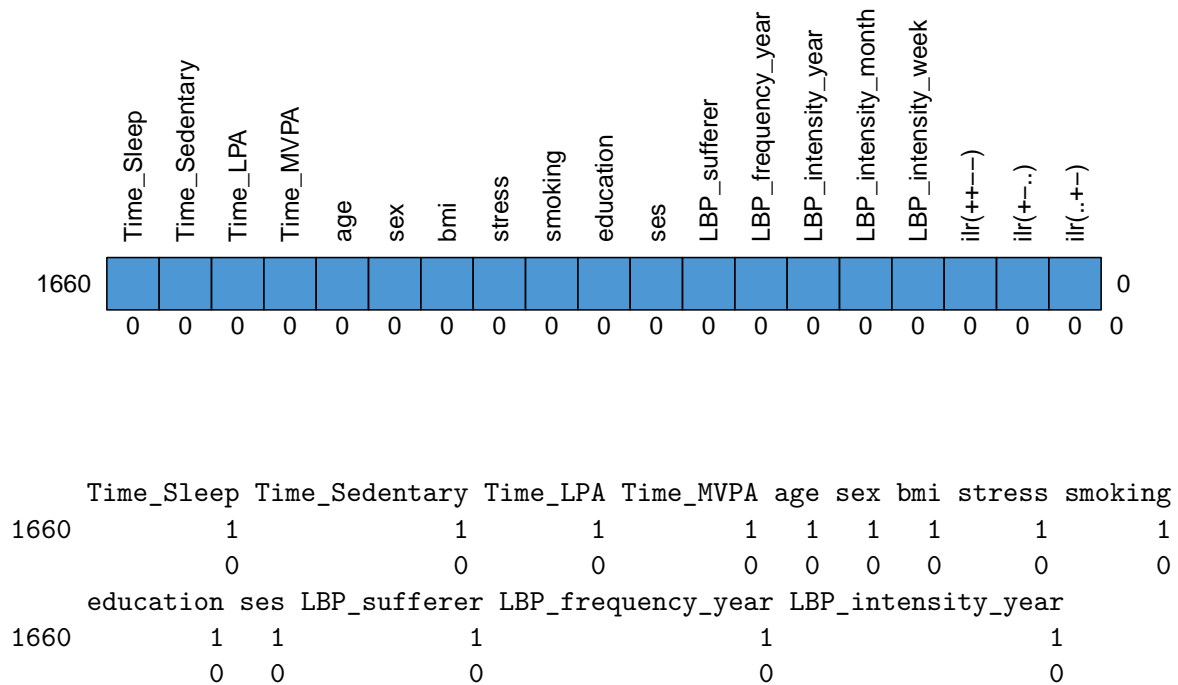
## 4 Exploratory analysis

### 4.1 Missing/NA value summaries

```
### Missing data summary for LBP sufferers
bpd_clean %>%
  dplyr::filter(LBP_sufferer == "yes") %>%
  md.pattern(., rotate.names = TRUE)
```

```

/\      /\
{  '----'  }
{  0    0  }
==> V <== No need for mice. This data set is completely observed.
\  \\/  /
  '-----'
```





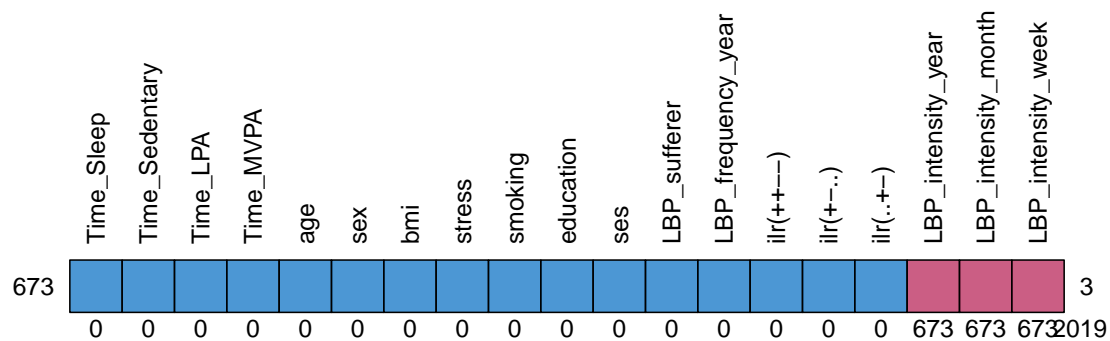
	LBP_intensity_month	LBP_intensity_week	ilr(++--)	ilr(+--.)	ilr(..+-)
1660	1	1	1	1	1 0
	0	0	0	0	0 0

```
### Missing data summary for _non_ LBP sufferers
```

```
bpd_clean %>%
```

```
  dplyr::filter(LBP_sufferer == "no") %>%
```

```
  md.pattern(., rotate.names = TRUE)
```



	Time_Sleep	Time_Sedentary	Time_LPA	Time_MVPA	age	sex	bmi	stress	smoking
673	1		1	1	1	1	1	1	1
	0		0	0	0	0	0	0	0

	education	ses	LBP_sufferer	LBP_frequency_year	ilr(++--)	ilr(+--.)	ilr(..+-)
673	1	1	1	1	1	1	1
	0	0	0	0	0	0	0

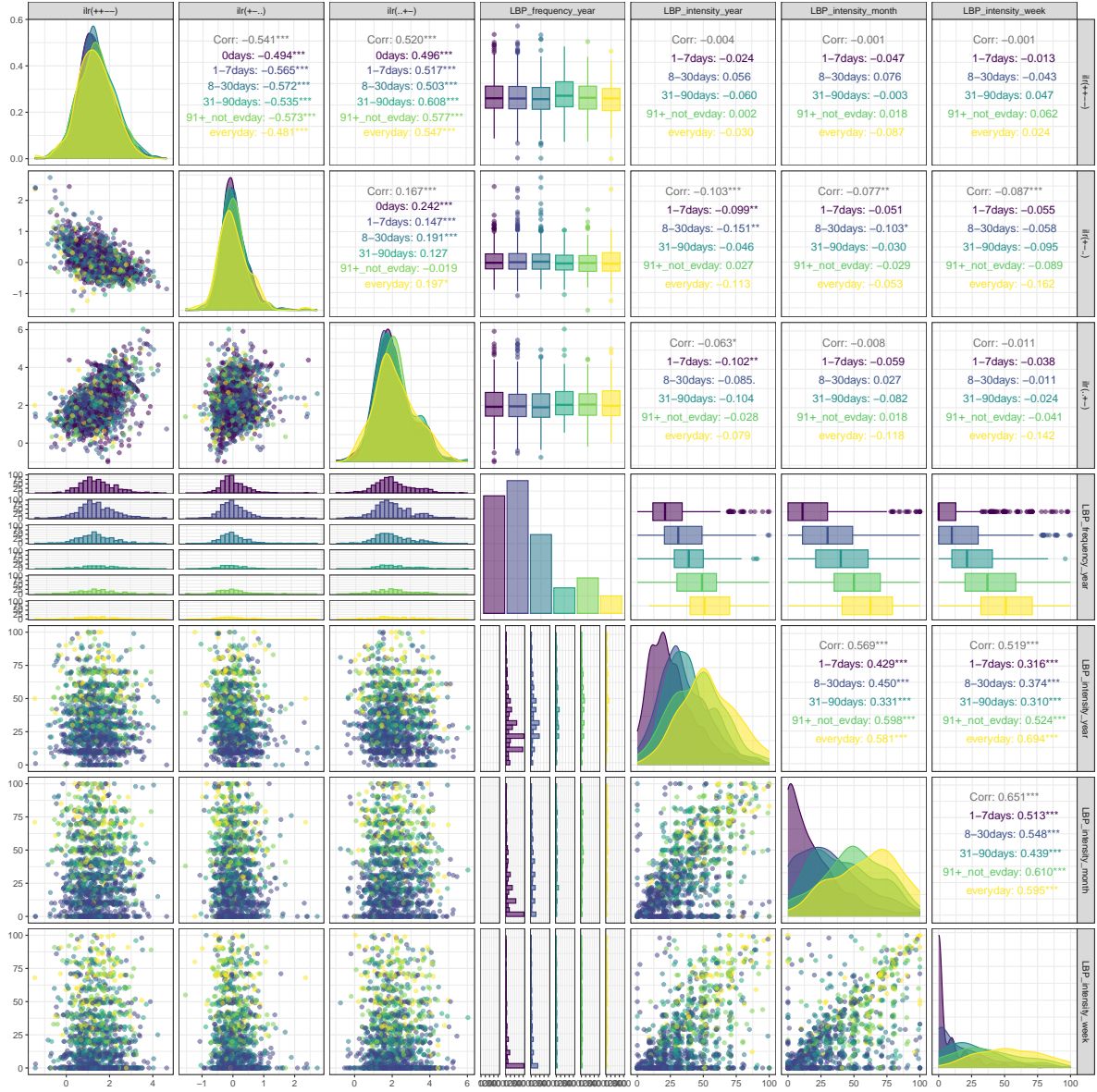
	LBP_intensity_year	LBP_intensity_month	LBP_intensity_week
673	0	0	0 3
	673	673	673 2019

```
### ==> data doesn't have mistiness for analysis
```

## 4.2 Pairwise plots between *ilrs* and outcome variables

```
### plot pairwise comparisons of time-use and outcomes
if (FALSE) { # takes 30 sec
  suppressWarnings({
    bpd_clean %>%
      dplyr::select(all_of(pred_comps), all_of(outcs)) %>%
      ggpairs(
        .,
        progress = FALSE,
        ggplot2::aes(
          colour = LBP_frequency_year,
          fill = LBP_frequency_year,
          alpha = 0.25
        )
      ) +
      theme_bw() +
      scale_colour_viridis_d()+
      scale_fill_viridis_d()
  })
}

### plot pairwise comparisons of _ilrs_ and outcomes
if (TRUE) { # takes 30 sec
  suppressWarnings({
    bpd_clean %>%
      dplyr::select(starts_with("ilr"), all_of(outcs)) %>%
      ggpairs(
        .,
        progress = FALSE,
        ggplot2::aes(
          colour = LBP_frequency_year,
          fill = LBP_frequency_year,
          alpha = 0.25
        )
      ) +
      theme_bw() +
      scale_colour_viridis_d()+
      scale_fill_viridis_d()
  })
}
```



## 5 Statistical analysis

### 5.1 Outcome 0: binary outcome of Pain = "yes"

#### 5.1.1 Model fit

```
bpd <-  
  bpd %>%  
    mutate(lbp_occurre = as.integer(LBP_sufferer == "yes"))  
  
(this_outcome <- "lbp_occurre")
```

```
[1] "lbp_occurre"
```

```
# (mod_form_null <- as.formula(paste0(this_outcome, " ~ ", rhs_formula)))  
(mod_form_ilrs <- as.formula(paste0(this_outcome, " ~ ", rhs_formula, " + ilr")))
```

```
lbp_occurre ~ age + sex + bmi + stress + smoking + education +  
  ses + ilr
```

```
table(bpd[, this_outcome], useNA = "ifany")
```

```
lbp_occurre  
  0    1  
673 1660
```

```
# logistic regression model __with__ ilrs  
bpd_occurre_ilrs <- glm(mod_form_ilrs, data = bpd, family = binomial())  
summary(bpd_occurre_ilrs)
```

Call:

```
glm(formula = mod_form_ilrs, family = binomial(), data = bpd)
```

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )	
(Intercept)	1.053768	0.384211	2.743	0.006094	**
age2_middle	0.340378	0.103168	3.299	0.000969	***

```

age3_older      0.262018    0.159537    1.642 0.100513
sex2_male       0.062784    0.111862    0.561 0.574618
bmi2_normal     0.164453    0.327142    0.503 0.615178
bmi3_overweight 0.378117    0.333707    1.133 0.257180
stress2_stressed 0.407584    0.103641    3.933 8.4e-05 ***
smoking2_nonsmoker -0.105372    0.126368   -0.834 0.404365
education2_higher -0.274614    0.112119   -2.449 0.014313 *
ses2_middle     -0.328406    0.177563   -1.850 0.064383 .
ses3_higher     -0.723348    0.215902   -3.350 0.000807 ***
ilrilr(++--)    -0.083947    0.103592   -0.810 0.417735
ilrilr(+--.)    -0.078951    0.177757   -0.444 0.656933
ilrilr(..+--)   0.005253    0.072374    0.073 0.942138
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

(Dispersion parameter for binomial family taken to be 1)

```

Null deviance: 2803.2  on 2332  degrees of freedom
Residual deviance: 2736.7  on 2319  degrees of freedom
AIC: 2764.7

```

Number of Fisher Scoring iterations: 4

```
Anova(bpd_occurr_ilrs)
```

Analysis of Deviance Table (Type II tests)

```

Response: lbp_occurr
      LR Chisq Df Pr(>Chisq)
age      11.0155  2  0.004055 **
sex       0.3161  1  0.573932
bmi       5.0563  2  0.079808 .
stress    15.7925  1  7.068e-05 ***
smoking    0.7022  1  0.402035
education  6.1009  1  0.013511 *
ses      12.5248  2  0.001907 **
ilr       1.2749  3  0.735097
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

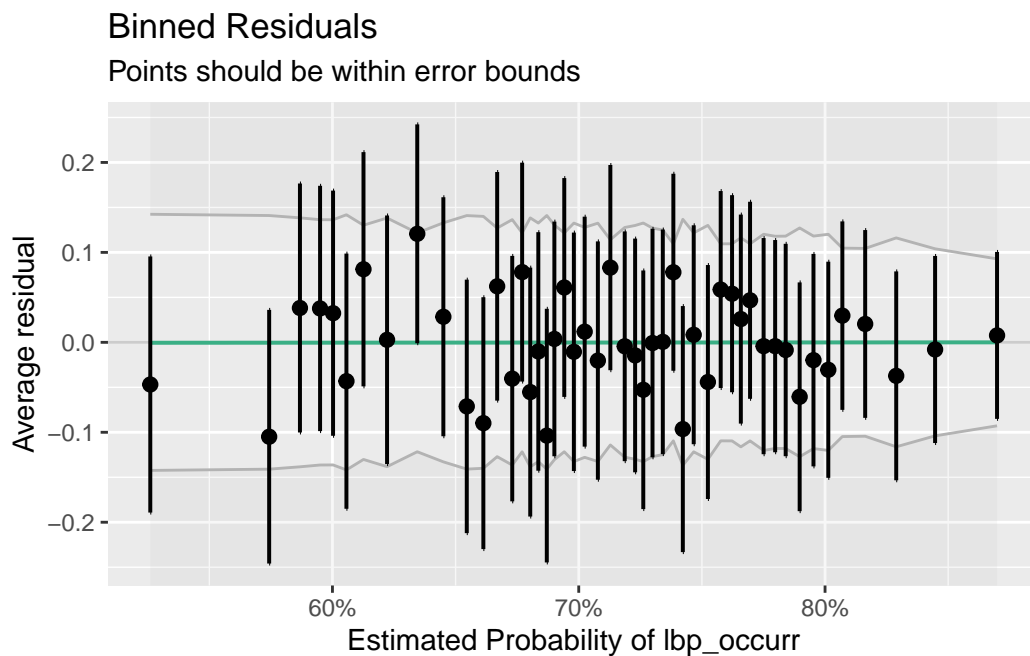
```

### 5.1.2 Model diagnostics

```
### check binned residuals are acceptable
# From the help file:
# Binned residual plots are achieved by "dividing the data into categories
# (bins) based on their fitted values, and then plotting the average residual
# versus the average fitted value for each bin." (Gelman, Hill 2007: 97).
# If the model were true, one would expect about 95% of the residuals to
# fall inside the error bounds.
bin_res_overall <- binned_residuals(bpd_occurr_ilrs)
bin_res_overall
```

Ok: About 100% of the residuals are inside the error bounds.

```
plot(bin_res_overall)
```



### 5.1.3 Model predictions

```
# create dataset for predictions
newdata <-
  bpd %>%
  dplyr::select(all_of(pred_covs), ilr) %>%
  distinct(pick(all_of(pred_covs)), .keep_all = TRUE) %>%
  arrange(pick(all_of(pred_covs)))

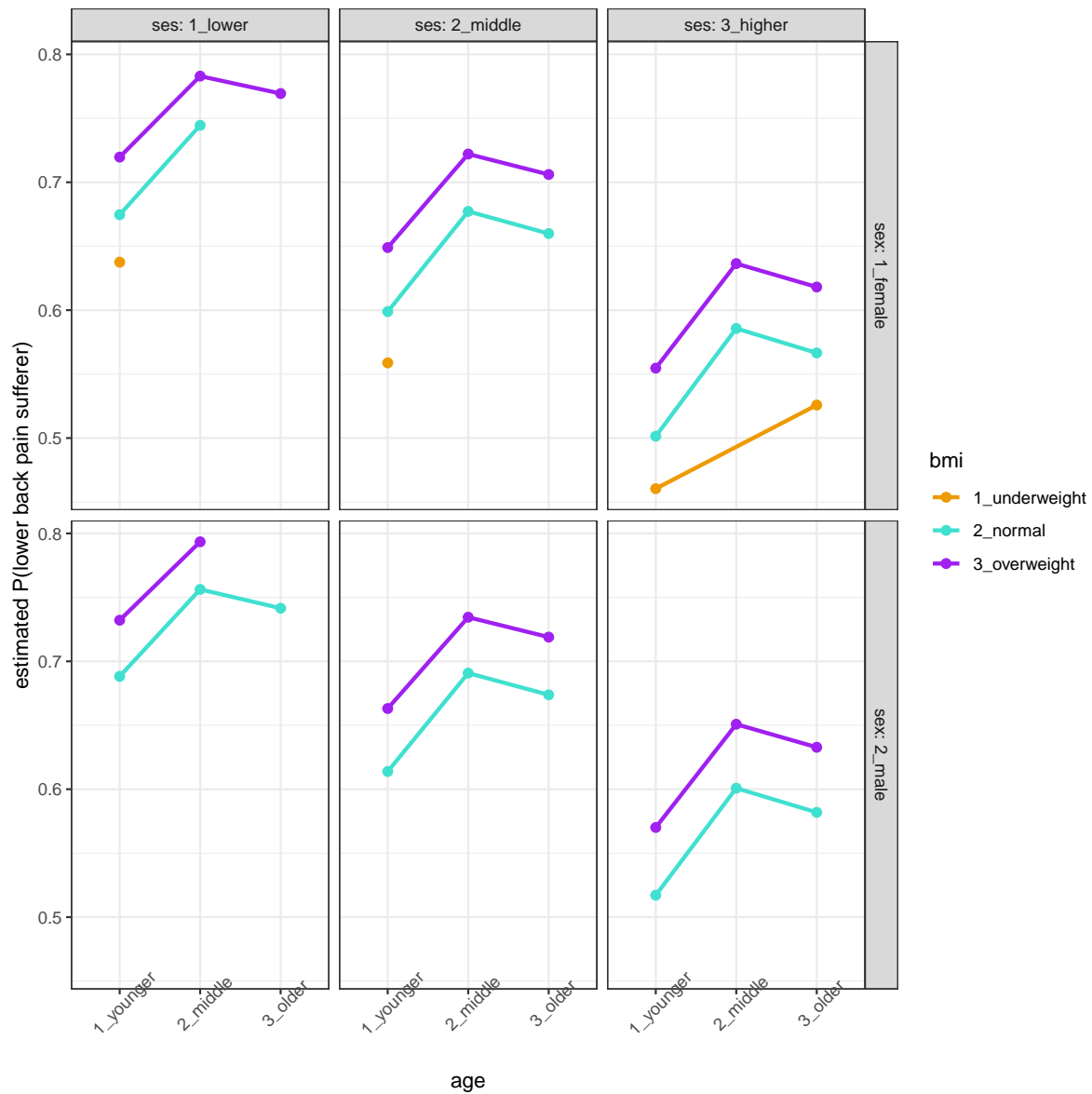
mean_ilr <- mean(bpd$ilr)
dev_null <- foreach(i = 1:nrow(newdata)) %do% {
  newdata$ilr[i, ] <- mean_ilr
}

# make preds and then put in long format for ggplot
predictions_probs <-
  cbind(
    `P(LBP)` = predict(bpd_occrr_ilrs, newdata, type = "response"),
    newdata
  ) %>%
  dplyr::select(-ilr)

# predictions_probs

## model predictions for specific values
predictions_probs %>%
  dplyr::filter(
    # sex == "1_female",
    stress == "1_normal",
    smoking == "2_nonsmoker",
    education == "2_higher",
    # ses == "2_middle"
  ) %>%
  ggplot(., aes(age, `P(LBP)`, group = bmi)) +
  geom_line(aes(colour = bmi), linewidth = 1) +
  geom_point(aes(colour = bmi), size = 2) +
  facet_grid(sex~ ses, labeller = label_both) +
  labs(x = "age", y = "estimated P(lower back pain sufferer)") +
  theme_bw() +
  scale_color_manual(values = c("orange2", "turquoise", "purple")) +
```

```
theme(axis.text.x = element_text(angle = 45))
```



```
# create a RHS of regression equation dataset for time-reallocation
predict_basis <-
  bpd %>%
  dplyr::select(all_of(pred_covs), all_of(pred_comps)) %>%
```



```

dplyr::filter(
  age == "2_middle",
  sex == "1_female",
  stress == "1_normal",
  smoking == "2_nonsmoker",
  education == "2_higher",
  ses == "2_middle",
  bmi == "2_normal"
)

### continuous scenario
# predict_basis$age <- mean(predict_basis$age)

(predict_basis <-
  predict_basis %>%
  distinct(across(all_of(pred_covs)), .keep_all = TRUE) %>%
  as.data.frame())

  age      sex      bmi  stress      smoking education      ses Time_Sleep
1 2_middle 1_female 2_normal 1_normal 2_nonsmoker 2_higher 2_middle 546.4286
Time_Sedentary Time_LPA Time_MVPA
1      418.2857 435.1429      40

# compositional mean: geometric mean to closure
# (comp_mean <- mean(acomp(bpd[, pred_comps])))
(comp_mean <- calc_comp_mean(bpd[, pred_comps], clo_val = 1440))

Time_Sleep Time_Sedentary      Time_LPA      Time_MVPA
474.36588      439.73363      499.43836      26.46213

predict_basis0 <- predict_basis
predict_basis0[, pred_comps] <- comp_mean

predict_basis0

  age      sex      bmi  stress      smoking education      ses Time_Sleep
1 2_middle 1_female 2_normal 1_normal 2_nonsmoker 2_higher 2_middle 474.3659
Time_Sedentary Time_LPA Time_MVPA
1      439.7336 499.4384 26.46213

```

```

# +15 minutes to Time_MVPA and -15 minutes from Time_Sedentary
comp_mean_changed <- comp_mean
comp_mean_changed["Time_MVPA"] <- comp_mean_changed["Time_MVPA"] + 15
comp_mean_changed["Time_Sedentary"] <- comp_mean_changed["Time_Sedentary"] - 15
# check
comp_mean_changed - comp_mean

```

Time_Sleep	Time_Sedentary	Time_LPA	Time_MVPA
0	-15	0	15

```

predict_basis1 <- predict_basis
predict_basis1[, pred_comps] <- comp_mean_changed

pred_df <- rbind(predict_basis0, predict_basis1)
pred_df <- add_ilrs_to_data(pred_df, comp_vars = pred_comps, sbp_matrix = sbp1)
pred_df

```

	age	sex	bmi	stress	smoking	education	ses	Time_Sleep
1	2_middle	1_female	2_normal	1_normal	2_nonsmoker	2_higher	2_middle	474.3659
2	2_middle	1_female	2_normal	1_normal	2_nonsmoker	2_higher	2_middle	474.3659
	Time_Sedentary	Time_LPA	Time_MVPA	ilr.1	ilr.2	ilr.3		
1	439.7336	499.4384	26.46213	1.37947472	0.05360560	2.07731687		
2	424.7336	499.4384	41.46213	1.13758831	0.07814711	1.75977934		

```

predict(bpd_occurr_ilrs, pred_df, type = "link")

```

	1	2
	0.7410846	0.7577845

```

# ratio of odds ratios
exp(diff(predict(bpd_occurr_ilrs, pred_df, type = "link")))

```

	2
	1.01684

```

get_pred_diff <- function(mod, new_dat) {
  log_odds_pred <- predict(mod, new_dat, type = "link")
  odds_ratio_ratio <- exp(log_odds_pred[2] - log_odds_pred[1])
  return(odds_ratio_ratio)
}
(est_v1 <- get_pred_diff(bpd_occurr_ilrs, pred_df))

```

2  
1.01684

```

fit_mod_boot <- function(data, i, pred_dat) {

  this_dat <- data[i, ]
  this_logis <- glm(mod_form_ilrs, data = this_dat, family = binomial())
  est <- get_pred_diff(this_logis, new_dat = pred_dat)
  return(est)

}

### CI method #1 (bootstrapping):
alpha <- 0.05
(ci_v1 <-
  c(
    est = est_v1,
    quantile(
      boot(bpd, fit_mod_boot, R = 100, pred_dat = pred_df)$t,
      c(alpha / 2, 1 - alpha / 2)
    )))

```

est.2      2.5%      97.5%  
1.0168402 0.9908345 1.0587788

```

### alternative CI method #2 (Wald approximation - re-transformed):
pred_df[, "ilr"]

```

```

      [,1]      [,2]      [,3]
[1,] 1.379475 0.05360560 2.077317
[2,] 1.137588 0.07814711 1.759779
attr("class")
[1] "rmult"

```

```

diff(pred_df[, "ilr"])

      [,1]      [,2]      [,3]
[1,] -0.2418864 0.02454151 -0.3175375
attr(,"class")
[1] "rmult"

x_0_red <- matrix(as.numeric(diff(pred_df[, "ilr"])), nrow = 1)
x_0_red

      [,1]      [,2]      [,3]
[1,] -0.2418864 0.02454151 -0.3175375

betas <- coef(bpd_occurr_ilrs)
nms_kp <- grepl("^ilr", names(betas))
betas_red <- as.matrix(betas[nms_kp])
Sigma <- stats::vcov(bpd_occurr_ilrs)
nms_kp <- grepl("^ilr", colnames(Sigma))
sigma_red <- Sigma[nms_kp, nms_kp]
sigma_red

      ilrilr(++-- ) ilrilr(+--.) ilrilr(..+-)
ilrilr(++-- ) 0.010731347 0.01369669 -0.005543191
ilrilr(+--.) 0.013696690 0.03159738 -0.008105390
ilrilr(..+-) -0.005543191 -0.00810539 0.005237994

est_red <- x_0_red %*% betas_red
se_red <- sqrt(x_0_red %*% sigma_red %*% t(x_0_red))
z_star <- qnorm(0.975)
(ci_v2 <-
  exp(c(
    est = est_red,
    lo = est_red - z_star * se_red,
    hi = est_red + z_star * se_red
  )))

      est      lo      hi
1.0168402 0.9836173 1.0511852

```

```

### alternative CI method #3 (delta method)
# (first order approximation, although still linear combin of param ests):
approx_ci <-
  deltaMethod(
    bpd_occurr_ilrs,
    "-0.2418864 * `ilrilr(++--)` + 0.02454151 * `ilrilr(+--)` + -0.3175375 * `ilrilr(..+)"
  )
(ci_v3 <-
  exp(c(
    est = approx_ci[["Estimate"]],
    lo = approx_ci[["2.5 %"]],
    hi = approx_ci[["97.5 %"]]
  )))

```

```

      est      lo      hi
1.0168402 0.9836173 1.0511852

```

```

### compare CIs
kable(rbind(ci_v1, ci_v2, ci_v3))

```

	est.2	2.5%	97.5%
ci_v1	1.01684	0.9908345	1.058779
ci_v2	1.01684	0.9836173	1.051185
ci_v3	1.01684	0.9836173	1.051185

```

do_multi_realloc <- function(mod, basis_data, timeusenames, time_changes, sbp_matrix = sbp

x0 <- basis_data

plot_dat <-
  foreach(i = 1:length(timeusenames), .combine = bind_rows) %do% {
    print(paste("i: ", i))
    foreach(j = 1:length(timeusenames), .combine = bind_rows) %do% {
      print(paste(" j: ", j))
      foreach(d = 1:length(time_changes), .combine = bind_rows) %do% {
        print(paste(" d: ", d))

        timeuse_to <- timeusenames[i]

```

```

timeuse_from <- timeusenames[j]
change_time <- time_changes[d]

proposed_change_1 <- x0[timeuse_to] + change_time
proposed_change_2 <- x0[timeuse_from] - change_time

if (timeuse_to == timeuse_from) {
  NULL # reallocation exceeds 0 or max time
} else if ((proposed_change_1 < 0) | (proposed_change_1 > 1440)) {
  NULL # reallocation exceeds 0 or max time
} else if ((proposed_change_2 < 0) | (proposed_change_2 > 1440)) {
  NULL # reallocation exceeds 0 or max time
} else {

  x1 <- x0
  x1[timeuse_to] <- x1[timeuse_to] + change_time
  x1[timeuse_from] <- x1[timeuse_from] - change_time

  pred_df <- rbind(x0, x1)
  pred_df <- add_ilrs_to_data(pred_df, comp_vars = timeusenames, sbp_matrix = sbp_matrix)

  ratio_of_odds_ratios <- get_pred_diff(mod, pred_df)

  bootstrapped_ests <- boot(bpd, fit_mod_boot, R = 1000, pred_dat = pred_df)$t
  ci_est <- quantile(as.numeric(bootstrapped_ests), c(alpha / 2, 1 - alpha / 2))

  tibble(
    to = timeuse_to,
    from = timeuse_from,
    change_time = change_time,
    ratio_of_odds_ratios = ratio_of_odds_ratios,
    ci_lo = ci_est[1],
    ci_hi = ci_est[2]
  )
}
}
}

plot_dat$to <- factor(plot_dat$to, levels = timeusenames)
plot_dat$from <- factor(plot_dat$from, levels = timeusenames)

```

```

    return(plot_dat)
}

# takes ~25 min (single core)

### Uncomment to generate bootstrapping

# tic()
# set.seed(1234)
# realloc_plot_data <-
#   do_multi_realloc(
#     bpd_occurr_ilrs,
#     predict_basis0,
#     pred_comps,
#     seq(-30, 30, by = 10)
#   )
# saveRDS(realloc_plot_data, file = "res/logistic_realloc_boot_res.rda")
# toc()

realloc_plot_data <- readRDS(file = "res/logistic_realloc_boot_res.rda")

levels(realloc_plot_data$to) <- paste0(levels(realloc_plot_data$to), "+Delta")
levels(realloc_plot_data$from) <- paste0(levels(realloc_plot_data$from), "-Delta")

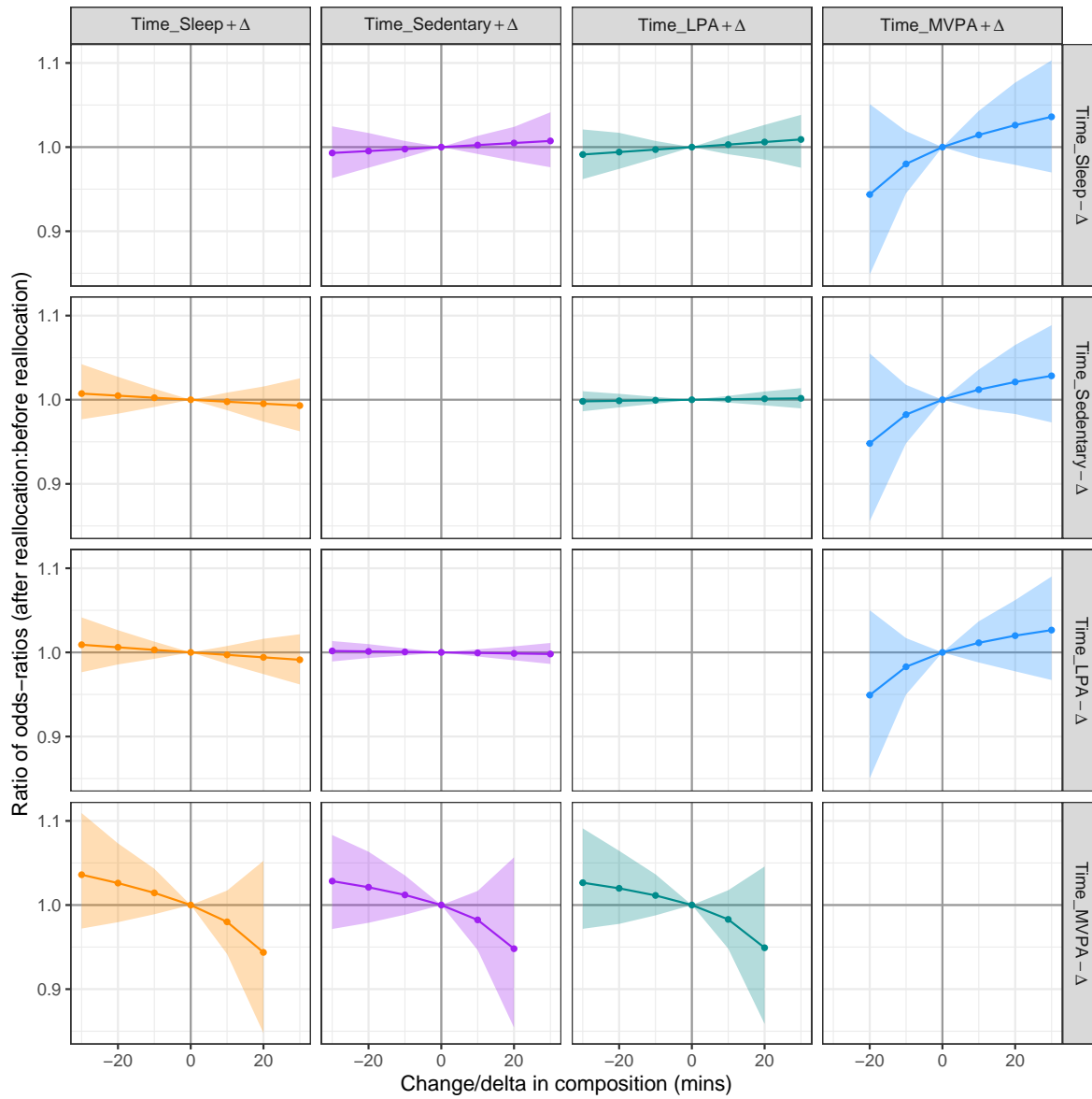
ggplot(realloc_plot_data) +
  geom_vline(xintercept = 0, col = "grey60") +
  geom_hline(yintercept = 1, col = "grey60") +
  geom_ribbon(aes(x = change_time, ymin = ci_lo, ymax = ci_hi, fill = to), alpha = 0.3) +
  geom_line(aes(x = change_time, y = ratio_of_odds_ratios, col = to)) +
  geom_point(aes(x = change_time, y = ratio_of_odds_ratios, col = to), size = 1) +
  facet_grid(from ~ to, labeller = label_parsed) +
  theme_bw() +
  scale_colour_manual(values = c("darkorange", "purple", "cyan4", "dodgerblue")) +
  scale_fill_manual(values = c("darkorange", "purple", "cyan4", "dodgerblue")) +
  labs(

```

```

x = paste0("Change/delta in composition (mins)"),
y = paste0("Ratio of odds-ratios (after reallocation:before reallocation)")
) +
theme(legend.position = "none")

```





```
ggsave(  
  filename = "fig/lbp_occur_logistic_odds.png",  
  dpi = 600, # print quality  
  width = 10,  
  height = 10  
)
```

## 5.2 Note for outcomes 1 to 2

The dataset for the remain outcomes will be limited to people who responded:

```
bpd_yes <- bpd %>% dplyr::filter(LBP_sufferer == "yes")  
nrow(bpd)
```

```
[1] 2333
```

```
nrow(bpd_yes)
```

```
[1] 1660
```

```
bpd_clean_yes <- bpd_clean %>% dplyr::filter(LBP_sufferer == "yes")
```

## 5.3 Outcome 1: LBP\_frequency\_year

### 5.3.1 Model fit

```
(this_outcome <- outcs[1])
```

```
[1] "LBP_frequency_year"
```

```
# (mod_form_null <- as.formula(paste0(this_outcome, " ~ ", rhs_formula)))  
(mod_form_ilrs <- as.formula(paste0(this_outcome, " ~ ", rhs_formula, " + ilr")))
```

```
LBP_frequency_year ~ age + sex + bmi + stress + smoking + education +  
  ses + ilr
```

```
table(bpd_yes[, this_outcome], useNA = "ifany")
```

```
LBP_frequency_year  
      0days      1-7days      8-30days      31-90days 91+_not_evday  
      0          760          451          146          203  
everyday  
      100
```

```
bpd_yes[[this_outcome]] <- fct_drop(bpd_yes[[this_outcome]])  
table(bpd_yes[, this_outcome], useNA = "ifany")
```

```
LBP_frequency_year  
      1-7days      8-30days      31-90days 91+_not_evday      everyday  
      760          451          146          203          100
```

```
## model without ilrs  
# bpd_ordinal_null <- polr(mod_form_null, data = bpd, Hess = TRUE, method = "logistic")  
# summary(bpd_ordinal_null)  
  
## model __with__ ilrs  
bpd_ordinal_ilrs <- polr(mod_form_ilrs, data = bpd_yes, Hess = TRUE, method = "logistic")
```

```
summary(bpd_ordinal_ilrs)
```

Call:

```
polr(formula = mod_form_ilrs, data = bpd_yes, Hess = TRUE, method = "logistic")
```

Coefficients:

	Value	Std. Error	t value
age2_middle	0.33918	0.10482	3.2360
age3_older	0.84204	0.16404	5.1331
sex2_male	-0.28829	0.11085	-2.6006
bmi2_normal	-0.02595	0.34969	-0.0742
bmi3_overweight	0.03744	0.35410	0.1057
stress2_stressed	0.57164	0.09929	5.7575
smoking2_nonsmoker	-0.17602	0.11788	-1.4932
education2_higher	-0.11931	0.10479	-1.1386
ses2_middle	-0.34624	0.14759	-2.3460
ses3_higher	-0.49685	0.20599	-2.4120
ilrilr(++--)	-0.30211	0.10322	-2.9268
ilrilr(+--..)	-0.56557	0.17197	-3.2888
ilrilr(..+-)	0.19891	0.07352	2.7055

Intercepts:

	Value	Std. Error	t value
1-7days 8-30days	-0.3467	0.3935	-0.8812
8-30days 31-90days	0.8660	0.3939	2.1987
31-90days 91+_not_evday	1.3945	0.3953	3.5277
91+_not_evday everyday	2.6782	0.4038	6.6327

Residual Deviance: 4392.209

AIC: 4426.209

```
Anova(bpd_ordinal_ilrs)
```

Analysis of Deviance Table (Type II tests)

Response: LBP\_frequency\_year

	LR	Chisq	Df	Pr(>Chisq)
age	27.596	2	1.018e-06	***
sex	6.822	1	0.009005	**
bmi	0.429	2	0.806925	

```

stress      33.255  1  8.084e-09 ***
smoking      2.218  1  0.136374
education    1.294  1  0.255262
ses          6.813  2  0.033158 *
ilr         11.658  3  0.008651 **
---

```

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

```

# pr <- profile(bpd_ordinal_ilrs)
# confint(pr)
# plot(pr)
# pairs(pr)

```

```
est_ci_df <- cbind(est = coef(bpd_ordinal_ilrs), confint(bpd_ordinal_ilrs)) # profiled CIs
```

Waiting for profiling to be done...

```
kable(est_ci_df, digits = 3) # these are the log-odds scale estimates (and CI)
```

	est	2.5 %	97.5 %
age2_middle	0.339	0.134	0.545
age3_older	0.842	0.520	1.164
sex2_male	-0.288	-0.506	-0.072
bmi2_normal	-0.026	-0.704	0.675
bmi3_overweight	0.037	-0.649	0.747
stress2_stressed	0.572	0.377	0.767
smoking2_nonsmoker	-0.176	-0.406	0.056
education2_higher	-0.119	-0.325	0.086
ses2_middle	-0.346	-0.635	-0.056
ses3_higher	-0.497	-0.902	-0.094
ilrilr(++-)	-0.302	-0.505	-0.100
ilrilr(+..)	-0.566	-0.904	-0.229
ilrilr(..+-)	0.199	0.055	0.343

```
kable(exp(est_ci_df), digits = 3) # these are the odds ratios (and approx CIs)
```

	est	2.5 %	97.5 %
age2_middle	1.404	1.144	1.725
age3_older	2.321	1.683	3.202
sex2_male	0.750	0.603	0.931
bmi2_normal	0.974	0.495	1.965
bmi3_overweight	1.038	0.523	2.111
stress2_stressed	1.771	1.458	2.152
smoking2_nonsmoker	0.839	0.666	1.057
education2_higher	0.888	0.723	1.090
ses2_middle	0.707	0.530	0.945
ses3_higher	0.608	0.406	0.910
ilr1r(++-)	0.739	0.603	0.905
ilr1r(+..)	0.568	0.405	0.795
ilr1r(..+)	1.220	1.057	1.410

Ordinal logistic regression has fit the model:

$$\begin{aligned}
\text{logit}(\hat{P}(Y \leq 1-7\text{days})) &= \hat{\beta}_{0,1-7\text{days}|8-30\text{days}} - \hat{\beta}_1 \times (\text{age}) - \dots - \hat{\beta}_p \times \text{ilr}(\dots) \\
\text{logit}(\hat{P}(Y \leq 8-30\text{days})) &= \hat{\beta}_{0,8-30\text{days}|31-90\text{days}} - \hat{\beta}_1 \times (\text{age}) - \dots - \hat{\beta}_p \times \text{ilr}(\dots) \\
\text{logit}(\hat{P}(Y \leq 31-90\text{days})) &= \hat{\beta}_{0,31-90\text{days}|91+_{\text{not\_evday}}} - \hat{\beta}_1 \times (\text{age}) - \dots - \hat{\beta}_p \times \text{ilr}(\dots) \\
\text{logit}(\hat{P}(Y \leq 91+_{\text{not\_evday}})) &= \hat{\beta}_{0,91+_{\text{not\_evday}}|\text{everyday}} - \hat{\beta}_1 (\text{age} == 2\_middle) - \dots - \hat{\beta}_p \times \text{ilr}(\dots)
\end{aligned}$$

### 5.3.2 Model diagnostics

```
# deviance test
g2 <- deviance(bpd_ordinal_ilrs)
df <- df.residual(bpd_ordinal_ilrs)
1 - pchisq(g2, df)
```

[1] 0

```
with(bpd_yes,
  table(
    LBP_frequency_year,
    as.numeric(LBP_frequency_year),
    useNA = "ifany"
```

```
)
)
```

```
LBP_frequency_year  1    2    3    4    5
1-7days            760    0    0    0    0
8-30days           0 451    0    0    0
31-90days          0    0 146    0    0
91+_not_evday       0    0    0 203    0
everyday            0    0    0    0 100
```

```
## checking parallel slopes assumptions can be done by fitting successive logistic regressions
## while creating a binary outcome using different thresholds of the ordinal outcome
### (note the rhs/linear predictor is negative so coeffs should be approx same
### as main model except negative)
```

```
# e.g. this is a single logistic regression
```

```
coef(glm(
  I(as.numeric(LBP_frequency_year) <= 1) ~
    age + sex + bmi + stress + smoking + education + ses + ilr,
  family = "binomial",
  data = bpd_yes
))
```

(Intercept)	age2_middle	age3_older	sex2_male
-0.60603839	-0.25411567	-0.65013781	0.24867275
bmi2_normal	bmi3_overweight	stress2_stressed	smoking2_nonsmoker
0.17851384	0.11126358	-0.50892555	0.22359339
education2_higher	ses2_middle	ses3_higher	ilrilr(++--)
0.08669237	0.33663193	0.46257283	0.28067682
ilrilr(+--)	ilrilr(..+-)		
0.44988574	-0.16448339		

```
# this is running multiple logistic regressions
```

```
## we want to see the coefficients to be roughly the same (intercepts and negative coeffs -
```

```
### note that the below shows there may be reason to include an age variable that has
### non-constant coefficient for each level of the outcome (or subgroup analyses for each
### we can see this because the age coeffs increase/decrease monotonically
```

```
foreach(i = 1:(length(levels(bpd_yes$LBP_frequency_year)) - 1), .combine = cbind) %do% {
```

```

log_coefs <-
  coef(glm(
    I(as.numeric(LBP_frequency_year) <= i) ~
      age + sex + bmi + stress + smoking + education + ses + ilr,
    family = "binomial",
    data = bpd_yes
  ))
log_coefs <- as.data.frame(log_coefs)
colnames(log_coefs) <- paste0("logit(P(Y<=", i, "))")
log_coefs
} %>%
  kable(., digits = 2)

```

	logit(P(Y<=1))	logit(P(Y<=2))	logit(P(Y<=3))	logit(P(Y<=4))
(Intercept)	-0.61	1.15	1.99	3.51
age2_middle	-0.25	-0.43	-0.50	-0.80
age3_older	-0.65	-0.99	-1.09	-1.69
sex2_male	0.25	0.33	0.29	0.31
bmi2_normal	0.18	0.00	-0.39	0.01
bmi3_overweight	0.11	0.01	-0.50	-0.25
stress2_stressed	-0.51	-0.65	-0.59	-0.94
smoking2_nonsmoker	0.22	0.12	0.10	-0.19
education2_higher	0.09	0.12	0.14	0.49
ses2_middle	0.34	0.32	0.35	0.31
ses3_higher	0.46	0.51	0.74	0.52
ilrilr(++-)	0.28	0.29	0.37	0.38
ilrilr(+..)	0.45	0.69	0.71	0.54
ilrilr(..+-)	-0.16	-0.25	-0.25	-0.25

### 5.3.3 Model predictions

```

# create dataset for predictions
newdata <-
  bpd_yes %>%
  dplyr::select(all_of(pred_covs), ilr) %>%
  distinct(pick(all_of(pred_covs)), .keep_all = TRUE) %>%
  arrange(pick(all_of(pred_covs)))

mean_ilr <- mean(bpd_yes$ilr)

```



```

dev_null <- foreach(i = 1:nrow(newdata)) %do% {
  newdata$ilr[i, ] <- mean_ilr
}

# make preds and then put in long format for ggplot
predictions_probs <-
  cbind(
    predict(bpd_ordinal_ilrs, newdata, type = "probs"),
    newdata
  ) %>%
  dplyr::select(-ilr) %>%
  pivot_longer(
    cols = -all_of(pred_covs),
    names_to = "outcome",
    values_to = "P(outc)"
  )

predictions_probs

# A tibble: 1,030 x 9
  age      sex      bmi      stress smoking education ses      outcome `P(outc)`
  <fct>    <chr>    <fct>    <chr>  <chr>    <chr>    <chr> <chr>      <dbl>
1 1_younger 1_female 1_underw~ 1_nor~ 1_smok~ 2_higher 1_lo~ 1-7days  0.451
2 1_younger 1_female 1_underw~ 1_nor~ 1_smok~ 2_higher 1_lo~ 8-30da~  0.283
3 1_younger 1_female 1_underw~ 1_nor~ 1_smok~ 2_higher 1_lo~ 31-90d~  0.0899
4 1_younger 1_female 1_underw~ 1_nor~ 1_smok~ 2_higher 1_lo~ 91+_no~  0.120
5 1_younger 1_female 1_underw~ 1_nor~ 1_smok~ 2_higher 1_lo~ everyd~  0.0558
6 1_younger 1_female 1_underw~ 1_nor~ 1_smok~ 2_higher 2_mi~ 1-7days  0.537
7 1_younger 1_female 1_underw~ 1_nor~ 1_smok~ 2_higher 2_mi~ 8-30da~  0.259
8 1_younger 1_female 1_underw~ 1_nor~ 1_smok~ 2_higher 2_mi~ 31-90d~  0.0727
9 1_younger 1_female 1_underw~ 1_nor~ 1_smok~ 2_higher 2_mi~ 91+_no~  0.0910
10 1_younger 1_female 1_underw~ 1_nor~ 1_smok~ 2_higher 2_mi~ everyd~  0.0401
# i 1,020 more rows

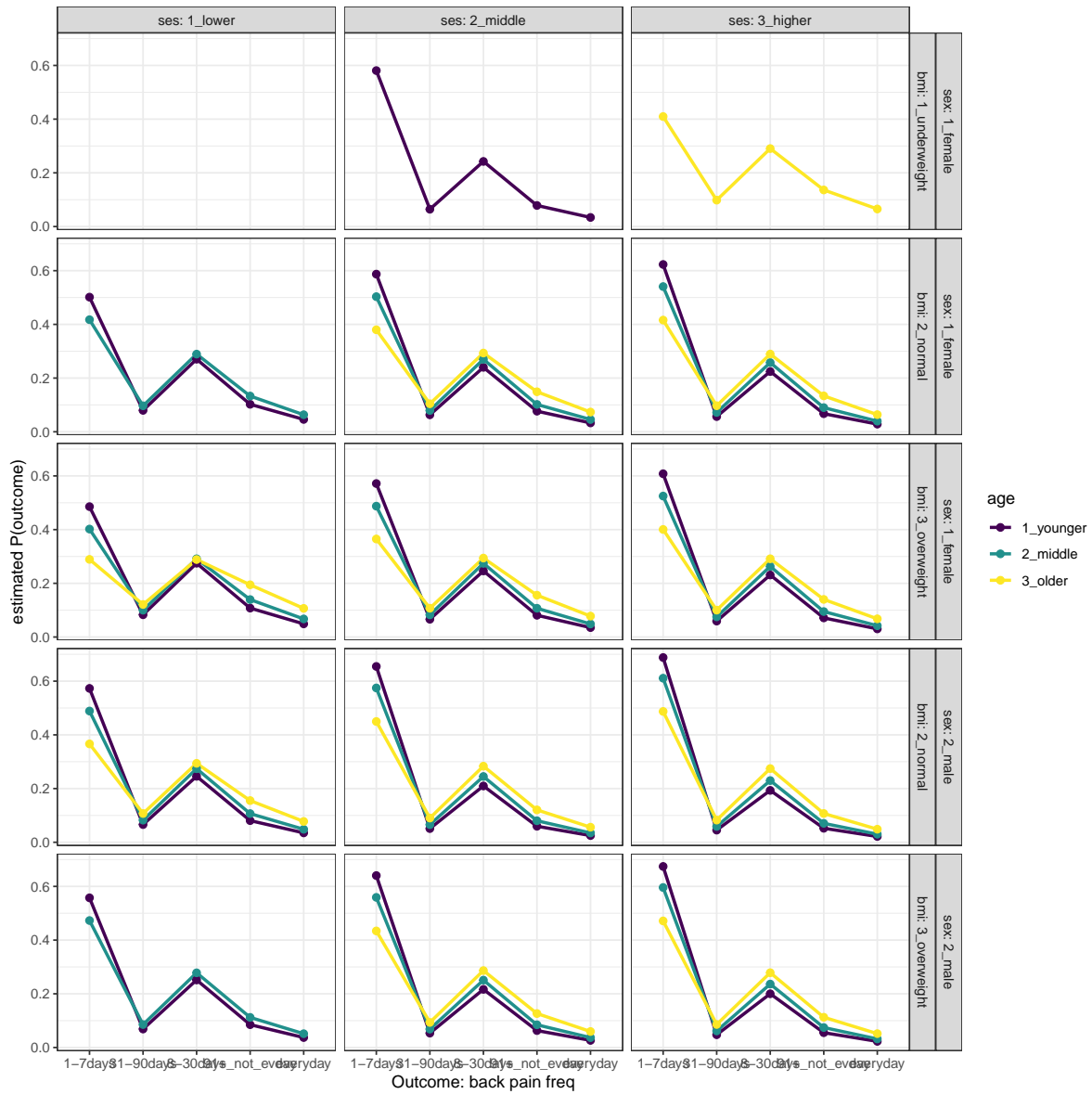
## model predictions for specific values
predictions_probs %>%
  dplyr::filter(
    # sex == "1_female",
    stress == "1_normal",
    smoking == "2_nonsmoker",
    education == "2_higher",

```

```

    # ses == "2_middle"
  ) %>%
  ggplot(., aes(outcome, `P(outc)`)) +
  geom_line(aes(colour = age, group = age), linewidth = 1) +
  geom_point(aes(colour = age), size = 2) +
  facet_grid(sex * bmi ~ ses, labeller = label_both) +
  labs(x = "Outcome: back pain freq", y = "estimated P(outcome)") +
  theme_bw() +
  scale_colour_viridis_d()

```



```
# create a RHS of regression equation dataset for time-reallocation
predict_basis <-
  bpd_yes %>%
  dplyr::select(all_of(pred_covs), all_of(pred_comps)) %>%
  dplyr::filter(
    age == "2_middle",
    sex == "1_female",

```

```

    stress == "1_normal",
    smoking == "2_nonsmoker",
    education == "2_higher",
    ses == "2_middle",
    bmi == "2_normal"
  )

### continuous situation
# predict_basis$age <- mean(predict_basis$age)

(predict_basis <-
  predict_basis %>%
    distinct(across(all_of(pred_covs)), .keep_all = TRUE) %>%
    as.data.frame())

```

	age	sex	bmi	stress	smoking	education	ses	Time_Sleep
1	2_middle	1_female	2_normal	1_normal	2_nonsmoker	2_higher	2_middle	546.4286
	Time_Sedentary	Time_LPA	Time_MVPA					
1	418.2857	435.1429	40					

```

# compositional mean: geometric mean to closure
# (comp_mean <- mean(acom(bpd_yes[, pred_comps])))
(comp_mean <- calc_comp_mean(bpd_yes[, pred_comps], clo_val = 1440))

```

Time_Sleep	Time_Sedentary	Time_LPA	Time_MVPA
472.8407	438.4062	502.1666	26.5865

```

predict_basis0 <- predict_basis
predict_basis0[, pred_comps] <- comp_mean

predict_basis0

```

	age	sex	bmi	stress	smoking	education	ses	Time_Sleep
1	2_middle	1_female	2_normal	1_normal	2_nonsmoker	2_higher	2_middle	472.8407
	Time_Sedentary	Time_LPA	Time_MVPA					
1	438.4062	502.1666	26.5865					

```
# +15 minutes to Time_MVPA and -15 minutes from Time_Sedentary
comp_mean_changed <- comp_mean
comp_mean_changed["Time_MVPA"] <- comp_mean_changed["Time_MVPA"] + 15
comp_mean_changed["Time_Sedentary"] <- comp_mean_changed["Time_Sedentary"] - 15
# check
comp_mean_changed - comp_mean
```

Time_Sleep	Time_Sedentary	Time_LPA	Time_MVPA
0	-15	0	15

```
predict_basis1 <- predict_basis
predict_basis1[, pred_comps] <- comp_mean_changed

pred_df <- rbind(predict_basis0, predict_basis1)
pred_df <- add_ilrs_to_data(pred_df, comp_vars = pred_comps, sbp_matrix = sbp1)
pred_df <- pred_df[, !(colnames(pred_df) %in% pred_comps)] # get rid of compositions
pred_df
```

	age	sex	bmi	stress	smoking	education	ses	ilr.1
1	2_middle	1_female	2_normal	1_normal	2_nonsmoker	2_higher	2_middle	1.37128443
2	2_middle	1_female	2_normal	1_normal	2_nonsmoker	2_higher	2_middle	1.13019151
	ilr.2	ilr.3						
1	0.05346627	2.07785318						
2	0.07808340	1.76151343						

```
# model.matrix(formula(bpd_ordinal_ilrs), data = cbind(LBP_frequency_year = 0, pred_df))

df <- bpd_yes[, attr(formula(bpd_ordinal_ilrs), "term.labels")]
# this is a list of levels for each factor in the original df (after applying factor function)
xlevs <- lapply(df[,sapply(df, is.character), drop = F], function(j) {
  levels(factor(j))
})

# calling "xlev = " builds out a model.matrix with identical levels as the original df
mm_new <- model.matrix( ~ ., data = pred_df, xlev = xlevs)
colnames(mm_new)
```

```
[1] "(Intercept)"          "age2_middle"          "age3_older"
```

```

[4] "sex2_male"          "bmi2_normal"        "bmi3_overweight"
[7] "stress2_stressed"   "smoking2_nonsmoker" "education2_higher"
[10] "ses2_middle"        "ses3_higher"        "ilr1"
[13] "ilr2"              "ilr3"

```

```
mm_new <- mm_new[, -1] # remove intercept
```

```

colnames(mm_new)[grepl("^ilr", colnames(mm_new))] <- paste0("ilr", create_ilr_names(sbp1))
# colnames(mm_new)
# don't need intercept # c("(Intercept)" = 1, coef(bpd_ordinal_ilrs))
betas <- as.matrix(coef(bpd_ordinal_ilrs)) # should be col matrix
# rownames(betas)

```

```

if (!all(colnames(mm_new) == rownames(betas))) {
  stop("design and parameter est matrices non-conform")
}

```

```

# note as linear predictor is taken from the K intercepts the ratio of odds ratios is flip
# i.e. after:before of odds is calculated as exp(before_log_odds / after_log_odds)
preds <- mm_new %*% betas
exp(preds[1] - preds[2])

```

```
[1] 1.004017
```

```

# check manual calcs agree with model
mm_old <- model.matrix( ~ ., data = df, xlev = xlevs)
mm_old <- mm_old[, -1] # remove intercept
# colnames(mm_old)

```

```

# model and manual calcs agree?
# note that bpd_ordinal_ilrs$lp are the eta/linear predictor that is taken
# away from the xi_k intercept
all(abs(as.numeric(mm_old %*% betas) - bpd_ordinal_ilrs$lp) < 1e-9)

```

```
[1] TRUE
```

```
# bpd_ordinal_ilrs$lp # linear predictor
```

```

get_pred_diff <- function(mod, new_dat) {
  betas_ <- as.matrix(coef(mod))
  if (!all(colnames(new_dat) == rownames(betas_))) {
    print(paste(paste(colnames(new_dat), collapse = "|"), "vs", paste(rownames(betas_), collapse = "|"), sep = " "))
    stop("design and parameter est matrices non-conform")
  }
  log_odds_pred <- as.numeric(new_dat %*% betas_)
  # note reversal of order (see above)
  odds_ratio_ratio <- exp(log_odds_pred[1] - log_odds_pred[2])
  return(odds_ratio_ratio)
}
(est_v1 <- get_pred_diff(bpd_ordinal_ilrs, mm_new))

```

[1] 1.004017

```

fit_mod_boot <- function(data, i, pred_dat) {

  this_dat <- data[i, ]
  this_ordinal <- polr(mod_form_ilrs, data = this_dat, Hess = TRUE, method = "logistic")

  df <- this_dat[, attr(formula(this_ordinal), "term.labels")]
  # this is a list of levels for each factor in the original df (after applying factor function)
  xlevs <- lapply(df[,sapply(df, is.character), drop = F], function(j) {
    levels(factor(j))
  })

  # calling "xlev = " builds out a model.matrix with identical levels as the original df
  mm_new <- model.matrix( ~ ., data = pred_dat, xlev = xlevs)
  mm_new <- mm_new[, -1] # remove intercept
  # make sure ilr colnames are legit/match coeffs
  colnames(mm_new)[grepl("^ilr", colnames(mm_new))] <- paste0("ilr", create_ilr_names(sbp1))
  colnames(mm_new)

  est <- get_pred_diff(this_ordinal, new_dat = mm_new)
  return(est)

}

### CI method #1 (bootstrapping):
alpha <- 0.05

```

```
(ci_v1 <-
  c(
    est = est_v1,
    quantile(
      boot(bpd_yes, fit_mod_boot, R = 100, pred_dat = pred_df)$t,
      c(alpha / 2, 1 - alpha / 2)
    )))
```

```
      est      2.5%      97.5%
1.0040167 0.9722781 1.0450194
```

```
### alternative CI method #2 (Wald approximation - re-transformed):
pred_df[, "ilr"]
```

```
      [,1]      [,2]      [,3]
[1,] 1.371284 0.05346627 2.077853
[2,] 1.130192 0.07808340 1.761513
attr("class")
[1] "rmult"
```

```
diff(pred_df[, "ilr"])
```

```
      [,1]      [,2]      [,3]
[1,] -0.2410929 0.02461713 -0.3163398
attr("class")
[1] "rmult"
```

```
# x_0_red <- matrix(- as.numeric(diff(pred_df[, "ilr"])), nrow = 1)
x_0_red <- matrix(as.numeric(pred_df[1, "ilr"] - pred_df[2, "ilr"]), nrow = 1)
x_0_red
```

```
      [,1]      [,2]      [,3]
[1,] 0.2410929 -0.02461713 0.3163398
```

```
betas <- coef(bpd_ordinal_ilrs)
nms_kp <- grepl("^ilr", names(betas))
```



```

betas_red <- as.matrix(betas[nms_kp])
Sigma <- stats::vcov(bpd_ordinal_ilrs)
nms_kp <- grepl("^ilr", colnames(Sigma))
sigma_red <- Sigma[nms_kp, nms_kp]
sigma_red

```

```

           ilrilr(++--) ilrilr(+--.) ilrilr(..+-)
ilrilr(++--)  0.01065435  0.01320158 -0.00568300
ilrilr(+--.)  0.01320158  0.02957295 -0.00789973
ilrilr(..+-) -0.00568300 -0.00789973  0.00540513

```

```

est_red <- x_0_red %*% betas_red
se_red <- sqrt(x_0_red %*% sigma_red %*% t(x_0_red))
z_star <- qnorm(0.975)
(ci_v2 <-
  exp(c(
    est = est_red,
    lo = est_red - z_star * se_red,
    hi = est_red + z_star * se_red
  )))

```

```

           est           lo           hi
1.0040167 0.9717601 1.0373441

```

```

### alternative CI method #3 (delta method)
# (first order approximation, although still linear combin of param ests):
as.numeric(x_0_red)

```

```

[1] 0.24109292 -0.02461713 0.31633975

```

```

(g_form <- paste(
  paste(
    as.numeric(x_0_red),
    "*",
    c("`ilrilr(++--)`", "`ilrilr(+--.)`", "`ilrilr(..+-)`")
  ),
  collapse = " + "
))

```

```
[1] "0.241092921978823 * `ilrilr(++--)` + -0.0246171278023131 * `ilrilr(+--)` + 0.316339752"
```

```
approx_ci <-deltaMethod(bpd_ordinal_ilrs, g_form)
(ci_v3 <-
  exp(c(
    est = approx_ci[["Estimate"]],
    lo = approx_ci[["2.5 %"]],
    hi = approx_ci[["97.5 %"]]
  )))
```

```
      est      lo      hi
1.0040167 0.9717601 1.0373441
```

```
### compare CIs
kable(rbind(ci_v1, ci_v2, ci_v3))
```

	est	2.5%	97.5%
ci_v1	1.004017	0.9722781	1.045019
ci_v2	1.004017	0.9717601	1.037344
ci_v3	1.004017	0.9717601	1.037344

```
do_multi_realloc <- function(mod, basis_data, timeusenames, time_changes, sbp_matrix = sbp
```

```
x0 <- basis_data
```

```
plot_dat <-
  foreach(i = 1:length(timeusenames), .combine = bind_rows) %do% {
    print(paste("i: ", i))
    foreach(j = 1:length(timeusenames), .combine = bind_rows) %do% {
      print(paste(" j: ", j))
      foreach(d = 1:length(time_changes), .combine = bind_rows) %do% { # %dopar%
        print(paste(" d: ", d))

        timeuse_to <- timeusenames[i]
        timeuse_from <- timeusenames[j]
        change_time <- time_changes[d]

        proposed_change_1 <- x0[timeuse_to] + change_time
```

```

proposed_change_2 <- x0[timeuse_from] - change_time

if (timeuse_to == timeuse_from) {
  NULL # reallocation exceeds 0 or max time
} else if ((proposed_change_1 < 0) | (proposed_change_1 > 1440)) {
  NULL # reallocation exceeds 0 or max time
} else if ((proposed_change_2 < 0) | (proposed_change_2 > 1440)) {
  NULL # reallocation exceeds 0 or max time
} else {

  x1 <- x0
  x1[timeuse_to] <- x1[timeuse_to] + change_time
  x1[timeuse_from] <- x1[timeuse_from] - change_time

  pred_df <- rbind(x0, x1)
  pred_df <- add_ilrs_to_data(pred_df, comp_vars = timeusenames, sbp_matrix = sbp_matrix)

  ### alternative CI method #3 (delta method)
  # x_0_red <- -as.numeric(diff(pred_df[, "ilr"]))
  x_0_red <- as.numeric(pred_df[1, "ilr"] - pred_df[2, "ilr"])
  # (first order approximation, although still linear combin of param ests):
  (g_form <- paste(
    paste(
      x_0_red,
      "*",
      c("`ilrilr(++--)`", "`ilrilr(+-.)`", "`ilrilr(..+)`")
    ),
    collapse = " + "
  ))
  approx_ci <- deltaMethod(bpd_ordinal_ilrs, g_form)
  this_ci <-
    exp(c(
      est = approx_ci[["Estimate"]],
      lo = approx_ci[["2.5 %"]],
      hi = approx_ci[["97.5 %"]]
    ))

  ### bootstrapping takes too long
  # pred_df <- pred_df[, !(colnames(pred_df) %in% timeusenames)] # get rid of co
  #

```

```

# df <- bpd_yes[, attr(formula(bpd_ordinal_ilrs), "term.labels")]
# # this is a list of levels for each factor in the original df (after applying
# xlevs <- lapply(df[,sapply(df, is.character), drop = F], function(j) {
#   levels(factor(j))
# })
#
# # calling "xlev = " builds out a model.matrix with identical levels as the o
# mm_new <- model.matrix( ~ ., data = pred_df, xlev = xlevs)
# mm_new <- mm_new[, -1] # remove intercept
# # make sure ilr colnames are legit/match coeffs
# colnames(mm_new)[grepl("^ilr", colnames(mm_new))] <- paste0("ilr", create_ilr)
#
# ratio_of_odds_ratios <- get_pred_diff(mod, new_dat = mm_new)
# bootstrapped_ests <- boot(bpd_yes, fit_mod_boot, R = 10, pred_dat = pred_df)
# ci_est <- quantile(as.numeric(bootstrapped_ests), c(alpha / 2, 1 - alpha / 2))

tibble(
  to = timeuse_to,
  from = timeuse_from,
  change_time = change_time,
  ratio_of_odds_ratios = this_ci["est"],
  ci_lo = this_ci["lo"],
  ci_hi = this_ci["hi"]
)
}
}
}

plot_dat$to <- factor(plot_dat$to, levels = timeusenames)
plot_dat$from <- factor(plot_dat$from, levels = timeusenames)

return(plot_dat)

}

# takes ~ 3h (single core) for bootstrapping
# takes ~ 4sec (single core) for delta/wald method

```

```

### Uncomment to generate bootstrapping

# tic()
# set.seed(1234)

# # library("doParallel")
# # no_cores <- detectCores() - 1 # Calculate the number of cores (leave one free)
# # cl <- makeCluster(no_cores) # Create clusters
# # registerDoParallel(cl) # and register

# realloc_plot_data <-
#   do_multi_realloc(
#     bpd_ordinal_ilrs,
#     predict_basis0,
#     pred_comps,
#     seq(-30, 30, by = 10)
#   )

# # # close para comp
# # stopCluster(cl)

# saveRDS(realloc_plot_data, file = "res/ordinal_realloc_wald_res.rda")
# toc()

# saveRDS(realloc_plot_data, file = "res/ordinal_realloc_boot_res.rda")
# realloc_plot_data <- readRDS(file = "res/ordinal_realloc_boot_res.rda")

realloc_plot_data <- readRDS(file = "res/ordinal_realloc_wald_res.rda")

levels(realloc_plot_data$to) <- paste0(levels(realloc_plot_data$to), "+Delta")
levels(realloc_plot_data$from) <- paste0(levels(realloc_plot_data$from), "-Delta")

ggplot(realloc_plot_data) +
  geom_vline(xintercept = 0, col = "grey60") +
  geom_hline(yintercept = 1, col = "grey60") +
  geom_ribbon(aes(x = change_time, ymin = ci_lo, ymax = ci_hi, fill = to), alpha = 0.3) +
  geom_line(aes(x = change_time, y = ratio_of_odds_ratios, col = to)) +
  geom_point(aes(x = change_time, y = ratio_of_odds_ratios, col = to), size = 1) +
  facet_grid(from ~ to, labeller = label_parsed) +

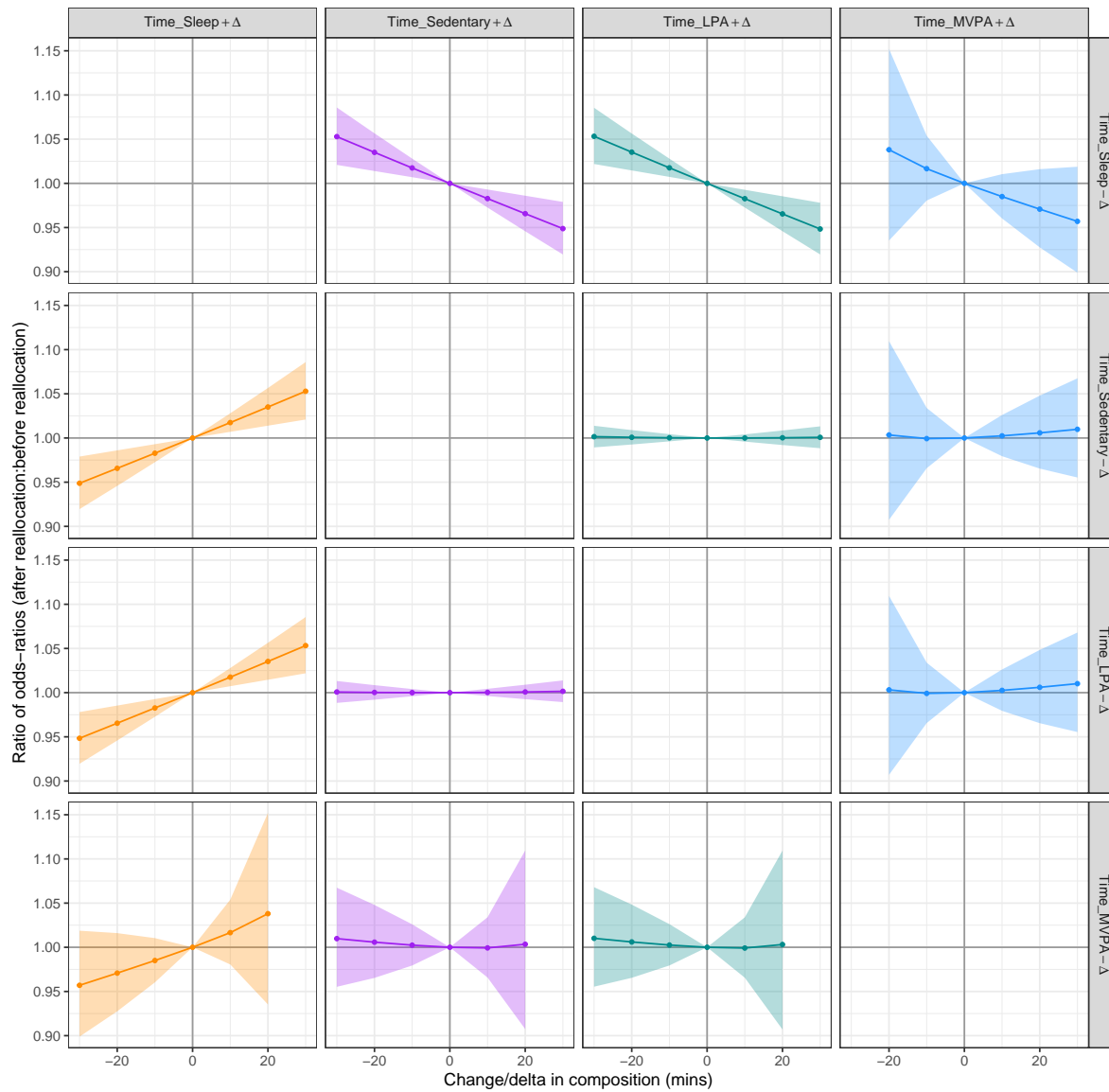
```

```

theme_bw() +
scale_colour_manual(values = c("darkorange", "purple", "cyan4", "dodgerblue")) +
scale_fill_manual(values = c("darkorange", "purple", "cyan4", "dodgerblue")) +
labs(
  x = paste0("Change/delta in composition (mins)"),
  y = paste0("Ratio of odds-ratios (after reallocation:before reallocation)"),
  subtitle = "Note that odds ratios relate to the probability of having _decreased_ freq
) +
theme(legend.position = "none")

```

Note that odds ratios relate to the probability of having `_decreased_` frequency (per year) of pain



```
ggsave(  
  filename = "fig/lbp_freq_ordinal_odds_v1.png",  
  dpi = 600, # print quality  
  width = 10,  
  height = 10  
)
```

```
time_lvls <- gsub("Time_", "", pred_comps)
```

```
rep_char <- function(n, char = " ") paste(rep(char, n), collapse = "")
rep_char(3)
```

```
[1] "   "
```

```
rep_char(0)
```

```
[1] ""
```

```
rep_char <- Vectorize(rep_char, vectorize.args = "n")
rep_char(0:7)
```

```
[1] ""      " "      "  "      "   "      "    "      "     "      "      "
[8] "       "        "         "          "           "            "             "
```

```
pd2 <-
  realloc_plot_data %>%
  mutate(
    to = gsub("Time_", "", to),
    from = gsub("Time_", "", from),
    to = gsub("+Delta", "", to, fixed = TRUE),
    from = gsub("-Delta", "", from, fixed = TRUE),
    to_len = nchar(to),
    to_max = max(to_len),
    from_len = nchar(from),
    from_max = max(from_len),
    to_pad = rep_char(pmax(0, from_max - to_len)),
    from_pad = rep_char(pmax(0, to_max - from_len)),
    to = factor(to, levels = time_lvls),
    from = factor(from, levels = time_lvls),
    to_num = as.numeric(to),
```



```

    from_num = as.numeric(from)
  ) %>%
  dplyr::filter(to_num > from_num) %>%
  mutate(
    ratio_of_odds_ratios = 1 / ratio_of_odds_ratios,
    tmp = 1 / ci_lo,
    ci_lo = 1 / ci_hi,
    ci_hi = tmp,
    # from_to = paste0(" ", "+", from, rep_char(10), from_pad, "\u2194", to_pad, rep_c
    from_to = paste0("+", from, rep_char(13), from_pad, "", to_pad, rep_char(13), "+", to)
  ) %>%
  arrange(from, to)

unique(pd2$from_to)

```

```

[1] "+Sleep                +Sedentary"
[2] "+Sleep                +LPA"
[3] "+Sleep                +MVPA"
[4] "+Sedentary            +LPA"
[5] "+Sedentary            +MVPA"
[6] "+LPA                  +MVPA"

```

```

pd2$from_to <- factor(pd2$from_to, levels = unique(pd2$from_to))

this_breaks <- seq(-30, 30, 10)
this_labs <- sprintf("+%2.0f", abs(seq(-30, 30, 10)))
this_labs[this_labs == "+ 0"] <- ""
this_labs

```

```

[1] "+30" "+20" "+10" ""      "+10" "+20" "+30"

```

```

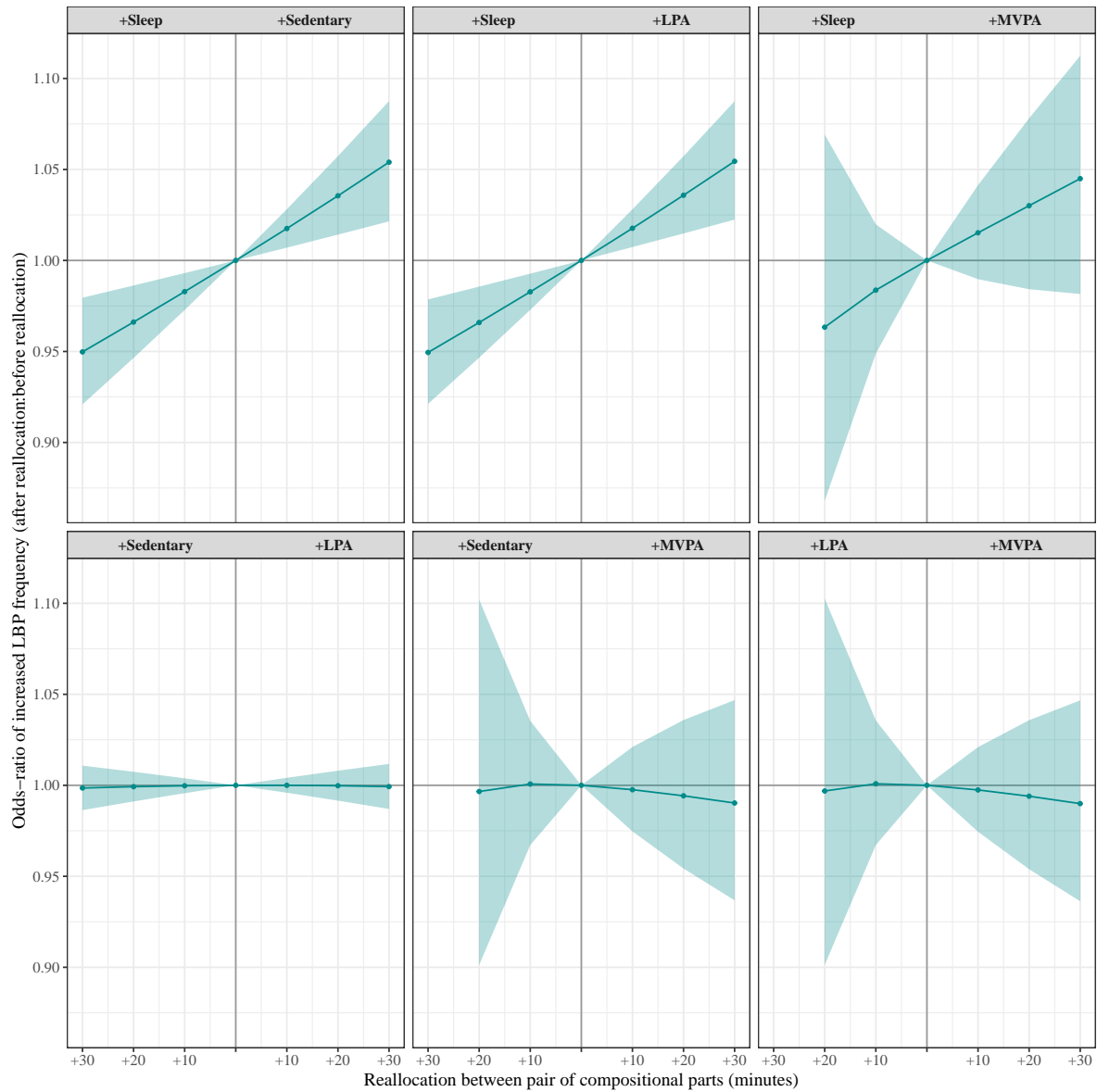
ggplot(pd2) +
  geom_vline(xintercept = 0, col = "grey60") +
  geom_hline(yintercept = 1, col = "grey60") +
  geom_ribbon(aes(x = change_time, ymin = ci_lo, ymax = ci_hi, fill = to), alpha = 0.3, co
  geom_line(aes(x = change_time, y = ratio_of_odds_ratios, col = to), col = "cyan4") +
  geom_point(aes(x = change_time, y = ratio_of_odds_ratios, col = to), size = 1, col = "c
  facet_wrap(~ from_to, labeller = label_bquote(.(from_to))) +
  theme_bw() +

```

```

scale_x_continuous(breaks = this_breaks, labels = this_labs) +
labs(
  x = paste0("Reallocation between pair of compositional parts (minutes)"),
  y = paste0("Odds-ratio of increased LBP frequency (after reallocation:before reallocation)",
  # subtitle = "Note that odds ratios relate to the probability of having _increased_ frequency")
) +
theme(
  legend.position = "none",
  text = element_text(family = "serif"),
  strip.text = element_text(size = 10, face = "bold"),
  axis.text = element_text(size = 10),
  axis.title = element_text(size = 12)
)

```



```
ggsave(filename = "fig/lbp_freq_ordinal_odds_v2.png", width = 14, height = 9, dpi = 600)
# ggsave(filename = "fig/lbp_freq_ordinal_odds.pdf", width = 10, height = 8)
```

```
# ---- outcome1_pred_not_use ----
```

```
# logitP(Yk x) =  $\eta_k$  -
# zeta_{1-7days|8-30days} = -0.2910
# eta = 0.3184 + -0.1786 + -0.1110 + -0.3463 + -0.3017
coef(bpd_ordinal_ilrs)
```

age2_middle	age3_older	sex2_male	bmi2_normal
0.33918326	0.84203588	-0.28828961	-0.02594625
bmi3_overweight	stress2_stressed	smoking2_nonsmoker	education2_higher
0.03744251	0.57164263	-0.17602164	-0.11931436
ses2_middle	ses3_higher	ilrilr(++-)	ilrilr(+-.)
-0.34624303	-0.49685106	-0.30210820	-0.56557283
ilrilr(..+)			
0.19890657			

```
# summary(bpd_ordinal_ilrs)
bpd_ordinal_ilrs$zeta
```

1-7days 8-30days	8-30days 31-90days	31-90days 91+_not_evday
-0.3467297	0.8659799	1.3944591
91+_not_evday everyday		
2.6782000		

```
# bpd_ordinal_ilrs$lp
# p_0 <- predict(bpd_ordinal_ilrs, pred_df, type = "prob")
# (lodr <- log(p_0 / (1-p_0)))
# # ratio of odds ratios
# exp(apply(lodr, 2, diff))
# # predicted class argmin_k{abs(zeta_k - eta)}?
# predict(bpd_ordinal_ilrs, type = "class")[1:3]
# p_m <- matrix(rep(bpd_ordinal_ilrs$lp, 5), ncol = 5)
# co_m <- matrix(rep(c(bpd_ordinal_ilrs$zeta, 0), nrow(p_m)), ncol = 5, byrow = TRUE)
# apply(abs(p_m - co_m), 1, which.min)[1:3]
#
# table(
#   predict(bpd_ordinal_ilrs, type = "class"),
#   apply(abs(p_m - co_m), 1, which.min) # + 1) %% 5
# )
```

## 5.4 Outcome 2: LBP\_intensity\_year

### 5.4.1 Model fit

```
(this_outcome <- outcs[2])
```

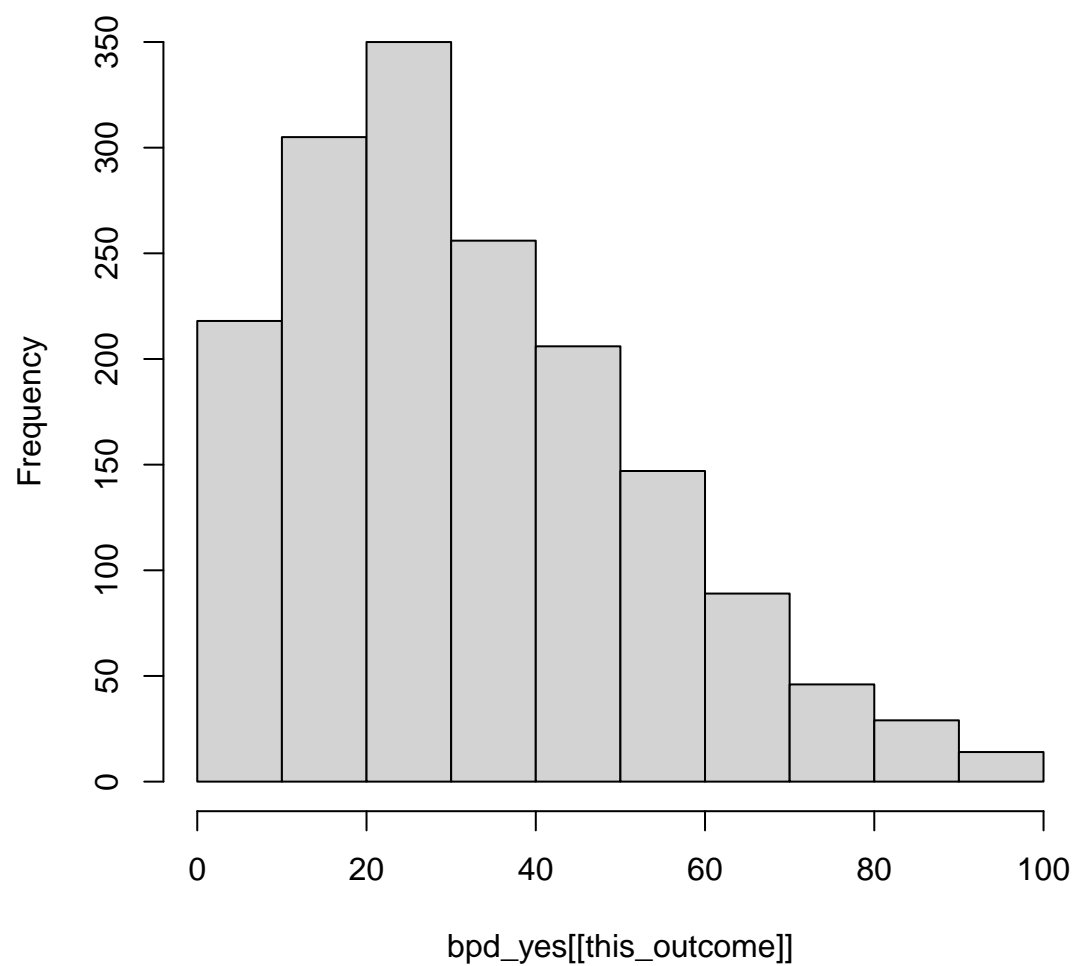
```
[1] "LBP_intensity_year"
```

```
# (mod_form_null <- as.formula(paste0(this_outcome, " ~ ", rhs_formula)))  
(mod_form_ilrs <- as.formula(paste0(this_outcome, " ~ ", rhs_formula, " + ilr")))
```

```
LBP_intensity_year ~ age + sex + bmi + stress + smoking + education +  
  ses + ilr
```

```
hist(bpd_yes[[this_outcome]])
```

**Histogram of bpd\_yes[[this\_outcome]]**



```
lbp_intensity_lm <- lm(mod_form_ilrs, data = bpd_yes)
summary(lbp_intensity_lm)
```

Call:

```
lm(formula = mod_form_ilrs, data = bpd_yes)
```

Residuals:

Min	1Q	Median	3Q	Max
-----	----	--------	----	-----

-42.94 -14.78 -3.03 12.16 69.47

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	40.3475	4.3139	9.353	< 2e-16 ***
age2_middle	0.5727	1.1059	0.518	0.604605
age3_older	7.2422	1.7224	4.205	2.75e-05 ***
sex2_male	-3.5483	1.1587	-3.062	0.002231 **
bmi2_normal	-0.5308	3.8464	-0.138	0.890251
bmi3_overweight	2.4503	3.8931	0.629	0.529177
stress2_stressed	4.4659	1.0541	4.237	2.39e-05 ***
smoking2_nonsmoker	1.3548	1.2786	1.060	0.289471
education2_higher	-3.2982	1.1148	-2.959	0.003134 **
ses2_middle	-4.2698	1.6008	-2.667	0.007722 **
ses3_higher	-7.4351	2.1993	-3.381	0.000740 ***
ilr1lr(++--)	-2.0302	1.0869	-1.868	0.061949 .
ilr1lr(+--.)	-6.9931	1.7974	-3.891	0.000104 ***
ilr1lr(..+-)	-0.4219	0.7671	-0.550	0.582430

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 19.81 on 1646 degrees of freedom

Multiple R-squared: 0.06251, Adjusted R-squared: 0.0551

F-statistic: 8.442 on 13 and 1646 DF, p-value: < 2.2e-16

```
car::Anova(lbp_intensity_lm)
```

Warning in printHypothesis(L, rhs, names(b)): one or more coefficients in the hypothesis include arithmetic operators in their names;  
the printed representation of the hypothesis will be omitted

Anova Table (Type II tests)

	Sum Sq	Df	F value	Pr(>F)
age	7709	2	9.8229	5.744e-05 ***
sex	3680	1	9.3784	0.002231 **
bmi	3279	2	4.1782	0.015489 *
stress	7043	1	17.9495	2.394e-05 ***
smoking	441	1	1.1228	0.289471
education	3435	1	8.7534	0.003134 **

```
ses          4605      2  5.8679  0.002888 **
ilr          11652      3  9.8986  1.810e-06 ***
Residuals 645859 1646
```

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

```
### This is logodds transform of outcome, not a good fit
# # move extreme values off boundary
# bpd_yes$intensity <- bpd_yes$LBP_intensity_year
# bpd_yes$intensity[bpd_yes$LBP_intensity_year < 0.5] <- 0.5
# bpd_yes$intensity[bpd_yes$LBP_intensity_year > (100 - 0.5)] <- 100 - 0.5
# bpd_yes$logodds_intensity <- with(bpd_yes, log((intensity / 100) / (1 - intensity / 100))
# bpd_yes$intensity <- NULL
# (mod_form_logodds_ilrs <- as.formula(paste0("logodds_intensity ~ ", rhs_formula, " + ilr
# lbp_intensity_logodds_lm <- lm(mod_form_logodds_ilrs, data = bpd_yes)
# summary(lbp_intensity_logodds_lm)
# car::Anova(lbp_intensity_logodds_lm)
# check_model(lbp_intensity_logodds_lm)

lbp_intensity_pois <- glm(mod_form_ilrs, family = "poisson", data = bpd_yes)
summary(lbp_intensity_pois)
```

Call:

```
glm(formula = mod_form_ilrs, family = "poisson", data = bpd_yes)
```

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )	
(Intercept)	3.693194	0.037276	99.077	< 2e-16	***
age2_middle	0.016658	0.009724	1.713	0.086684	.
age3_older	0.205646	0.014486	14.197	< 2e-16	***
sex2_male	-0.105547	0.010228	-10.319	< 2e-16	***
bmi2_normal	-0.017978	0.033512	-0.536	0.591644	
bmi3_overweight	0.069579	0.033867	2.054	0.039929	*
stress2_stressed	0.129912	0.009082	14.305	< 2e-16	***
smoking2_nonsmoker	0.040861	0.011211	3.645	0.000268	***
education2_higher	-0.095585	0.009529	-10.031	< 2e-16	***
ses2_middle	-0.112664	0.013084	-8.611	< 2e-16	***
ses3_higher	-0.209084	0.019007	-11.000	< 2e-16	***
ilrilr(++--)	-0.057087	0.009171	-6.225	4.83e-10	***
ilrilr(+--.)	-0.202889	0.015413	-13.164	< 2e-16	***
ilrilr(..+-)	-0.014136	0.006519	-2.168	0.030126	*



---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

Null deviance: 21235 on 1659 degrees of freedom  
Residual deviance: 19981 on 1646 degrees of freedom  
AIC: 28412

Number of Fisher Scoring iterations: 5

```
# check the goodness of fit test not significant,  
# p > 0.05: indicates model fit the data  
# p < 0.05: indicates model DOES NOIT fit the data  
with(  
  lbp_intensity_pois,  
  cbind(  
    res.deviance = deviance,  
    df = df.residual,  
    p = pchisq(deviance, df.residual, lower.tail = FALSE)  
  )  
)
```

```
      res.deviance    df p  
[1,]      19981.33 1646 0
```

```
lbp_intensity_nb <- glm.nb(mod_form_ilrs, data = bpd_yes)  
summary(lbp_intensity_nb)
```

Call:

```
glm.nb(formula = mod_form_ilrs, data = bpd_yes, init.theta = 2.48631581,  
       link = log)
```

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )	
(Intercept)	3.679463	0.143204	25.694	< 2e-16	***
age2_middle	0.016056	0.036738	0.437	0.662083	
age3_older	0.200337	0.057057	3.511	0.000446	***
sex2_male	-0.104393	0.038509	-2.711	0.006710	**
bmi2_normal	-0.003187	0.127766	-0.025	0.980099	

bmi3_overweight	0.086268	0.129300	0.667	0.504650	
stress2_stressed	0.133104	0.034966	3.807	0.000141	***
smoking2_nonsmoker	0.042290	0.042475	0.996	0.319415	
education2_higher	-0.099687	0.036963	-2.697	0.006998	**
ses2_middle	-0.105312	0.052931	-1.990	0.046636	*
ses3_higher	-0.207810	0.072984	-2.847	0.004409	**
ilrilr(++--)	-0.053990	0.036021	-1.499	0.133915	
ilrilr(+--)	-0.199774	0.059664	-3.348	0.000813	***
ilrilr(..+-)	-0.019407	0.025432	-0.763	0.445391	

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(2.4863) family taken to be 1)

Null deviance: 1983.2 on 1659 degrees of freedom  
 Residual deviance: 1897.9 on 1646 degrees of freedom  
 AIC: 14547

Number of Fisher Scoring iterations: 1

Theta: 2.4863  
 Std. Err.: 0.0942

2 x log-likelihood: -14517.0260

```
car::Anova(lbp_intensity_nb)
```

Analysis of Deviance Table (Type II tests)

Response: LBP\_intensity\_year

	LR	Chisq	Df	Pr(>Chisq)	
age	14.0426	2	0.0008927	***	
sex	7.2582	1	0.0070576	**	
bmi	6.8347	2	0.0327987	*	
stress	14.4891	1	0.0001410	***	
smoking	0.9800	1	0.3221987		
education	7.2378	1	0.0071383	**	
ses	8.0144	2	0.0181840	*	
ilr	23.4695	3	3.223e-05	***	

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

```
(est <- cbind(Estimate = coef(lbp_intensity_nb), confint(lbp_intensity_nb)))
```

Waiting for profiling to be done...

	Estimate	2.5 %	97.5 %
(Intercept)	3.679462775	3.40299626	3.969011870
age2_middle	0.016055946	-0.05598109	0.087856035
age3_older	0.200337071	0.08850561	0.313532033
sex2_male	-0.104393094	-0.17957684	-0.028571870
bmi2_normal	-0.003187056	-0.26156507	0.239011069
bmi3_overweight	0.086268083	-0.17496158	0.331713392
stress2_stressed	0.133103716	0.06446472	0.201989018
smoking2_nonsmoker	0.042290260	-0.04175439	0.125095866
education2_higher	-0.099687445	-0.17285762	-0.026981253
ses2_middle	-0.105311819	-0.21064524	-0.001965261
ses3_higher	-0.207809938	-0.35179957	-0.063592059
ilrilr(++--)	-0.053989622	-0.12392266	0.015888706
ilrilr(+--.)	-0.199773802	-0.31581509	-0.083907349
ilrilr(..+-)	-0.019407373	-0.06865999	0.029729611

```
exp(est)
```

	Estimate	2.5 %	97.5 %
(Intercept)	39.6251008	30.0540149	52.9322011
age2_middle	1.0161855	0.9455570	1.0918309
age3_older	1.2218145	1.0925404	1.3682493
sex2_male	0.9008711	0.8356237	0.9718324
bmi2_normal	0.9968180	0.7698458	1.2699926
bmi3_overweight	1.0900985	0.8394893	1.3933534
stress2_stressed	1.1423685	1.0665879	1.2238346
smoking2_nonsmoker	1.0431972	0.9591053	1.1332571
education2_higher	0.9051203	0.8412574	0.9733795
ses2_middle	0.9000438	0.8100614	0.9980367
ses3_higher	0.8123614	0.7034211	0.9383877
ilrilr(++--)	0.9474419	0.8834482	1.0160156
ilrilr(+--.)	0.8189160	0.7291943	0.9195164
ilrilr(..+-)	0.9807797	0.9336441	1.0301759

```

# likelihood ratio test
# dispersion parameter check is it equal to zero (if not, NB mod preferred)
pchisq(
  2 * (logLik(lbp_intensity_nb) - logLik(lbp_intensity_pois)),
  df = 1,
  lower.tail = FALSE
)

'log Lik.' 0 (df=15)

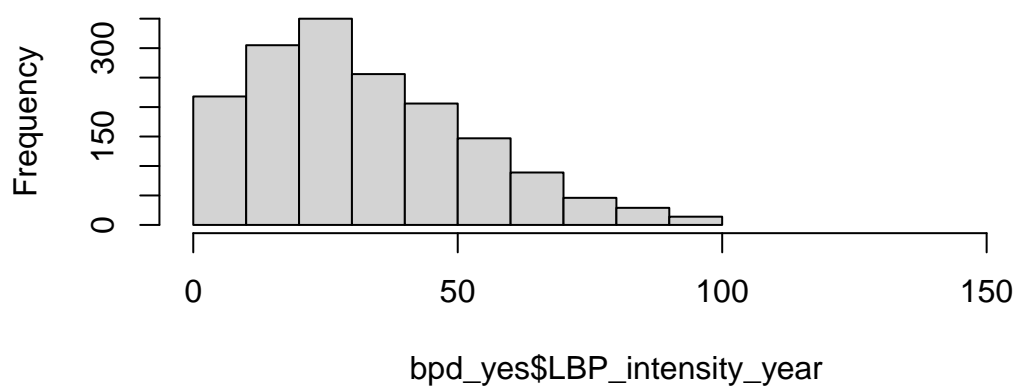
```

```

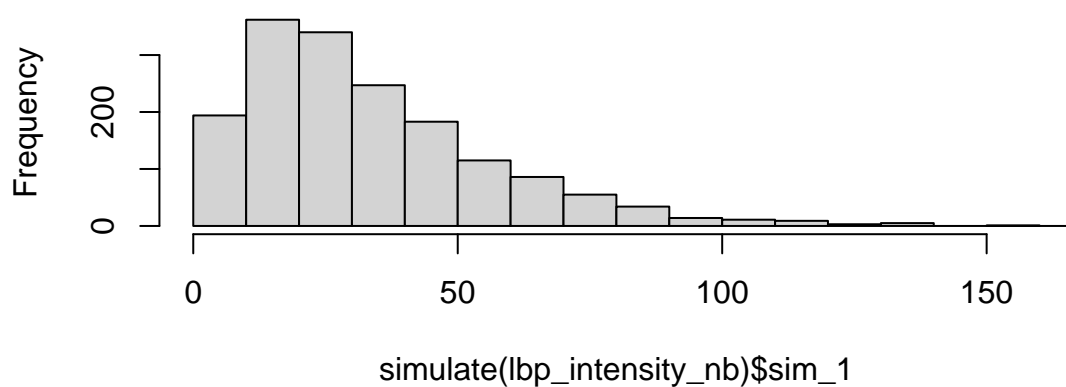
par(mfrow = c(2, 1))
hist(bpd_yes$LBP_intensity_year, xlim = c(0, 160), breaks = 10,
     main = "observed lower back pain intensity (0-100)")
hist(simulate(lbp_intensity_nb)$sim_1, xlim = c(0, 160), breaks = 20,
     main = "neg binomial predicted values (0-inf)")

```

### observed lower back pain intensity (0–100)



### neg binomial predicted values (0–inf)



```
par(mfrow = c(1, 1))
```

```
# Pain intensity could be categorised as (Boonstra et al., 2014):  
# - no pain (0)  
# - mild pain (1-38)  
# - moderate pain (39-57)  
# - severe pain (58-100)
```

```

bpd_yes$intens_ord <-
  cut(
    bpd_yes[[this_outcome]],
    breaks = c(-1, 0, 38, 57, 101),
    labels = c(
      "no pain (0)", "mild pain (1-38)",
      "moderate pain (39-57)", "severe pain (58-100)"
    )
  )
class(bpd_yes$intens_ord)

```

```
[1] "factor"
```

```

table(
  bpd_yes$intens_ord,
  cut(bpd_yes[[this_outcome]], breaks = c(-1, 0, 38, 57, 101)),
  useNA = "ifany"
)

```

	(-1,0]	(0,38]	(38,57]	(57,101]
no pain (0)	45	0	0	0
mild pain (1-38)	0	974	0	0
moderate pain (39-57)	0	0	401	0
severe pain (58-100)	0	0	0	240

```
(mod_form_ord_ilrs <- as.formula(paste0("intens_ord ~ ", rhs_formula, " + ilr")))
```

```

intens_ord ~ age + sex + bmi + stress + smoking + education +
  ses + ilr

```

```

## model __with__ ilrs
bpd_intens_ord_ilrs <- polr(mod_form_ord_ilrs, data = bpd_yes, Hess = TRUE, method = "logi
summary(bpd_intens_ord_ilrs)

```

Call:

```

polr(formula = mod_form_ord_ilrs, data = bpd_yes, Hess = TRUE,
  method = "logistic")

```

Coefficients:

	Value	Std. Error	t value
age2_middle	0.05704	0.11183	0.5100
age3_older	0.60483	0.16942	3.5700
sex2_male	-0.41189	0.11909	-3.4585
bmi2_normal	0.14649	0.38765	0.3779
bmi3_overweight	0.47744	0.39207	1.2177
stress2_stressed	0.43625	0.10534	4.1416
smoking2_nonsmoker	0.15893	0.12884	1.2335
education2_higher	-0.29335	0.11024	-2.6612
ses2_middle	-0.43164	0.15546	-2.7766
ses3_higher	-0.79524	0.22275	-3.5701
ilrilr(++--)	-0.14674	0.10909	-1.3451
ilrilr(+-.)	-0.60042	0.18119	-3.3138
ilrilr(..+-)	-0.04818	0.07715	-0.6245

Intercepts:

	Value	Std. Error	t value
no pain (0) mild pain (1-38)	-4.0583	0.4556	-8.9078
mild pain (1-38) moderate pain (39-57)	0.1189	0.4312	0.2757
moderate pain (39-57) severe pain (58-100)	1.4918	0.4332	3.4438

Residual Deviance: 3330.714

AIC: 3362.714

Anova(bpd\_intens\_ord\_ilrs)

Analysis of Deviance Table (Type II tests)

Response: intens\_ord

	LR	Chisq	Df	Pr(>Chisq)	
age	13.8442	2	0.0009858	***	
sex	12.1816	1	0.0004826	***	
bmi	10.7801	2	0.0045617	**	
stress	17.1554	1	3.444e-05	***	
smoking	1.5339	1	0.2155292		
education	7.0539	1	0.0079094	**	
ses	13.2177	2	0.0013484	**	
ilr	22.0152	3	6.476e-05	***	

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

```
# profiled CIs
est_ci_df <- cbind(est = coef(bpd_intens_ord_ilrs), confint(bpd_intens_ord_ilrs))
```

Waiting for profiling to be done...

```
kable(est_ci_df, digits = 3) # these are the log-odds scale estimates (and CI)
```

	est	2.5 %	97.5 %
age2_middle	0.057	-0.162	0.277
age3_older	0.605	0.272	0.937
sex2_male	-0.412	-0.647	-0.180
bmi2_normal	0.146	-0.595	0.932
bmi3_overweight	0.477	-0.273	1.271
stress2_stressed	0.436	0.230	0.643
smoking2_nonsmoker	0.159	-0.092	0.413
education2_higher	-0.293	-0.509	-0.077
ses2_middle	-0.432	-0.736	-0.126
ses3_higher	-0.795	-1.234	-0.360
ilr1r(++-)	-0.147	-0.361	0.067
ilr1r(+..)	-0.600	-0.957	-0.246
ilr1r(..+)	-0.048	-0.199	0.103

```
kable(exp(est_ci_df), digits = 3) # these are the odds ratios (and approx CIs)
```

	est	2.5 %	97.5 %
age2_middle	1.059	0.851	1.319
age3_older	1.831	1.313	2.551
sex2_male	0.662	0.524	0.835
bmi2_normal	1.158	0.551	2.539
bmi3_overweight	1.612	0.761	3.564
stress2_stressed	1.547	1.258	1.902
smoking2_nonsmoker	1.172	0.912	1.512
education2_higher	0.746	0.601	0.926
ses2_middle	0.649	0.479	0.882
ses3_higher	0.451	0.291	0.697
ilr1r(++-)	0.864	0.697	1.069
ilr1r(+..)	0.549	0.384	0.782



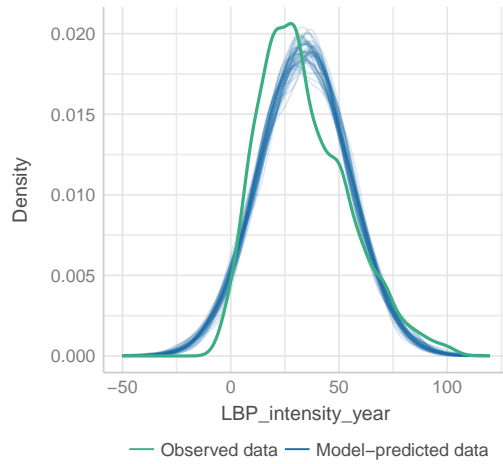
	est	2.5 %	97.5 %
ilr1r(..+-)	0.953	0.819	1.109

### 5.4.2 Model diagnostics

```
## plain linear model
check_model(lbp_intensity_lm) # acceptable?
```

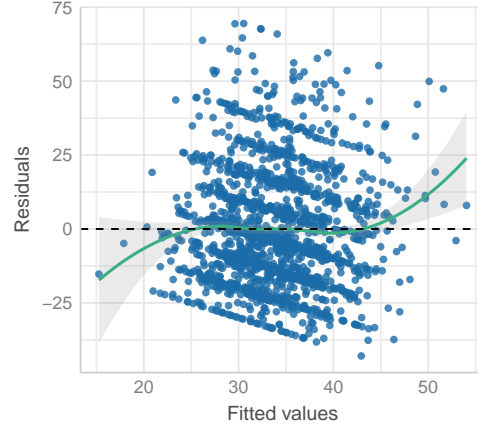
### Posterior Predictive Check

Model-predicted lines should resemble observed data line



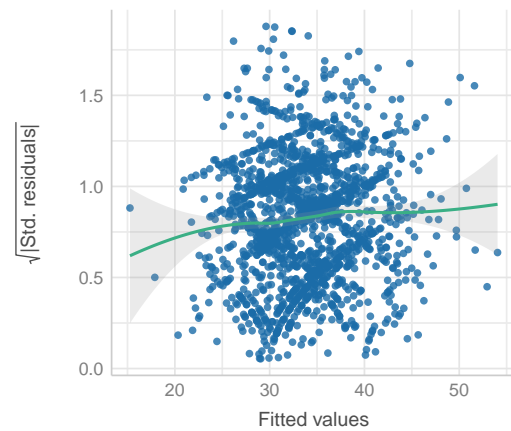
### Linearity

Reference line should be flat and horizontal



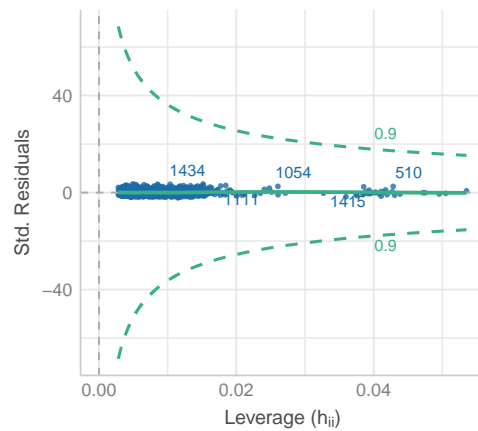
### Homogeneity of Variance

Reference line should be flat and horizontal



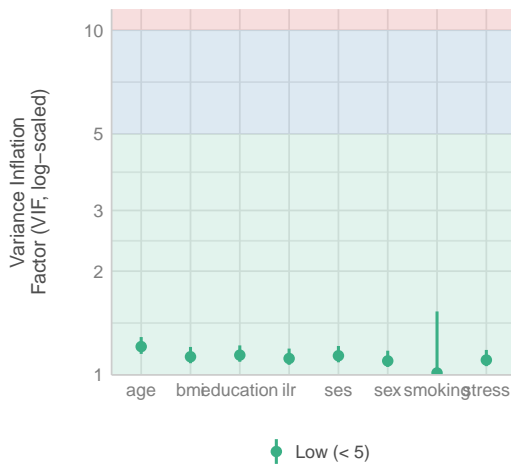
### Influential Observations

Points should be inside the contour lines



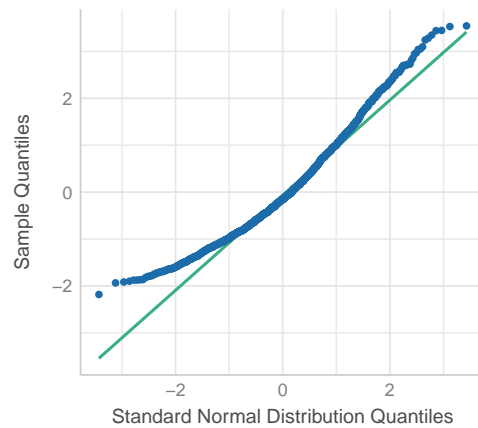
### Collinearity

High collinearity (VIF) may inflate parameter uncertainty



### Normality of Residuals

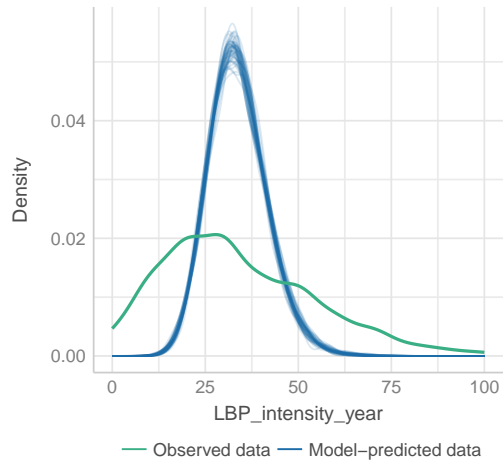
Dots should fall along the line



```
## Poisson regression (bad)
check_model(lbp_intensity_pois) # horrible
```

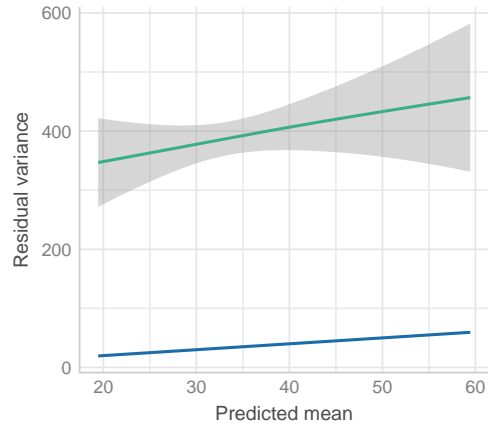
### Posterior Predictive Check

Model-predicted lines should resemble observed data line



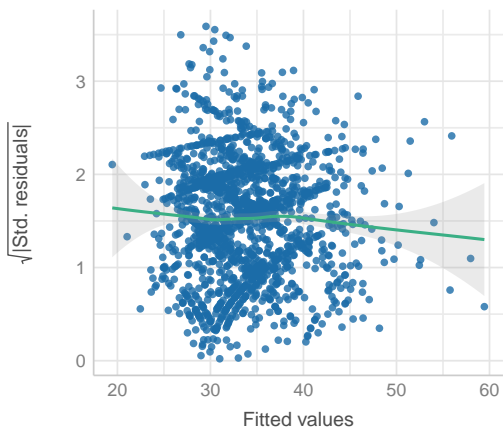
### Overdispersion and zero-inflation

Observed residual variance (green) should follow predicted res



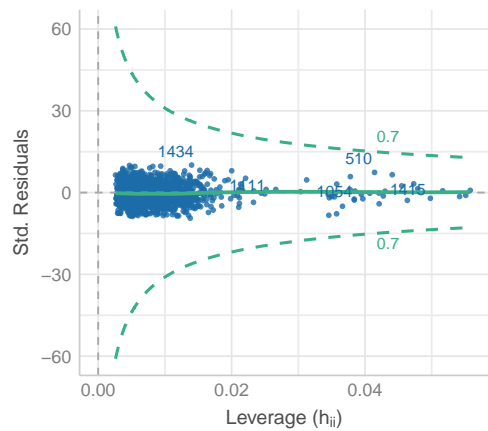
### Homogeneity of Variance

Reference line should be flat and horizontal



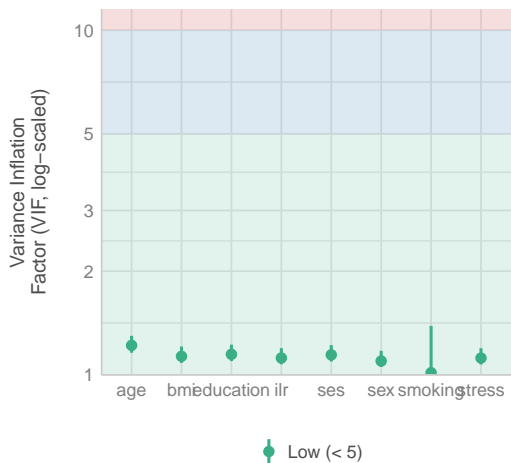
### Influential Observations

Points should be inside the contour lines



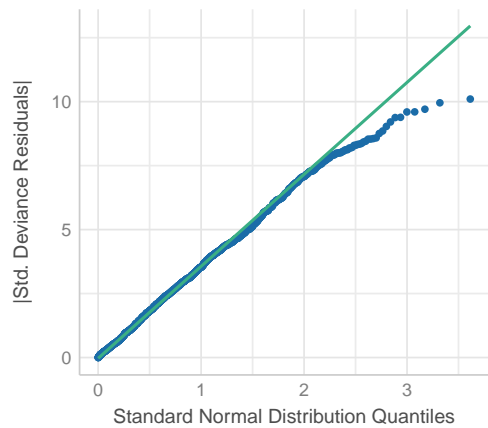
### Collinearity

High collinearity (VIF) may inflate parameter uncertainty



### Normality of Residuals

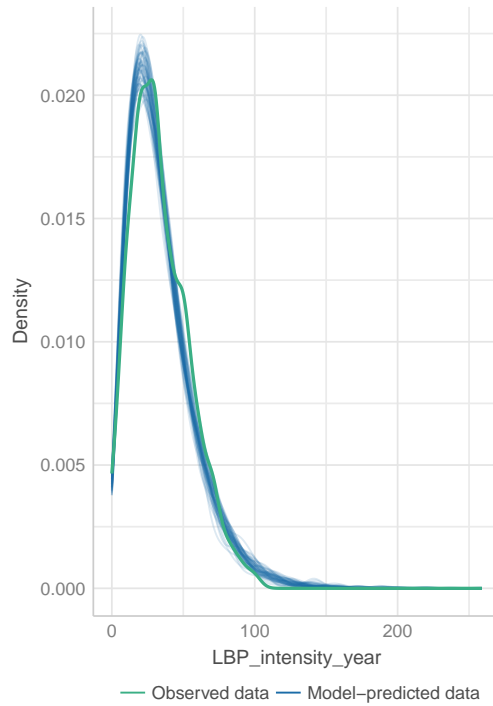
Dots should fall along the line



```
## Negative Binomial regression
check_model(
  lbp_intensity_nb,
  check = c("pp_check", "homogeneity", "outliers", "vif")
)
```

### Posterior Predictive Check

Model-predicted lines should resemble observed data line



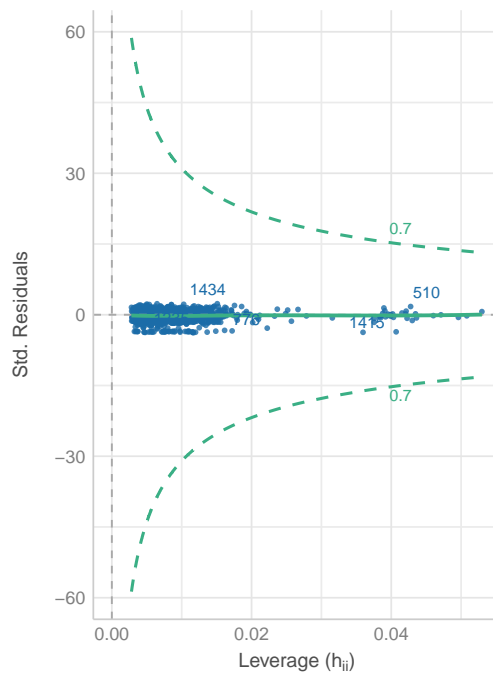
### Homogeneity of Variance

Reference line should be flat and horizontal



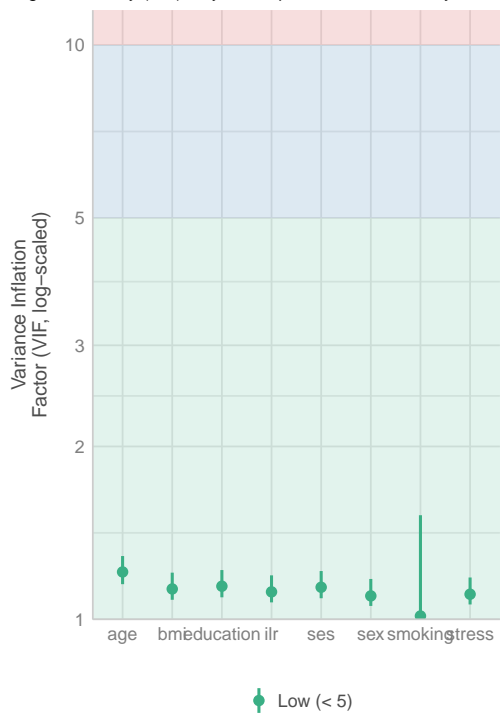
### Influential Observations

Points should be inside the contour lines

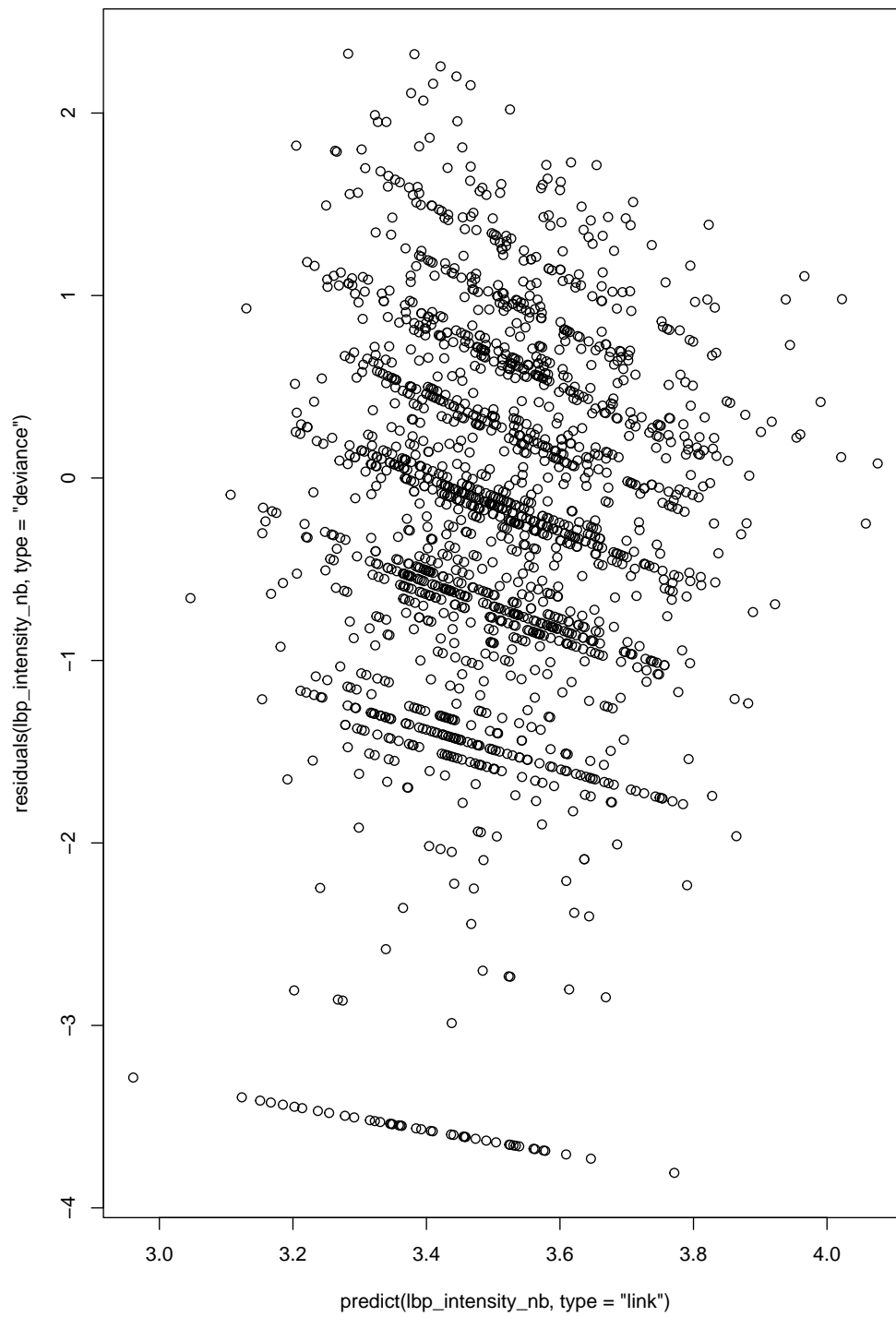


### Collinearity

High collinearity (VIF) may inflate parameter uncertainty



```
plot(  
  predict(lbp_intensity_nb, type = "link"),  
  residuals(lbp_intensity_nb, type = "deviance")  
)
```





```
deviance(lbp_intensity_nb)/lbp_intensity_nb$df.residual
```

```
[1] 1.153043
```

```
## Ordinal logistic regression (looks ok)
# this is running multiple logistic regressions
## we want to see the coefficients to be roughly the same EXCEPT for the
## (intercept) values
foreach(i = 2:length(levels(bpd_yes$intens_ord)), .combine = cbind) %do% {
  log_coefs <-
    coef(glm(
      I(as.numeric(intens_ord) >= i) ~
        age + sex + bmi + stress + smoking + education + ses + ilr,
      family = "binomial",
      data = bpd_yes
    ))
  log_coefs <- as.data.frame(log_coefs)
  colnames(log_coefs) <- paste0("logit(P(Y>=", i, "))")
  log_coefs
} %>%
  kable(., digits = 2)
```

	logit(P(Y>=2))	logit(P(Y>=3))	logit(P(Y>=4))
(Intercept)	5.80	-0.17	-1.95
age2_middle	-0.28	0.08	0.16
age3_older	0.49	0.62	0.65
sex2_male	-0.76	-0.38	-0.41
bmi2_normal	0.56	0.06	0.28
bmi3_overweight	0.73	0.38	0.68
stress2_stressed	0.52	0.46	0.40
smoking2_nonsmoker	0.34	0.09	0.37
education2_higher	-0.21	-0.36	-0.07
ses2_middle	-0.40	-0.34	-0.70
ses3_higher	-1.14	-0.68	-0.96
ilr(lr(++-))	0.23	-0.17	-0.01
ilr(lr(++..))	-0.67	-0.57	-0.38
ilr(lr(+++-))	-1.03	0.03	-0.07

### 5.4.3 Model predictions

```
# create dataset for predictions
newdata <-
  bpd_yes %>%
  dplyr::select(all_of(pred_covs), ilr) %>%
  distinct(pick(all_of(pred_covs)), .keep_all = TRUE) %>%
  arrange(pick(all_of(pred_covs)))

(mean_ilr <- mean(bpd_yes$ilr))

ilr(++--) ilr(+-. ) ilr(..+-)
1.37128443 0.05346627 2.07785318
attr(,"class")
[1] "rmult"

dev_null <- foreach(i = 1:nrow(newdata)) %do% {
  newdata$ilr[i, ] <- mean_ilr
}

# make preds and then put in long format for ggplot
predictions_intens <-
  cbind(
    pain_intens = predict(lbp_intensity_nb, newdata, type = "response"),
    newdata
  ) %>%
  dplyr::select(-ilr)

head(predictions_intens)
```

	pain_intens	age	sex	bmi	stress	smoking	education
1	31.64985	1_younger	1_female	1_underweight	1_normal	1_smoker	2_higher
2	28.48625	1_younger	1_female	1_underweight	1_normal	1_smoker	2_higher
3	36.47806	1_younger	1_female	1_underweight	1_normal	2_nonsmoker	1_lower
4	32.83186	1_younger	1_female	1_underweight	1_normal	2_nonsmoker	1_lower
5	29.71678	1_younger	1_female	1_underweight	1_normal	2_nonsmoker	2_higher
6	32.54180	1_younger	1_female	1_underweight	2_stressed	1_smoker	2_higher

```
ses
1 1_lower
2 2_middle
```

```

3 1_lower
4 2_middle
5 2_middle
6 2_middle

```

```

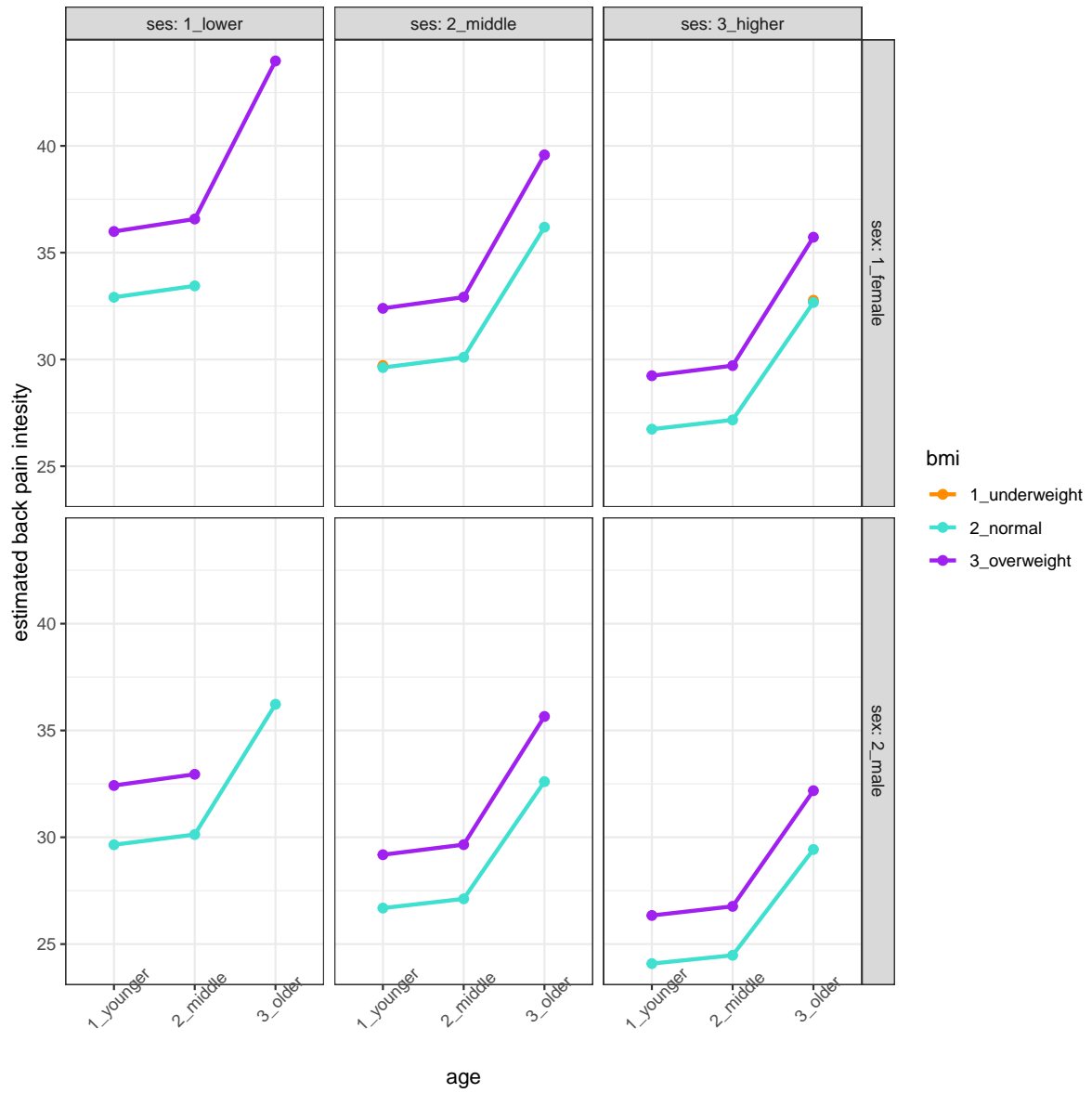
# newdata2 <- cbind(newdata2, predict(lbp_nb, newdata2, type = "link", se.fit=TRUE))
# newdata2 <- within(newdata2, {
#   lbp_pred <- exp(fit)
#   LL <- exp(fit - 1.96 * se.fit)
#   UL <- exp(fit + 1.96 * se.fit)
# })

```

```

predictions_intens %>%
  dplyr::filter(
    # sex == "1_female",
    stress == "1_normal",
    smoking == "2_nonsmoker",
    education == "2_higher",
    # ses == "2_middle"
  ) %>%
  ggplot(., aes(age, pain_intens, group = bmi)) +
  geom_line(aes(colour = bmi), linewidth = 1) +
  geom_point(aes(colour = bmi), size = 2) +
  facet_grid(sex~ ses, labeller = label_both) +
  labs(x = "age", y = "estimated back pain intensity") +
  theme_bw() +
  scale_color_manual(values = c("darkorange", "turquoise", "purple")) +
  theme(axis.text.x = element_text(angle = 45))

```



### 5.4.3.1 Absolute scale

```
# create a RHS of regression equation dataset for time-reallocation
(predict_basis <-
  bpd_yes %>%
  dplyr::select(all_of(pred_covs), all_of(pred_comps)) %>%
  dplyr::filter(
    age == "2_middle",
    sex == "1_female",
    stress == "1_normal",
    smoking == "2_nonsmoker",
    education == "2_higher",
    ses == "2_middle",
    bmi == "2_normal"
  ) %>%
  distinct(across(all_of(pred_covs)), .keep_all = TRUE) %>%
  as.data.frame())
```

	age	sex	bmi	stress	smoking	education	ses	Time_Sleep
1	2_middle	1_female	2_normal	1_normal	2_nonsmoker	2_higher	2_middle	546.4286
	Time_Sedentary	Time_LPA	Time_MVPA					
1	418.2857	435.1429	40					

```
# compositional mean: geometric mean to closure
# (comp_mean <- mean(acomp(bpd_yes[, pred_comps])))
(comp_mean <- calc_comp_mean(bpd_yes[, pred_comps], clo_val = 1440))
```

Time_Sleep	Time_Sedentary	Time_LPA	Time_MVPA
472.8407	438.4062	502.1666	26.5865

```
predict_basis0 <- predict_basis
predict_basis0[, pred_comps] <- comp_mean
```

```
predict_basis0
```

	age	sex	bmi	stress	smoking	education	ses	Time_Sleep
1	2_middle	1_female	2_normal	1_normal	2_nonsmoker	2_higher	2_middle	472.8407
	Time_Sedentary	Time_LPA	Time_MVPA					
1	438.4062	502.1666	26.5865					

```

# +15 minutes to Time_MVPA and -15 minutes from Time_Sedentary
comp_mean_changed <- comp_mean
comp_mean_changed["Time_MVPA"] <- comp_mean_changed["Time_MVPA"] + 15
comp_mean_changed["Time_Sedentary"] <- comp_mean_changed["Time_Sedentary"] - 15
# check
comp_mean_changed - comp_mean

```

Time_Sleep	Time_Sedentary	Time_LPA	Time_MVPA
0	-15	0	15

```

predict_basis1 <- predict_basis
predict_basis1[, pred_comps] <- comp_mean_changed

pred_df <- rbind(predict_basis0, predict_basis1)
pred_df <- add_ilrs_to_data(pred_df, comp_vars = pred_comps, sbp_matrix = sbp1)
pred_df

```

	age	sex	bmi	stress	smoking	education	ses	Time_Sleep
1	2_middle	1_female	2_normal	1_normal	2_nonsmoker	2_higher	2_middle	472.8407
2	2_middle	1_female	2_normal	1_normal	2_nonsmoker	2_higher	2_middle	472.8407
	Time_Sedentary	Time_LPA	Time_MVPA	ilr.1	ilr.2	ilr.3		
1	438.4062	502.1666	26.5865	1.37128443	0.05346627	2.07785318		
2	423.4062	502.1666	41.5865	1.13019151	0.07808340	1.76151343		

```

predict(lbp_intensity_nb, pred_df, type = "link")

```

	1	2
	3.404581	3.418819

```

# exponentiate difference in the log back pain intensity (ratio of back pain preds)
exp(diff(predict(lbp_intensity_nb, pred_df, type = "link")))

```

	2
	1.01434

```

# abs difference in the mean back pain intensity
diff(predict(lbp_intensity_nb, pred_df, type = "response"))

```

2  
0.4316527

```
(p_0 <- predict(lbp_intensity_nb, pred_df, type = "response"))
```

1 2  
30.10167 30.53332

```
# % increase in pain intensity  
(p_0[2] - p_0[1]) / p_0[1]
```

2  
0.01433983

```
# ratio version  
get_pred_diff_rat <- function(mod, new_dat) {  
  log_ratio_pred <- predict(mod, new_dat, type = "link")  
  ratio_outc <- exp(log_ratio_pred[2] - log_ratio_pred[1])  
  return(ratio_outc)  
}  
get_pred_diff_rat(lbp_intensity_nb, pred_df)
```

2  
1.01434

```
# absolute difference version  
get_pred_diff_abs <- function(mod, new_dat) {  
  log_ratio_pred <- predict(mod, new_dat, type = "response")  
  ratio_outc <- log_ratio_pred[2] - log_ratio_pred[1]  
  return(ratio_outc)  
}  
get_pred_diff_abs(lbp_intensity_nb, pred_df)
```

2  
0.4316527

```
# wrapper:
get_pred_diff <- function(mod, new_dat, type = "abs") {
  if (type == "abs") {
    return(get_pred_diff_abs(mod = mod, new_dat = new_dat))
  } else if (type == "rat") {
    return(get_pred_diff_rat(mod = mod, new_dat = new_dat))
  } else {
    stop("'type' must be 'abs' (absolute difference) or 'rat' (ratio)")
  }
}
get_pred_diff(lbp_intensity_nb, pred_df, type = "abs")
```

```
      2
0.4316527
```

```
get_pred_diff(lbp_intensity_nb, pred_df, type = "rat")
```

```
      2
1.01434
```

```
fit_mod_boot <- function(data, i, pred_dat, type = "abs") {

  this_dat <- data[i, ]
  this_nbr <- glm.nb(mod_form_ilrs, data = this_dat)
  est <- get_pred_diff(this_nbr, new_dat = pred_dat, type = type)
  return(est)

}
alpha <- 0.05
quantile(boot(bpd_yes, fit_mod_boot, R = 10, pred_dat = pred_df)$t, c(alpha / 2, 1 - alpha))
```

```
      2.5%      97.5%
0.05446169 0.54096532
```

```
do_multi_realloc <- function(mod, basis_data, timeusenames, time_changes, sbp_matrix = sbp_matrix) {

  x0 <- basis_data
```



```

plot_dat <-
  foreach(i = 1:length(timeusenames), .combine = bind_rows) %do% {
    print(paste("i: ", i))
    foreach(j = 1:length(timeusenames), .combine = bind_rows) %do% {
      print(paste("  j: ", j))
      foreach(d = 1:length(time_changes), .combine = bind_rows) %do% {
        print(paste("    d: ", d))

        timeuse_to <- timeusenames[i]
        timeuse_from <- timeusenames[j]
        change_time <- time_changes[d]

        proposed_change_1 <- x0[timeuse_to] + change_time
        proposed_change_2 <- x0[timeuse_from] - change_time

        if (timeuse_to == timeuse_from) {
          NULL # reallocation exceeds 0 or max time
        } else if ((proposed_change_1 < 0) | (proposed_change_1 > 1440)) {
          NULL # reallocation exceeds 0 or max time
        } else if ((proposed_change_2 < 0) | (proposed_change_2 > 1440)) {
          NULL # reallocation exceeds 0 or max time
        } else {

          x1 <- x0
          x1[timeuse_to] <- x1[timeuse_to] + change_time
          x1[timeuse_from] <- x1[timeuse_from] - change_time

          pred_df <- rbind(x0, x1)
          pred_df <- add_ilrs_to_data(pred_df, comp_vars = timeusenames, sbp_matrix = sbp_matrix)

          outc_ratio <- get_pred_diff(mod, pred_df)

          bootstrapped_ests <- boot(bpd_yes, fit_mod_boot, R = 1000, pred_dat = pred_df)
          ci_est <- quantile(as.numeric(bootstrapped_ests), c(alpha / 2, 1 - alpha / 2))

          tibble(
            to = timeuse_to,
            from = timeuse_from,
            change_time = change_time,
            outc_ratio = outc_ratio,
            ci_lo = ci_est[1],

```

```

        ci_hi = ci_est[2]
      )
    }
  }
}

plot_dat$to <- factor(plot_dat$to, levels = timeusenames)
plot_dat$from <- factor(plot_dat$from, levels = timeusenames)

return(plot_dat)
}

set.seed(1234)

# takes ~60 min (single core) for bootstrapped CIs (R = 1000)
# takes ~ 6 min (single core) for bootstrapped CIs (R = 100)

### Uncomment to generate bootstrapping
# tic()
# realloc_plot_data <-
#   do_multi_realloc(
#     lbp_intensity_nb,
#     predict_basis0,
#     pred_comps,
#     seq(-30, 30, by = 10)
#   )
# saveRDS(realloc_plot_data, file = "res/negbin_realloc_boot_res(abs).rda")
# toc()

realloc_plot_data <- readRDS(file = "res/negbin_realloc_boot_res(abs).rda")

levels(realloc_plot_data$to) <- paste0(levels(realloc_plot_data$to), "+Delta")
levels(realloc_plot_data$from) <- paste0(levels(realloc_plot_data$from), "-Delta")

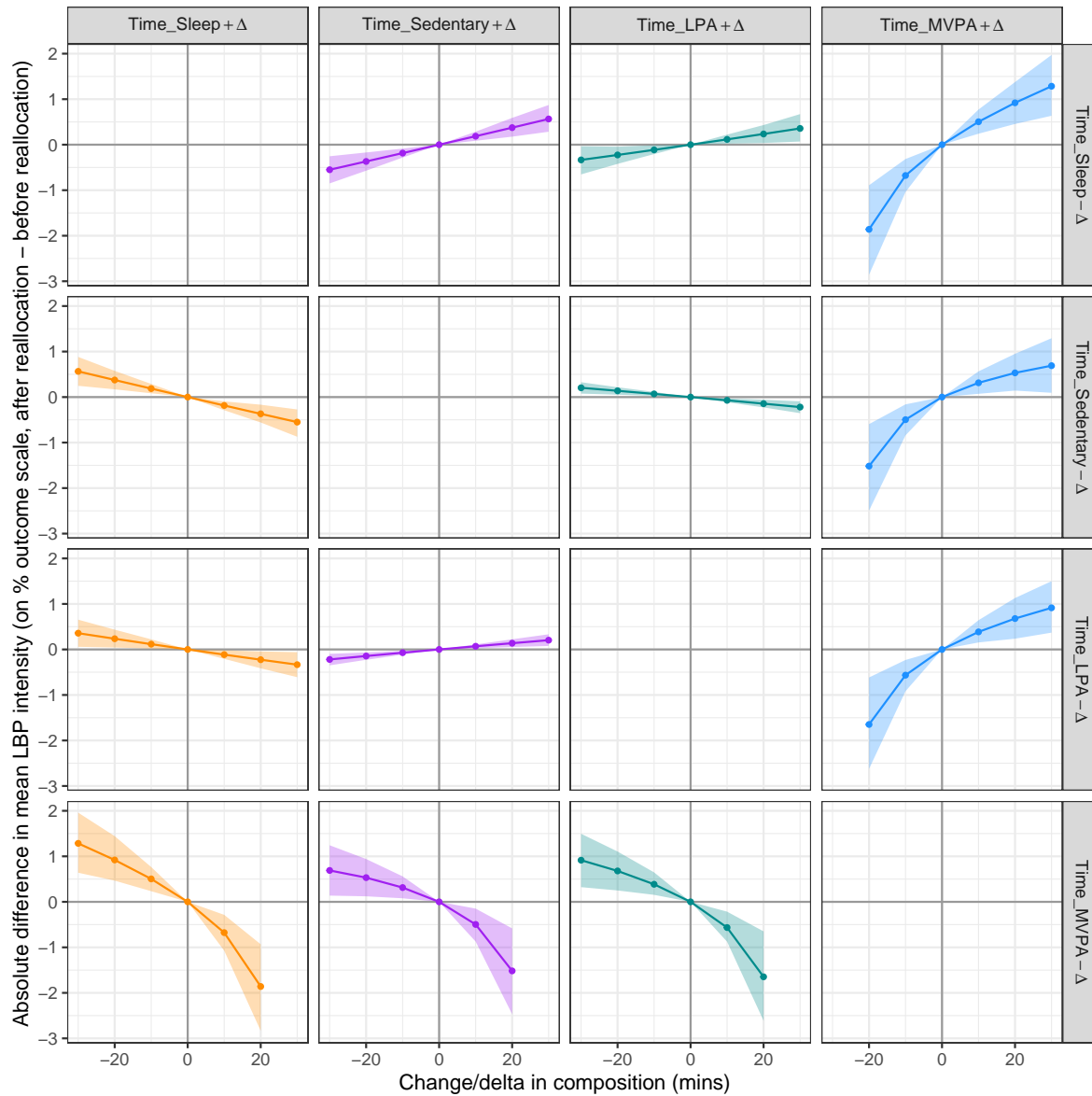
ggplot(realloc_plot_data) +

```

```

geom_vline(xintercept = 0, col = "grey60") +
geom_hline(yintercept = 0, col = "grey60") +
geom_ribbon(aes(x = change_time, ymin = ci_lo, ymax = ci_hi, fill = to), alpha = 0.3) +
geom_line(aes(x = change_time, y = outc_ratio, col = to)) +
geom_point(aes(x = change_time, y = outc_ratio, col = to), size = 1) +
facet_grid(from ~ to, labeller = label_parsed) +
theme_bw() +
scale_colour_manual(values = c("darkorange", "purple", "cyan4", "dodgerblue")) +
scale_fill_manual(values = c("darkorange", "purple", "cyan4", "dodgerblue")) +
labs(
  x = paste0("Change/delta in composition (mins)"),
  y = paste0(
    "Absolute difference in mean LBP intensity (on % outcome scale, ",
    "after reallocation - before reallocation)")
) +
theme(legend.position = "none")

```



```
ggsave(
  filename = "fig/lbp_intens_negbin_abs_v1.png",
  dpi = 600, # print quality
  width = 10,
  height = 10
)
```

```

pd2 <-
  realloc_plot_data %>%
  mutate(
    to = gsub("Time_", "", to),
    from = gsub("Time_", "", from),
    to = gsub("+Delta", "", to, fixed = TRUE),
    from = gsub("-Delta", "", from, fixed = TRUE),
    to_len = nchar(to),
    to_max = max(to_len),
    from_len = nchar(from),
    from_max = max(from_len),
    to_pad = rep_char(pmax(0, from_max - to_len)),
    from_pad = rep_char(pmax(0, to_max - from_len)),
    to = factor(to, levels = time_lvls),
    from = factor(from, levels = time_lvls),
    to_num = as.numeric(to),
    from_num = as.numeric(from)
  ) %>%
  dplyr::filter(to_num > from_num) %>%
  mutate(
    # from_to = paste0("      ", "+", from, rep_char(10), from_pad, "\u2194", to_pad, rep_c
    from_to = paste0("+", from, rep_char(13), from_pad, "", to_pad, rep_char(13), "+", to)
  ) %>%
  arrange(from, to)

unique(pd2$from_to)

```

```

[1] "+Sleep          +Sedentary"
[2] "+Sleep          +LPA"
[3] "+Sleep          +MVPA"
[4] "+Sedentary      +LPA"
[5] "+Sedentary      +MVPA"
[6] "+LPA            +MVPA"

```

```

pd2$from_to <- factor(pd2$from_to, levels = unique(pd2$from_to))

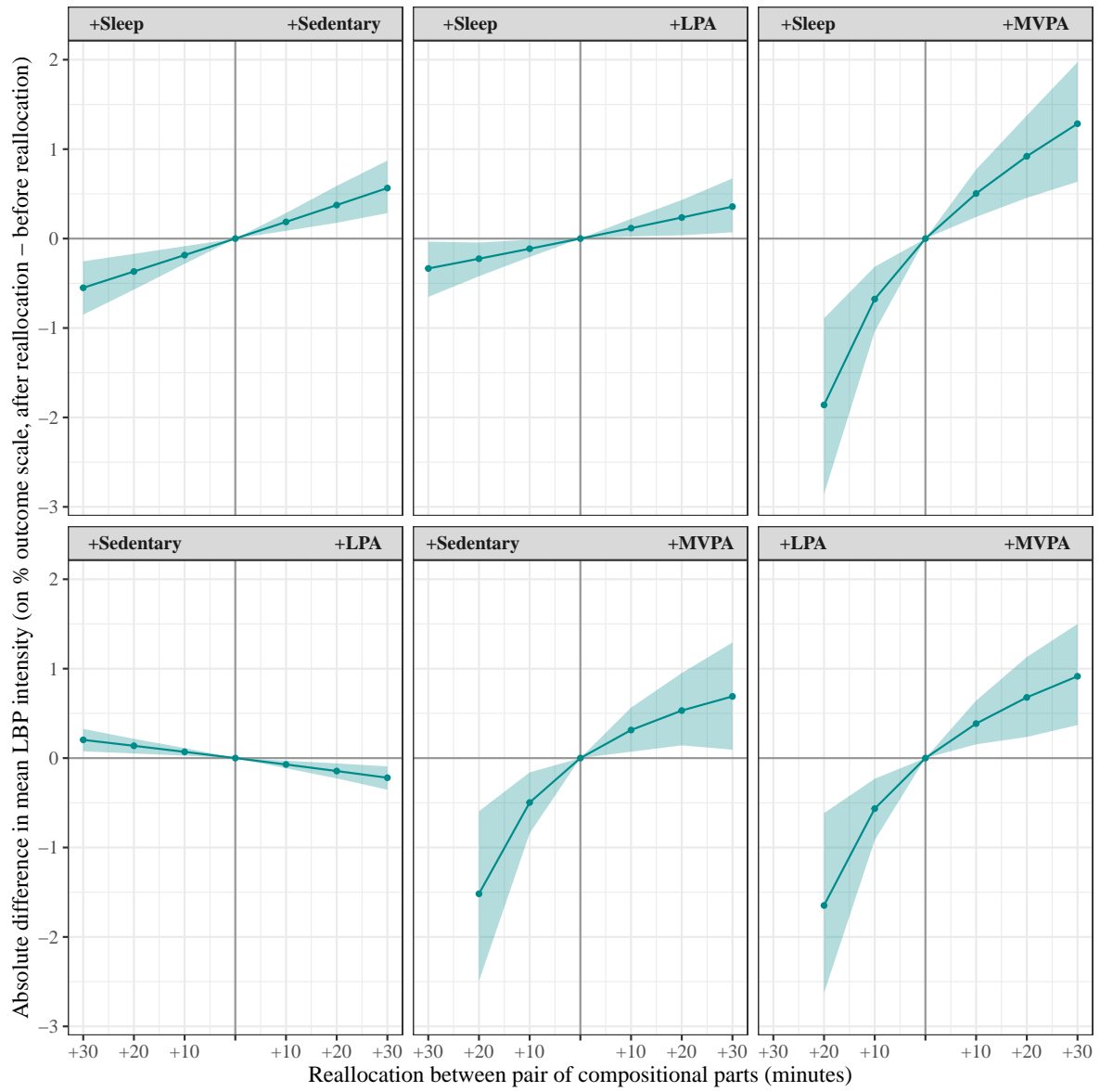
this_breaks <- seq(-30, 30, 10)
this_labs <- sprintf("+%2.0f", abs(seq(-30, 30, 10)))
this_labs[this_labs == "+ 0"] <- ""

```

```
this_labs
```

```
[1] "+30" "+20" "+10" ""      "+10" "+20" "+30"
```

```
ggplot(pd2) +
  geom_vline(xintercept = 0, col = "grey60") +
  geom_hline(yintercept = 0, col = "grey60") +
  geom_ribbon(aes(x = change_time, ymin = ci_lo, ymax = ci_hi, fill = to), alpha = 0.3, col = "grey60") +
  geom_line(aes(x = change_time, y = outc_ratio, col = to), col = "cyan4") +
  geom_point(aes(x = change_time, y = outc_ratio, col = to), size = 1, col = "cyan4") +
  facet_wrap(~ from_to, labeller = label_bquote(.(from_to))) +
  theme_bw() +
  scale_x_continuous(breaks = this_breaks, labels = this_labs) +
  labs(
    x = paste0("Reallocation between pair of compositional parts (minutes)"),
    y = paste0(
      "Absolute difference in mean LBP intensity (on % outcome scale, ",
      "after reallocation - before reallocation)"
    )
  )
# subtitle = "Note that odds ratios relate to the probability of having _increased_ fr
) +
theme(
  legend.position = "none",
  text = element_text(family = "serif"),
  strip.text = element_text(size = 10, face = "bold"),
  axis.text = element_text(size = 10),
  axis.title = element_text(size = 12)
)
```



```
ggsave(filename = "fig/lbp_intens_negbin_abs_v2.png", width = 14, height = 9, dpi = 600)
```

### 5.4.3.2 Ratio scale

```
# wrapper:
get_pred_diff <- function(mod, new_dat, type = "rat") {
  if (type == "abs") {
    return(get_pred_diff_abs(mod = mod, new_dat = new_dat))
  } else if (type == "rat") {
    return(get_pred_diff_rat(mod = mod, new_dat = new_dat))
  } else {
    stop("'type' must be 'abs' (absolute difference) or 'rat' (ratio)")
  }
}
get_pred_diff(lbp_intensity_nb, pred_df, type = "abs")
```

```
      2
0.4316527
```

```
get_pred_diff(lbp_intensity_nb, pred_df, type = "rat")
```

```
      2
1.01434
```

```
get_pred_diff(lbp_intensity_nb, pred_df)
```

```
      2
1.01434
```

```
fit_mod_boot <- function(data, i, pred_dat, type = "rat") {

  this_dat <- data[i, ]
  this_nbr <- glm.nb(mod_form_ilrs, data = this_dat)
  est <- get_pred_diff(this_nbr, new_dat = pred_dat, type = type)
  return(est)

}
alpha <- 0.05
quantile(boot(bpd_yes, fit_mod_boot, R = 10, pred_dat = pred_df)$t, c(alpha / 2, 1 - alpha))
```



2.5% 97.5%  
1.009664 1.026014

```
# takes ~60 min (single core) for bootstrapped CIs (R = 1000)
# takes ~ 6 min (single core) for bootstrapped CIs (R = 100)
```

```
### Uncomment to generate bootstrapping
```

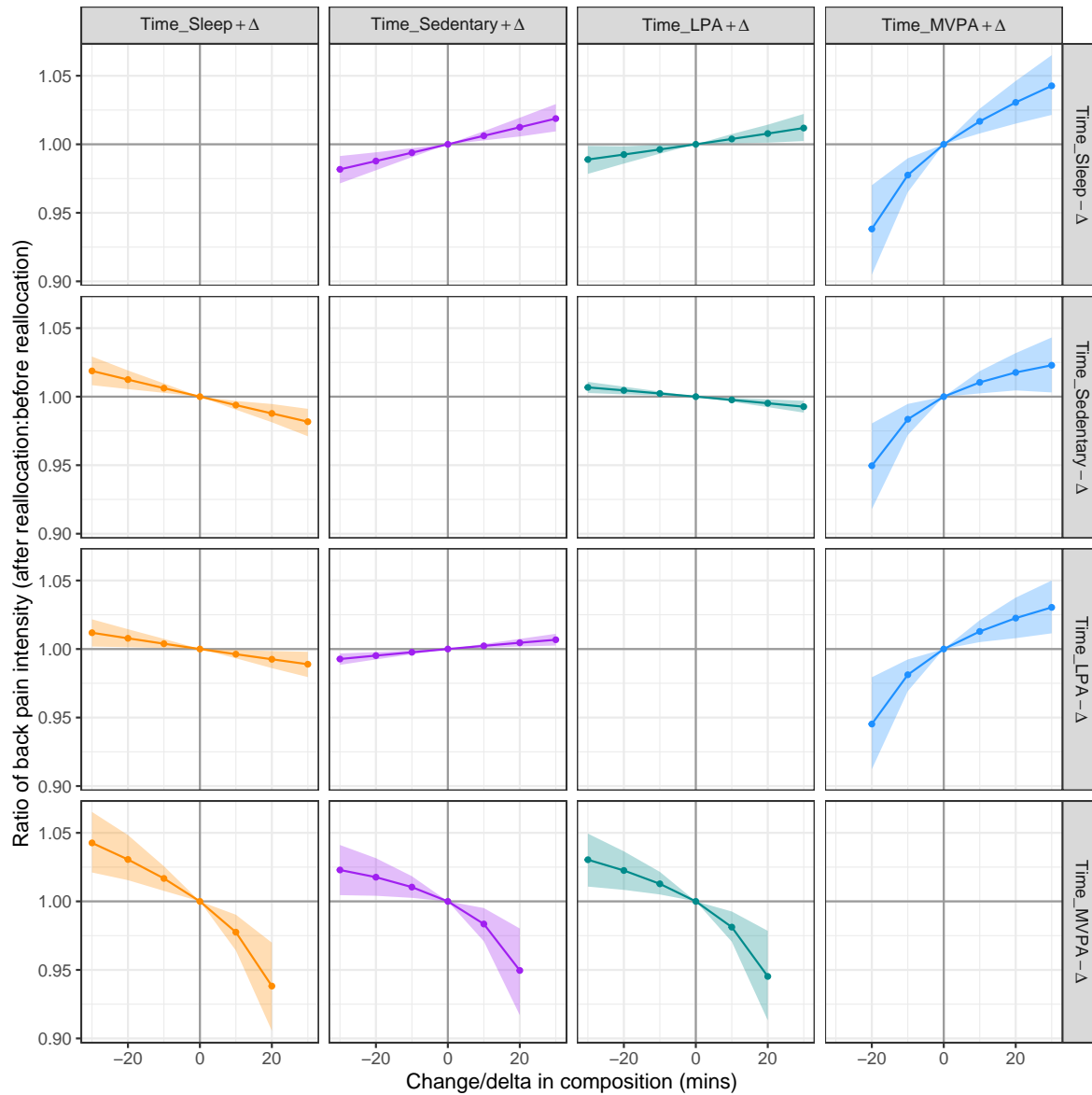
```
# set.seed(1234)
# tic()
# realloc_plot_data <-
#   do_multi_realloc(
#     lbp_intensity_nb,
#     predict_basis0,
#     pred_comps,
#     seq(-30, 30, by = 10)
#   )
# saveRDS(realloc_plot_data, file = "res/negbin_realloc_boot_res(rat).rda")
# toc()
```

```
realloc_plot_data <- readRDS(file = "res/negbin_realloc_boot_res(rat).rda")
```

```
levels(realloc_plot_data$to) <- paste0(levels(realloc_plot_data$to), "+Delta")
levels(realloc_plot_data$from) <- paste0(levels(realloc_plot_data$from), "-Delta")
```

```
ggplot(realloc_plot_data) +
  geom_vline(xintercept = 0, col = "grey60") +
  geom_hline(yintercept = 1, col = "grey60") +
  geom_ribbon(aes(x = change_time, ymin = ci_lo, ymax = ci_hi, fill = to), alpha = 0.3) +
  geom_line(aes(x = change_time, y = outc_ratio, col = to)) +
  geom_point(aes(x = change_time, y = outc_ratio, col = to), size = 1) +
  facet_grid(from ~ to, labeller = label_parsed) +
  theme_bw() +
  scale_colour_manual(values = c("darkorange", "purple", "cyan4", "dodgerblue")) +
  scale_fill_manual(values = c("darkorange", "purple", "cyan4", "dodgerblue")) +
  labs(
    x = paste0("Change/delta in composition (mins)"),
    y = paste0("Ratio of back pain intensity (after reallocation:before reallocation)")
  )
```

```
) +  
theme(legend.position = "none")
```



```
ggsave(  
  filename = "fig/lbp_intens_negbin_rat_v1.png",  
  dpi = 600, # print quality
```

```

width = 10,
height = 10
)

pd2 <-
  realloc_plot_data %>%
  mutate(
    to = gsub("Time_", "", to),
    from = gsub("Time_", "", from),
    to = gsub("+Delta", "", to, fixed = TRUE),
    from = gsub("-Delta", "", from, fixed = TRUE),
    to_len = nchar(to),
    to_max = max(to_len),
    from_len = nchar(from),
    from_max = max(from_len),
    to_pad = rep_char(pmax(0, from_max - to_len)),
    from_pad = rep_char(pmax(0, to_max - from_len)),
    to = factor(to, levels = time_lvls),
    from = factor(from, levels = time_lvls),
    to_num = as.numeric(to),
    from_num = as.numeric(from)
  ) %>%
  dplyr::filter(to_num > from_num) %>%
  mutate(
    # from_to = paste0("      ", "+", from, rep_char(10), from_pad, "\u2194", to_pad, rep_c
    from_to = paste0("+", from, rep_char(13), from_pad, "", to_pad, rep_char(13), "+", to)
  ) %>%
  arrange(from, to)

unique(pd2$from_to)

```

```

[1] "+Sleep          +Sedentary"
[2] "+Sleep          +LPA"
[3] "+Sleep          +MVPA"
[4] "+Sedentary      +LPA"
[5] "+Sedentary      +MVPA"
[6] "+LPA            +MVPA"

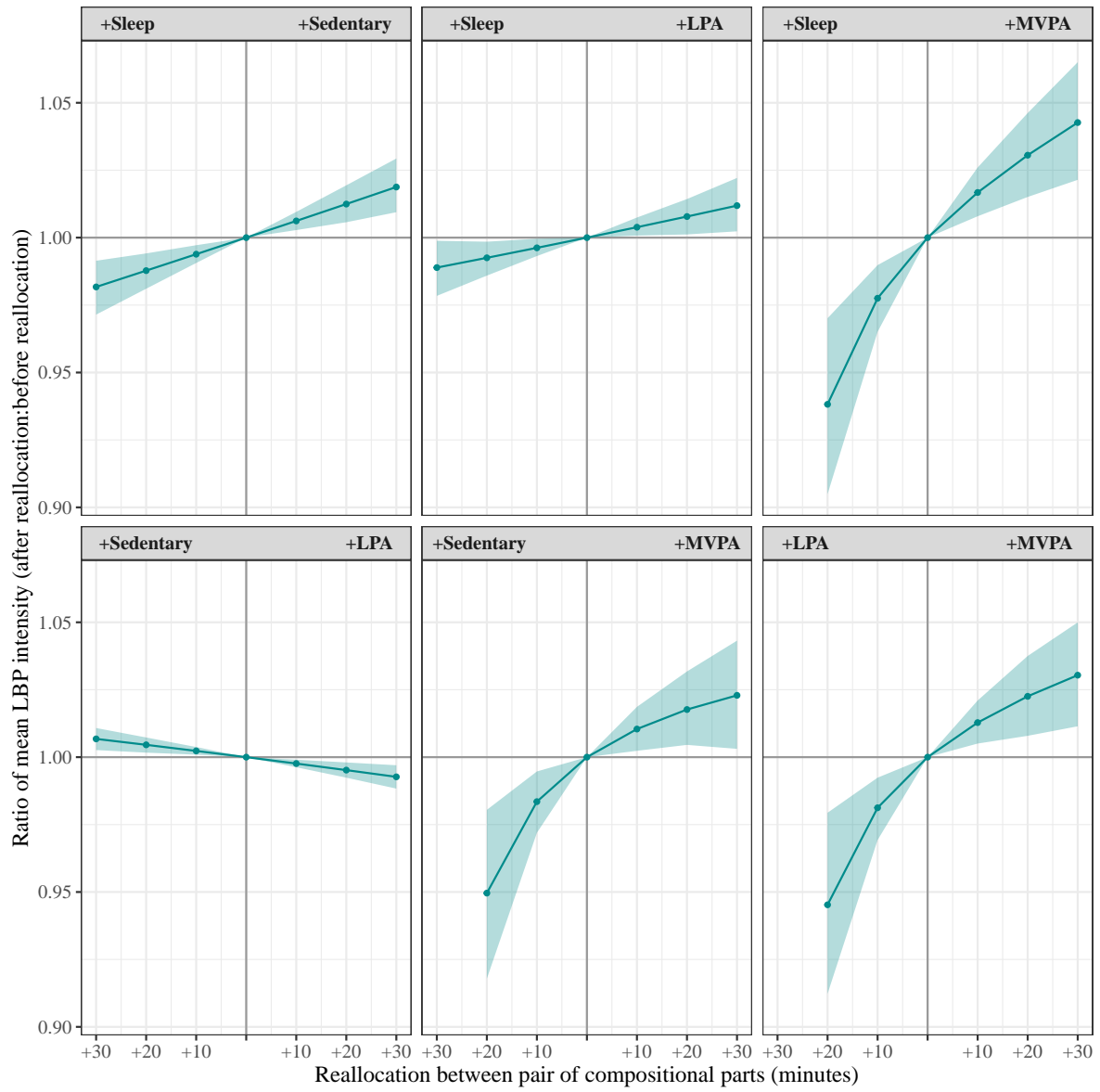
```

```
pd2$from_to <- factor(pd2$from_to, levels = unique(pd2$from_to))
```

```
this_breaks <- seq(-30, 30, 10)
this_labs <- sprintf("+%2.0f", abs(seq(-30, 30, 10)))
this_labs[this_labs == "+ 0"] <- ""
this_labs
```

```
[1] "+30" "+20" "+10" ""      "+10" "+20" "+30"
```

```
ggplot(pd2) +
  geom_vline(xintercept = 0, col = "grey60") +
  geom_hline(yintercept = 1, col = "grey60") +
  geom_ribbon(aes(x = change_time, ymin = ci_lo, ymax = ci_hi, fill = to), alpha = 0.3, col = "grey60") +
  geom_line(aes(x = change_time, y = outc_ratio, col = to), col = "cyan4") +
  geom_point(aes(x = change_time, y = outc_ratio, col = to), size = 1, col = "cyan4") +
  facet_wrap(~ from_to, labeller = label_bquote(.(from_to))) +
  theme_bw() +
  scale_x_continuous(breaks = this_breaks, labels = this_labs) +
  labs(
    x = paste0("Reallocation between pair of compositional parts (minutes)"),
    y = paste0("Ratio of mean LBP intensity (after reallocation:before reallocation)")
    # subtitle = "Note that odds ratios relate to the probability of having _increased_ fr
  ) +
  theme(
    legend.position = "none",
    text = element_text(family = "serif"),
    strip.text = element_text(size = 10, face = "bold"),
    axis.text = element_text(size = 10),
    axis.title = element_text(size = 12)
  )
```



```
ggsave(filename = "fig/lbp_intens_negbin_rat_v2.png", width = 14, height = 9, dpi = 600)
```

## 6 Session information

```
format(Sys.time(), '%d %b %Y')
```

```
[1] "26 Sep 2023"
```

```
sessionInfo()
```

```
R version 4.3.1 (2023-06-16 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19045)
```

```
Matrix products: default
```

```
locale:
```

```
[1] LC_COLLATE=English_Australia.utf8  LC_CTYPE=English_Australia.utf8
[3] LC_MONETARY=English_Australia.utf8 LC_NUMERIC=C
[5] LC_TIME=English_Australia.utf8
```

```
time zone: Australia/Darwin
tzcode source: internal
```

```
attached base packages:
```

```
[1] stats      graphics  grDevices  utils      datasets  methods    base
```

```
other attached packages:
```

```
[1] tictoc_1.2          boot_1.3-28.1      foreach_1.5.2
[4] knitr_1.43          car_3.1-2          carData_3.0-5
[7] mice_3.16.0         performance_0.10.4 zCompositions_1.4.0-1
[10] truncnorm_1.0-9     NADA_1.6-1.1      survival_3.5-5
[13] MASS_7.3-60         compositions_2.0-6 GGally_2.1.2
[16] ggplot2_3.4.2       forcats_1.0.0     readr_2.1.4
[19] tidyr_1.3.0         dplyr_1.1.2
```

```
loaded via a namespace (and not attached):
```

```
[1] tidyselect_1.2.0    viridisLite_0.4.2  farver_2.1.1      fastmap_1.1.1
[5] reshape_0.8.9      tensorA_0.36.2     bayestestR_0.13.1 digest_0.6.33
[9] rpart_4.1.19       lifecycle_1.0.3    magrittr_2.0.3    compiler_4.3.1
[13] rlang_1.1.1        tools_4.3.1        utf8_1.2.3        yaml_2.3.7
```

[17]	labeling_0.4.2	bit_4.0.5	plyr_1.8.8	RColorBrewer_1.1-3
[21]	abind_1.4-5	withr_2.5.0	purrr_1.0.1	datawizard_0.8.0
[25]	nnet_7.3-19	grid_4.3.1	fansi_1.0.4	jomo_2.7-6
[29]	colorspace_2.1-0	scales_1.2.1	iterators_1.0.14	insight_0.19.3
[33]	cli_3.6.1	rmarkdown_2.23	crayon_1.5.2	ragg_1.2.5
[37]	generics_0.1.3	rstudioapi_0.15.0	robustbase_0.99-0	tzdb_0.4.0
[41]	minqa_1.2.5	bayesm_3.1-5	splines_4.3.1	parallel_4.3.1
[45]	vctrs_0.6.3	glmnet_4.1-8	Matrix_1.6-0	jsonlite_1.8.7
[49]	patchwork_1.1.3	hms_1.1.3	ggrepel_0.9.3	bit64_4.0.5
[53]	mitml_0.4-5	systemfonts_1.0.4	see_0.8.0	glue_1.6.2
[57]	nloptr_2.0.3	DEoptimR_1.1-1	pan_1.9	codetools_0.2-19
[61]	shape_1.4.6	gtable_0.3.3	lme4_1.1-34	munsell_0.5.0
[65]	tibble_3.2.1	pillar_1.9.0	htmltools_0.5.5	R6_2.5.1
[69]	textshaping_0.3.6	vroom_1.6.3	evaluate_0.21	lattice_0.21-8
[73]	backports_1.4.1	broom_1.0.5	Rcpp_1.0.11	nlme_3.1-162
[77]	mgcv_1.8-42	xfun_0.39	pkgconfig_2.0.3	