Overview of the package LMMstar

Brice Ozenne

February 13, 2024

This vignette details how partial residuals can be used to illustrate model fit in a linear regression and a linear mixed model when using the package LMMstar. We thus start by loading the package:

```
library(LMMstar)
```

1 Univariate linear regression

To illustrate the use of partial residuals we will use the `state.x77` dataset:

```
df1 <- data.frame(lifeExp = state.x77[,4],
                  illiteracy = state.x77[,3],
                  income = state.x77[,2]/1000,
                  murder = state.x77[,5],
                  edu = cut(state.x77[,6],c(0,50,60,100)))
head(df1,4)
```

```
         lifeExp illiteracy income murder      edu
Alabama    69.05        2.1  3.624   15.1   (0,50]
Alaska     69.31        1.5  6.315   11.3 (60,100]
Arizona    70.55        1.8  4.530    7.8  (50,60]
Arkansas   70.66        1.9  3.378   10.1   (0,50]
```

which contains information about life expectancy (`lifeExp`), income (`income`), illeteracy (`illiteracy`), murder rate (`murder`), and the percentage of high-school graduates (as categorical variable) in various states in the USA. For later use we display a few descriptive for each covariate value:

```
summarize(lifeExp + illiteracy + income + murder + edu ~ 1, data = df1,
          columns = c("observed","missing","mean","min","max","sd"))
```

```
       outcome observed missing    mean    min    max        sd
1      lifeExp       50       0 70.8786 67.960 73.600 1.3423936
2   illiteracy       50       0  1.1700  0.500  2.800 0.6095331
3       income       50       0  4.4358  3.098  6.315 0.6144699
4       murder       50       0  7.3780  1.400 15.100 3.6915397
5  edu:(50,60]       50       0  0.5600  0.000  1.000 0.5014265
6 edu:(60,100]       50       0  0.1600  0.000  1.000 0.3703280
```

and check there are no missing values. Here for the categorical covariates the mean indicates the relative frequency of occurrence (56% and 16%) and other columns like `sd` should be ignored.

1.1 No interaction

Suppose we are interested in relating life expectancy (Y) to income (X). We cannot directly illustrate this relationship, as it could be confounded by other variables such as illiteracy (Z_1), murder rate (Z_2). and education (Z_3). We will therefore use a linear model to control for those variables ($\boldsymbol{Z} = (Z_1, Z_2, Z_3)$) where, for simplicity, we assume a linear effect for all variables:

$$Y = \alpha + \beta X + \gamma_1 Z_1 + \gamma_2 Z_2 + \gamma_2 Z_3 + \varepsilon$$

```
e.lm <- lmm(lifeExp ~ income + illiteracy + murder + edu, data = df1)
model.tables(e.lm)
```

```
              estimate         se      df      lower      upper      p.value
(Intercept) 71.5782755 1.13848841 44.0088 69.2838158 73.8727352 0.000000e+00
income       0.1926977 0.25264925 44.0088 -0.3164805  0.7018759 4.497063e-01
illiteracy   0.1759019 0.32096701 44.0088 -0.4709610  0.8227647 5.864356e-01
murder      -0.2782232 0.04785536 44.0088 -0.3746688 -0.1817776 6.328010e-07
edu(50,60]   0.3014130 0.41401530 44.0088 -0.5329753  1.1358013 4.704551e-01
edu(60,100]  0.7730650 0.51311796 44.0088 -0.2610505  1.8071804 1.390575e-01
```

Note that the estimates are nearly identical to the ones of the `lm` function:

```
coef(e.lm) - coef(lm(lifeExp ~ income + illiteracy + murder + edu, data = df1))
```

```
  (Intercept)        income    illiteracy        murder    edu(50,60]   edu(60,100]
-2.700062e-13  2.033929e-13 -2.898792e-13  6.272760e-15 -4.223843e-13 -5.306866e-13
```

A graphical display can now be obtained by modifying the original outcome, life expectancy, had every state had the same Illiteracy and Murder:

```
df1$pres <- residuals(e.lm, type = "partial", var = c("(Intercept)","income"))
head(df1)
```

```
           lifeExp illiteracy income murder      edu     pres
Alabama      69.05        2.1  3.624   15.1   (0,50] 72.88178
Alaska       69.31        1.5  6.315   11.3 (60,100] 71.41700
Arizona      70.55        1.8  4.530    7.8  (50,60] 72.10210
Arkansas     70.66        1.9  3.378   10.1   (0,50] 73.13584
California   71.71        1.1  5.114   10.3 (60,100] 73.60914
Colorado     72.06        0.7  4.884    6.8 (60,100] 73.05572
```

By default, the partial residuals are computed substracting the effect of the covariates, i.e. had each state got no illiteracy, no murder, and the lowest education level:

```
c(69.05 - 0.17590 * 2.1 - (-0.27822) * 15.1,
  69.31 - 0.17590 * 1.5 - (-0.27822) * 11.3 - 0.77306)
```

```
[1] 72.88173 71.41698
```

This is why the element `"(Intercept)"` is specified in the `var` argument: to avoid to substract the mean value from the outcome and keep a plausible range of values for the outcome.

One may wish to compute the life expectancy had every state got a specific illiteracy, murder rate, and education. Say the most common in the sample: 1.17, 7.378, and (50,60] based on the descriptive statistics. This can be obtained by specifying an attribute to the argument type:

```
type <- "partial"
attr(type, "reference") <- data.frame(illiteracy = 1.17, murder = 7.378,
                                      edu = factor("(50,60]",levels(df1$edu)))
df1$pres2 <- residuals(e.lm, type = type, var = c("(Intercept)","income"))
head(df1)
```

```
           lifeExp illiteracy income murder      edu     pres    pres2
Alabama      69.05        2.1  3.624   15.1    (0,50] 72.88178 71.33626
Alaska       69.31        1.5  6.315   11.3 (60,100] 71.41700 69.87149
Arizona      70.55        1.8  4.530    7.8  (50,60] 72.10210 70.55659
Arkansas     70.66        1.9  3.378   10.1    (0,50] 73.13584 71.59033
California   71.71        1.1  5.114   10.3 (60,100] 73.60914 72.06363
Colorado     72.06        0.7  4.884    6.8 (60,100] 73.05572 71.51021
```

or computing by hand accounting for the reference level:

```
c(69.05 - 0.17590 * (2.1-1.170) - (-0.27822) * (15.1-7.378) + 0.30141,
  69.31 - 0.17590 * (1.5-1.170) - (-0.27822) * (11.3-7.378) + 0.30141 - 0.77306)
```

```
[1] 71.33624 69.87148
```

Note that changing the reference level only shift the partial residual value by a constant, here:

```
unique(df1$pres2 - df1$pres)
```

```
[1] -1.545513
```

so does not affect the relation between the outcome (here `LifeExp`) and the exposure of interest (here `Income`). One can then get a graphical display either manually using ggplot:

```
gg.pres <- ggplot(df1) + geom_point(aes(x=income, y=pres))
gg.pres <- gg.pres + geom_abline(intercept = coef(e.lm)["(Intercept)"],
                                 slope = coef(e.lm)["income"])
gg.pres <- gg.pres + ggtitle("(B) partial residuals")
gg.pres
```

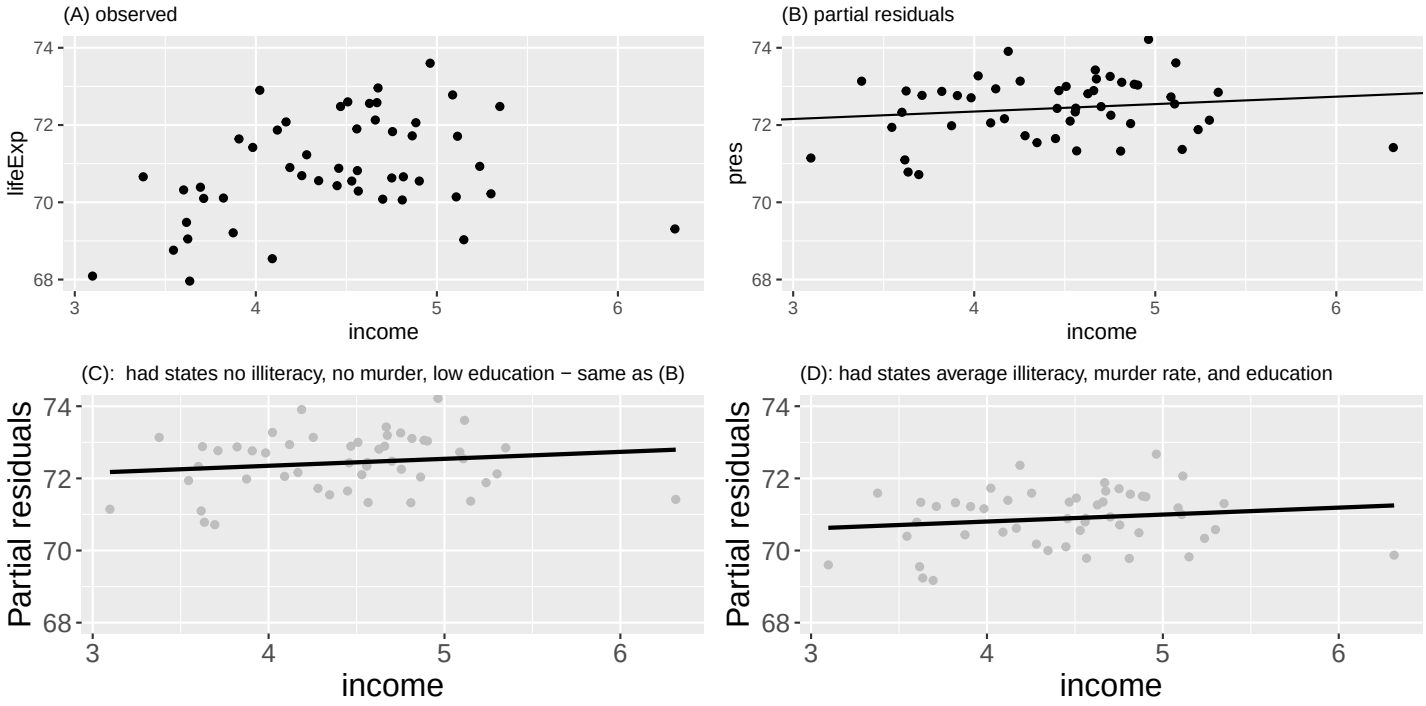
or directly via the plot function:

```
plot(e.lm, time = "income", type = "partial", var = c("(Intercept)","income")) # C
plot(e.lm, time = "income", type = type, var = c("(Intercept)","income")) # D
```

These can be compared to displaying the observed outcome vs. income:

```
gg.obs <- ggplot(df1) + geom_point(aes(x=income, y=lifeExp))
gg.obs <- gg.obs + ggtitle("(A) observed")
gg.obs
```

where it is apparent that by using the partial residuals, the data has been normalized and exhibit less variability.



The output of the `plot` method is a list containing an element plot with the ggplot object and an element data with the dataset. To avoid actually displaying the graph one can use `autoplot` to only output the result:

```
ls.plot <- autoplot(e.lm, time = "income", type = "partial", var = c("(Intercept)","income"))
str(ls.plot$data)
class(ls.plot$plot)
```

```
Classes 'residuals_lmm' and 'data.frame':       50 obs. of  3 variables:
 $ income   : num  3.62 6.32 4.53 3.38 5.11 ...
 $ fitted   : num  72.3 72.8 72.5 72.2 72.6 ...
 $ r.partial: num  72.9 71.4 72.1 73.1 73.6 ...
[1] "gg"     "ggplot"
```

One can re-create the plot based on the data argument or modify the existing plot based on the ggplot object, e.g. display with the y axis between 68 and 74:

```
ls.plot$plot  + coord_cartesian(ylim=c(68,74))
```

1.2 What about confidence intervals?

A common question is whether one can display confidence intervals for the regression line. It is possible to add confidence intervals on the plot either via the argument `ci.alpha`:

```
plot(e.lm, time = "income", type = "partial", var = c("(Intercept)","income"),
     ci.alpha = 0.25)
```

or by requesting confidence intervals for the fitted lines via the argument `pres.ci` when calling `residuals`:

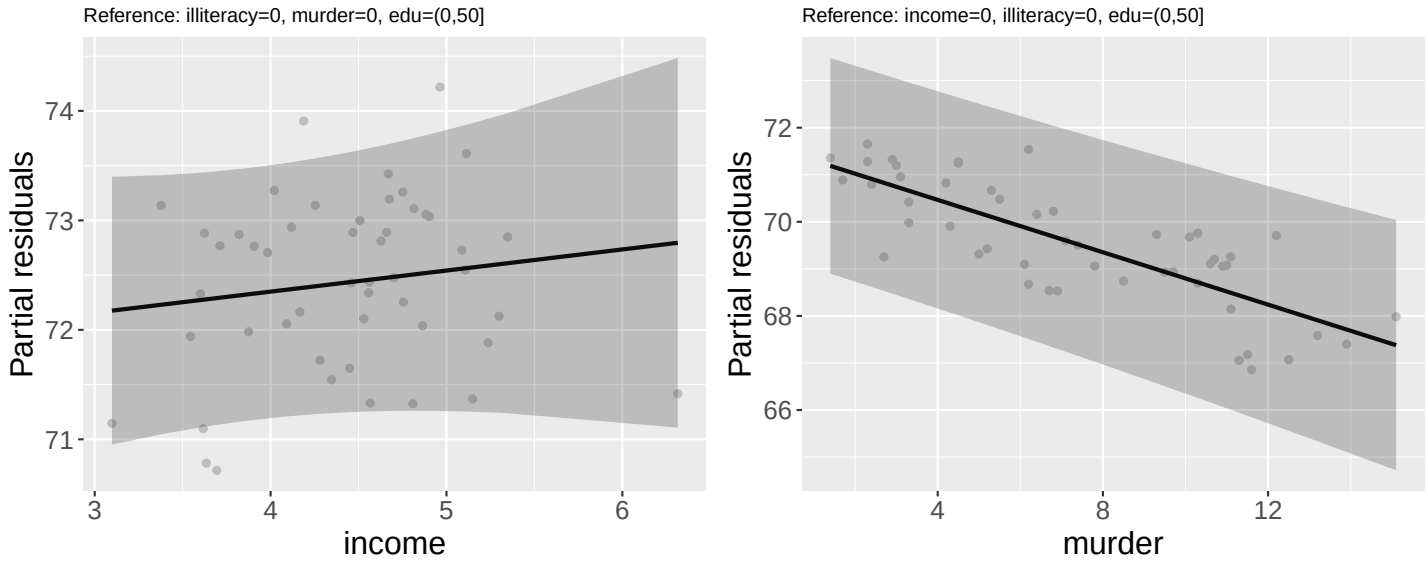
```
pres.ci <- residuals(e.lm, type = "partial", var = c("(Intercept)","income"),
                     keep.data = TRUE, fitted.ci = TRUE)
head(pres.ci)
```

```
  lifeExp illiteracy income murder    edu   fitted fitted.lower fitted.upper r.partial
1   69.05          0  3.624      0 (0,50] 72.27661     71.11458     73.43864  72.88178
2   69.31          0  6.315      0 (0,50] 72.79516     71.10708     74.48324  71.41700
3   70.55          0  4.530      0 (0,50] 72.45120     71.25294     73.64945  72.10210
4   70.66          0  3.378      0 (0,50] 72.22921     71.04575     73.41266  73.13584
5   71.71          0  5.114      0 (0,50] 72.56373     71.25357     73.87389  73.60914
6   72.06          0  4.884      0 (0,50] 72.51941     71.26050     73.77832  73.05572
```

which can be added to the previous graphical display, e.g.:

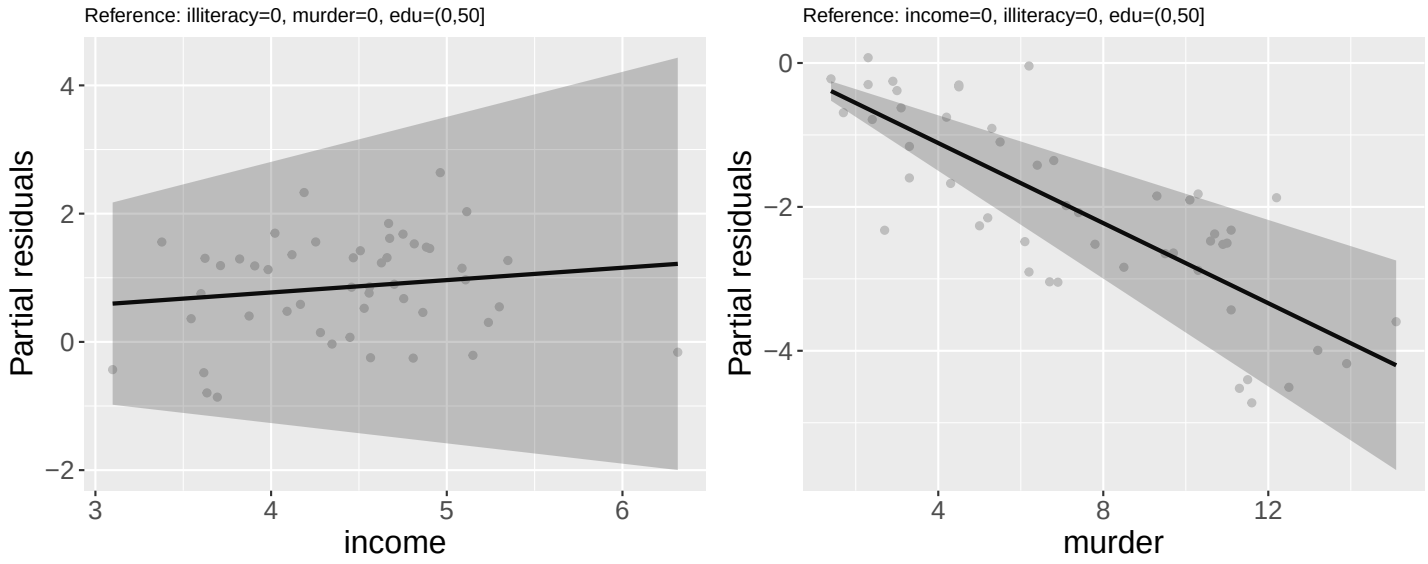
```
gg.pres + geom_ribbon(data = pres.ci, alpha = 0.1,
                      aes(ymin = fitted.lower, ymax = fitted.upper, x = income))
```

The first plot is displayed in the left panel of the figure below. A similar partial residual plot but now for the `murder` variable is displayed in the right panel.



In many case the uncertainty represented here is of little interest, since it is the uncertainty of the intercept plus the exposure effect. This is why even though the `murder` variable was highly significant ($p<0.001$) whereas the income variable was not significant ($p=0.45$) the confidence intervals looks large in both cases. To only capture the uncertainty relative to the `income` or `murder` variable one should remove the intercept value, e.g. by omitting `"(Intercept)"` from the `var` argument:

```
plot(e.lm, time = "income", type = "partial", var = "income", ci.alpha = 0.25)
plot(e.lm, time = "murder", type = "partial", var = "murder", ci.alpha = 0.25)
```



The unpleasant side effect is that the range of values on the y-axis appears unrealistic now. The statistical uncertainty may therefore be better communicated otherwise, e.g. reporting confidence intervals or p-values related to the covariate effect and keeping the partial residual plot free of confidence intervals.

1.3 Interaction with a categorical variable

Suppose that we are now interested in relating life expectancy (Y) to both income (X_1) for various level of education ($X_2 \in \{a, b, c\}$), adjusting for other variables such as illiteracy (Z_1) and murder rate (Z_2). As before we assume a linear effect for all variables:

$$Y = \alpha + \beta_{1a} X_1 \mathbb{1}_{X_2=a} + \beta_{1b} X_1 \mathbb{1}_{X_2=b} + \beta_{1c} X_1 \mathbb{1}_{X_2=c} + \gamma_1 Z_1 + \gamma_2 Z_2 + \varepsilon$$

where $\mathbb{1}_x$ denotes the indicator variable taking value 1 when x is true and 0 otherwise. This model can be estimated with the following R code

```
e.lmI <- lmm(lifeExp ~ income:edu + illiteracy + murder, data = df1)
model.tables(e.lmI)
```

	estimate	se	df	lower	upper	p.value
(Intercept)	71.7858373	1.20951681	44.0088	69.3482301	74.2234444	0.000000e+00
illiteracy	0.1286978	0.31914517	44.0088	-0.5144934	0.7718890	6.887110e-01
murder	-0.2794017	0.04820845	44.0088	-0.3765589	-0.1822445	6.727632e-07
income:edu(0,50]	0.1714686	0.29772543	44.0088	-0.4285542	0.7714914	5.675972e-01
income:edu(50,60]	0.2252558	0.25210982	44.0088	-0.2828353	0.7333469	3.764587e-01
income:edu(60,100]	0.3037682	0.23692879	44.0088	-0.1737277	0.7812641	2.065179e-01

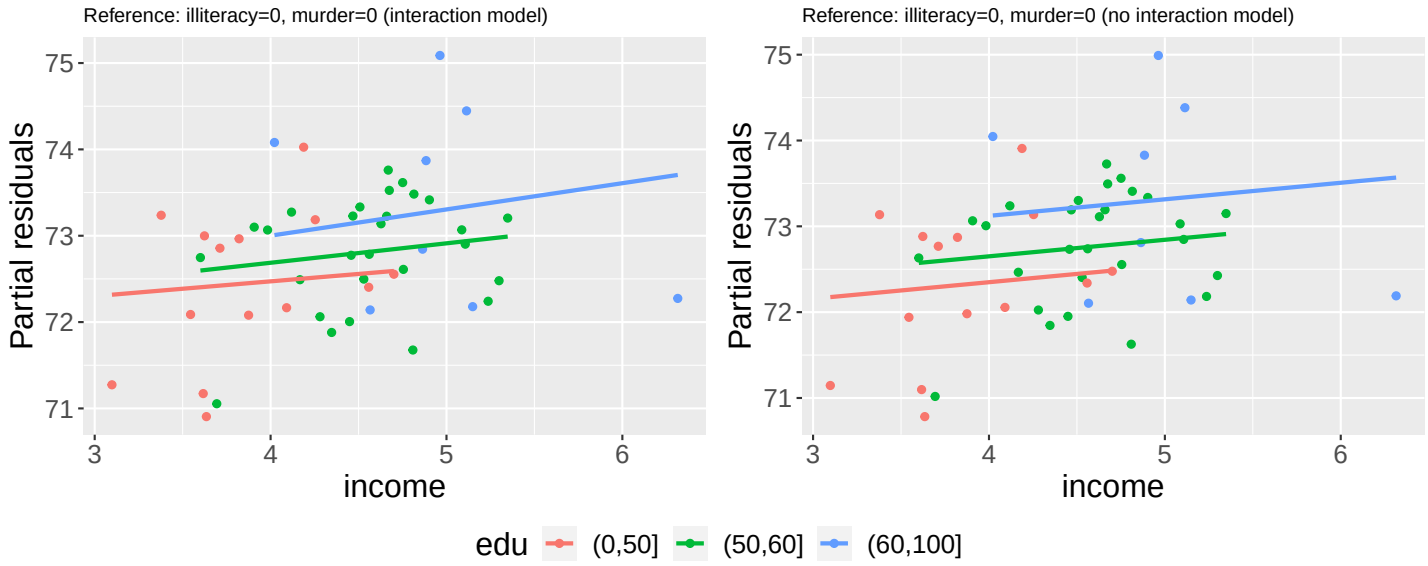
Note: this model is the same as `lmm(lifeExp ~ income*edu + illiteracy + murder, data = df1)` but uses a different parametrisation.

Similarly as before, we can use the `plot` function to display the partial residuals with respect to both `income` and `edu`:

```
plot(e.lmI, time = "income", type = "partial", var = c("(Intercept)","income","edu"))
```

which can be compared to a plot assuming no interaction:

```
plot(e.lm, time = "income", type = "partial", var = c("(Intercept)","income","edu"))
```



The partial residuals can also be output via the `residuals` method:

```
residuals(e.lmI, type = "partial", var = c("(Intercept)","income","edu"))[1:5]
```

```
[1] 72.99870 72.27419 72.49768 73.23743 74.44627
```

and one can check that they are evaluated by substracting the effect of the other variables (here `illiteracy` and `murder`), e.g.:

```
c(69.05 - 0.12870 * 2.1 - (-0.27940) * 15.1,
  69.31 - 0.12870 * 1.5 - (-0.27940) * 11.3)
```

```
[1] 72.99867 72.27417
```

Here we computed partial residuals representing the life expectancy in the states had there be no murder nor illiteracy. We could also consider the case of average murder rate and illiteracy. We would need to define a new reference (i.e. drop `edu`):

```
typeI <- "partial"
attr(typeI, "reference") <- data.frame(illiteracy = 1.17, murder = 7.378)
residuals(e.lmI, type = typeI, var = c("(Intercept)","income"))[1:5]
```

```
[1] 71.08785 70.36334 70.58683 71.32658 72.53542
```

which we can also retrieve by hand:

```
c(69.05 - 0.12870 * (2.1-1.170) - (-0.27940) * (15.1-7.378),
  69.31 - 0.12870 * (1.5-1.170) - (-0.27940) * (11.3-7.378))
```

```
[1] 71.08784 70.36334
```

2 Linear mixed model

To illustrate the use of partial residuals we will use data from a two-arm randomized trial comparing the quality of the vision over time of patients under placebo vs. active drug. We first re-shape the data:

```
data(armd.wide, package = "nlmeU")
library(reshape2)
armd.long <- reshape2::melt(armd.wide,
                            measure.vars = paste0("visual",c(0,4,12,24,52)),
                            id.var = c("subject","lesion","treat.f","miss.pat"),
                            variable.name = "week",
                            value.name = "visual")
armd.long$week <- factor(armd.long$week,
                         level = paste0("visual",c(0,4,12,24,52)),
                         labels = c(0,4,12,24,52))
```

make sure that

3 R session

Details of the R session used to generate this document:

```
sessionInfo()
```

```
R version 4.2.0 (2022-04-22 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19045)

Matrix products: default

locale:
[1] LC_COLLATE=Danish_Denmark.utf8  LC_CTYPE=Danish_Denmark.utf8    LC_MONETARY=Danish_Denmark.utf8
[4] LC_NUMERIC=C                    LC_TIME=Danish_Denmark.utf8

attached base packages:
[1] parallel  grid      stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] mice_3.14.0          sandwich_3.0-2       scales_1.2.1         rlang_1.1.1
 [5] pbapply_1.7-0        numDeriv_2016.8-1.1  nlme_3.1-158         lava_1.7.2.1
 [9] doSNOW_1.0.20        snow_0.4-4           iterators_1.0.14     foreach_1.5.2
[13] copula_1.1-2         lme4_1.1-29          Matrix_1.5-1         LMMstar_1.0.0
[17] ggpubr_0.4.0         multcomp_1.4-22      TH.data_1.1-1        MASS_7.3-57
[21] survival_3.3-1       mvtnorm_1.2-3        qqtest_1.2.0         emmeans_1.8.8-090002
[25] ggplot2_3.4.3

loaded via a namespace (and not attached):
 [1] butils.base_1.2     minqa_1.2.4         colorspace_2.1-0    ggsignif_0.6.3
```

```
 [5] ellipsis_0.3.2      estimability_1.4.1  parameters_0.18.2   fs_1.6.3
 [9] listenv_0.9.0       farver_2.1.1        remotes_2.4.2       gsl_2.1-8
[13] fansi_1.0.4         codetools_0.2-18    splines_4.2.0       doParallel_1.0.17
[17] cachem_1.0.8        pkgload_1.3.0       nloptr_2.0.3        broom_0.8.0
[21] stabledist_0.7-1    effectsize_0.7.0.5  shiny_1.7.2         compiler_4.2.0
[25] backports_1.4.1     fastmap_1.1.1       cli_3.6.1           later_1.3.0
[29] htmltools_0.5.6     prettyunits_1.1.1   tools_4.2.0         lmerTest_3.1-3
[33] coda_0.19-4         gtable_0.3.4        glue_1.6.2          reshape2_1.4.4
[37] dplyr_1.1.3         Rcpp_1.0.11         carData_3.0-5       vctrs_0.6.3
[41] insight_0.18.4      stringr_1.5.0       globals_0.16.2      ps_1.7.1
[45] mime_0.12           miniUI_0.1.1.1      lifecycle_1.0.3     devtools_2.4.4
[49] rstatix_0.7.0       future_1.31.0       zoo_1.8-11          promises_1.2.0.1
[53] memoise_2.0.1       gridExtra_2.3       stringi_1.7.12      bayestestR_0.13.0
[57] pcaPP_2.0-3         boot_1.3-28         pkgbuild_1.3.1      pkgconfig_2.0.3
[61] lattice_0.20-45     purrr_1.0.2         htmlwidgets_1.6.2   labeling_0.4.3
[65] cowplot_1.1.1       tidyselect_1.2.0    processx_3.6.1      parallelly_1.34.0
[69] plyr_1.8.7          magrittr_2.0.3      R6_2.5.1            generics_0.1.3
[73] profvis_0.3.7       ADGofTest_0.3       pillar_1.9.0        withr_2.5.1
[77] mgcv_1.8-40         datawizard_0.6.1    abind_1.4-5         pspline_1.0-19
[81] tibble_3.2.1        future.apply_1.10.0 crayon_1.5.1        car_3.1-0
[85] utf8_1.2.3          urlchecker_1.0.1    usethis_2.1.6       data.table_1.14.2
[89] callr_3.7.2         digest_0.6.33       xtable_1.8-4        tidyr_1.3.0
[93] httpuv_1.6.5        stats4_4.2.0        munsell_0.5.0       sessioninfo_1.2.2
```

References