Bias correction in GLMs

Bicko, Jonathan & Ben

2021 Jun 21 (Mon)

Introduction

We intend to investigate our prediction based on known truth and any bias potentially introduced by non-linear averaging, conditioning or random effect. We'll start with a simple case of a only fixed effect model and then consider a mixed effect model.

Simulation

We perform a simple simulation for a fixed effect model

$$
\begin{aligned}
\text{logit}(\text{status} = 1) &= \eta \\
\eta &= \beta_0 + \beta_\text{A}\text{Age} + \beta_\text{W}\text{Wealthindex} \\
\text{Age} &\sim \text{Uniform}(0.2, 1) \\
\text{Wealthindex} &\sim \text{Normal}(0, 1) \\
\beta_0 &= 0.7 \\
\beta_\text{A} &= 0.3 \\
\beta_\text{W} &= 0.6
\end{aligned}
$$

```
N <- 1e5
beta0 <- 0.7
betaA <- 0.2
betaW <- 0.5

age_max <- 1
age_min <- 0.2
age <- runif(N, age_min, age_max)

wealthindex <- rnorm(N, 0, 1)

eta <- beta0 + betaA * age + betaW * wealthindex
sim_df <- (data.frame(age=age, wealthindex=wealthindex, eta=eta)
    %>% mutate(status = rbinom(N, 1, plogis(eta)))
    %>% select(-eta)
)
true_prop <- mean(sim_df$status)
print(true_prop)
```

```
## [1] 0.68391
```

```
head(sim_df)
```

```
##         age wealthindex status
## 1 0.8452918  1.42167552      0
## 2 0.2563395 -0.09636578      1
## 3 0.4192913 -0.16746300      0
## 4 0.7882493 -1.69799152      0
## 5 0.3893671 -0.60398779      1
## 6 0.8806260 -0.97554943      1
```

Simple logistic model

```
simple_mod <- glm(status ~ age + wealthindex, data = sim_df, family="binomial")
```

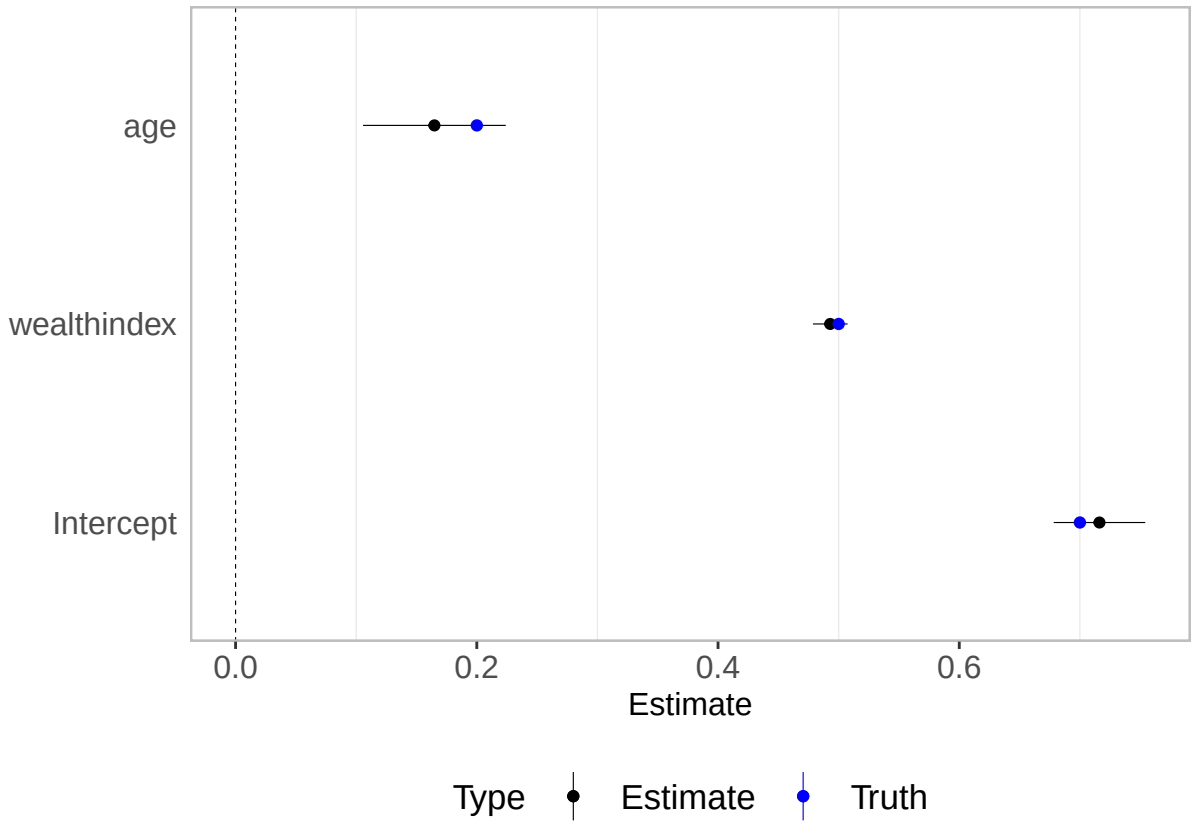
Coefficient plots

```
## True beta
true_beta_df <- data.frame(term=c("Intercept", "age", "wealthindex")
    , estimate=c(beta0, betaA, betaW)
)

## Tidy coef estimates
coef_df <- (broom::tidy(simple_mod, conf.int=TRUE)
#   %>% dotwhisker::by_2sd(sim_df)
    %>% mutate(term = gsub("\\(|\\)", "", term))
)
print(coef_df)
```

```
## # A tibble: 3 x 7
##   term        estimate std.error statistic   p.value conf.low conf.high
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>    <dbl>     <dbl>
## 1 Intercept      0.716   0.0194      37.0  1.29e-299    0.678     0.754
## 2 age            0.165   0.0302       5.46 4.72e-  8    0.106     0.224
## 3 wealthindex    0.493   0.00733     67.3  0.           0.479     0.507
```

```
simple_coef_plot <- (plotEsize(coef_df)
    + geom_point(data=true_beta_df, aes(x=term, y=estimate, colour="Truth"))
    + labs(colour="Type")
    + scale_colour_manual(values=c("black", "blue"))
)
print(simple_coef_plot)
```



Variable effect plots – varpred and population averaging approach

```
# Age
## varpred way
simple_vareff_age <- varpred(simple_mod, "age", isolate=FALSE, modelname="varpred")

# Wealth index
## Not bias adjusted
simple_vareff_wealthindex <- varpred(simple_mod, "wealthindex", isolate=TRUE, modelname="varpred")

## Bias adjusted
simple_vareff_wealthindex_adjust <- varpred(simple_mod, "wealthindex", isolate=TRUE, pop.ave=TRUE, model

vareff_wealthindex <- simple_vareff_wealthindex
vareff_wealthindex$preds <- do.call("rbind", list(vareff_wealthindex$preds, simple_vareff_wealthindex_ad
wealthindex_plot <- (plot(vareff_wealthindex)
    + labs(y="", colour="Model")
    + geom_hline(yintercept=true_prop, lty=2, colour="grey")
    + scale_colour_manual(values=c("black", "blue"))
    + theme(legend.position="bottom")
)
```

```
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
```

Effect sizes on logit scale:

```
coef_df_logit <- (coef_df
    %>% select(term, estimate, conf.low, conf.high)
```

```
    %>% group_by(term)
    %>% summarise_all(plogis)
)
print(coef_df_logit)
```

```
## # A tibble: 3 x 4
##   term        estimate conf.low conf.high
##   <chr>          <dbl>    <dbl>     <dbl>
## 1 age            0.541    0.526     0.556
## 2 Intercept      0.672    0.663     0.680
## 3 wealthindex    0.621    0.617     0.624
```

Population averaging

- Averages of the entire population of the non-focal predictor

```
popavefun <- function(mod, focal, non.focal, level=0.95, modelname="Pop. ave", ...) {
    mf <- model.matrix(mod)
    mm <- (mf
        %>% data.frame()
        %>% mutate_at(non.focal, mean)
        %>% as.matrix()
    )
    vc <- vcov(mod)
    linpred <- as.vector(mm %*% coef(mod))
    pse_var <- sqrt(rowSums(mm * t(tcrossprod(data.matrix(vc), mm))))
    z.val <- qnorm(1 - (1 - level)/2)
    pred_df <- (mf
        %>% data.frame()
        %>% select_at(focal)
        %>% mutate(fit = linpred
            , lwr = plogis(fit - z.val*pse_var)
            , upr = plogis(fit + z.val*pse_var)
            , fit = plogis(fit)
            , model = modelname
            , se = NA
        )
    )
    return(pred_df)
}

# simple_vareff_age_pop <- varpred(simple_mod, "age", isolate=TRUE, modelname="Pop. ave")
simple_vareff_age_pop <- popavefun(simple_mod, "age", "wealthindex", modelname = "Pop. ave")

vareff_age <- simple_vareff_age
vareff_age$preds <- do.call("rbind", list(vareff_age$preds, simple_vareff_age_pop))
age_plot <- (plot(vareff_age)
    + labs(y="Prob. of improved \n service", colour="Model")
    + geom_hline(yintercept=true_prop, lty=2, colour="grey")
    + scale_colour_manual(values=c("black", "blue"))
    + theme(legend.position="bottom")
)
```

```
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
```

```
ggarrange(age_plot, wealthindex_plot, common.legend=TRUE)
```

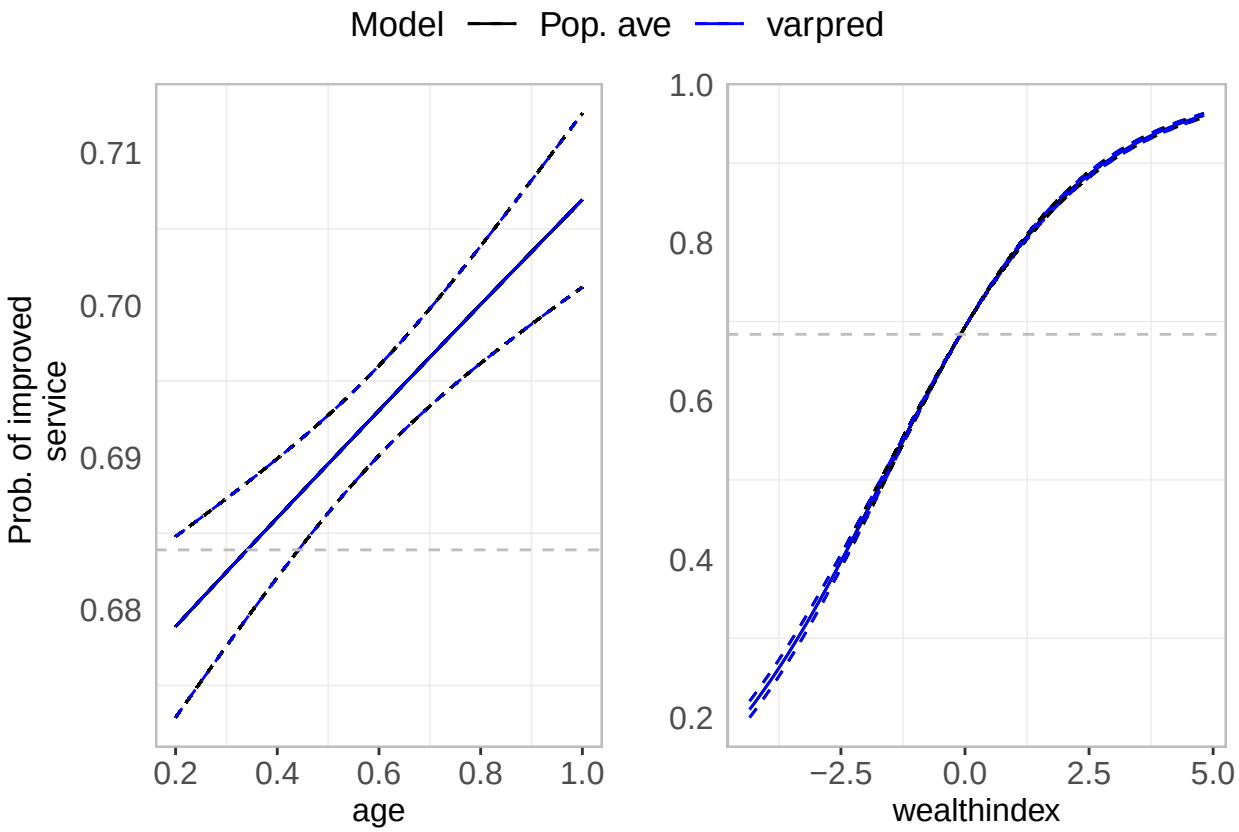


Figure 1: A comparison of population averaged and varpred-based predictions. For `age`, we implement the naive approach to compute the predictions in `popavefun` function and then implement the same in `vareffects` so as to use the centering machineries. In both cases, the population averaging and varpred gives similar estimates. The estimated population average is very close to the observed in the case of `wealthindex` but slightly higher in the case of `age`, see previous paragraph.

The observed population average is 0.68391 while the estimated population averages are:

- age: 0.6930223
- wealthindex: 0.6839543

Random effect model

```
# Simulation parameters
nHH <- 1000     # Number of HH (primary units) per year

nyrs <- 10  # Number of years to simulate
yrs <- 2000 + c(1:nyrs) # Years to simulate
N <- nyrs * nHH

## HH random effect sd
hhSD <- 0.5

# Generate dataset template
```

```
temp_df <- (data.frame(hhid = rep(c(1:nHH), each = nyrs)
        , years = rep(yrs, nHH)
        , age = runif(n=N, age_min, age_max)
        , wealthindex = rnorm(n = N, 0, 1)
    )
)

# Simulate HH-level random effects (residual error)
hhRE <- rnorm(nHH, hhSD)
temp_df$hhRE <- hhRE[temp_df$hhid]

sim_df <- (temp_df
    %>% mutate(eta = beta0 + betaA * age + betaW * wealthindex + hhRE
        , status = rbinom(N, 1, plogis(eta))
    )
    %>% select(-eta)
)
true_prop_reff <- mean(sim_df$status)
print(true_prop)
```

```
## [1] 0.68391
```

```
print(head(sim_df, 50))
```

```
##    hhid years       age  wealthindex       hhRE status
## 1     1  2001 0.7839212 -1.692179648  0.4206595      0
## 2     1  2002 0.9037890 -0.263030055  0.4206595      1
## 3     1  2003 0.8576889  0.002815092  0.4206595      0
## 4     1  2004 0.8144230 -0.675395593  0.4206595      1
## 5     1  2005 0.7186861 -1.538036772  0.4206595      1
## 6     1  2006 0.7597485  0.391245229  0.4206595      1
## 7     1  2007 0.6149017  2.056054031  0.4206595      1
## 8     1  2008 0.3254058 -1.450445719  0.4206595      0
## 9     1  2009 0.4590421  1.032356344  0.4206595      1
## 10    1  2010 0.9655930  0.078181544  0.4206595      1
## 11    2  2001 0.5771032 -1.330230239 -0.5219804      0
## 12    2  2002 0.7350279 -0.998451556 -0.5219804      0
## 13    2  2003 0.7330121 -1.023386445 -0.5219804      0
## 14    2  2004 0.5971016  1.088433607 -0.5219804      1
## 15    2  2005 0.4273264  0.085035379 -0.5219804      1
## 16    2  2006 0.7700529 -0.343819595 -0.5219804      1
## 17    2  2007 0.6748961 -0.617258698 -0.5219804      0
## 18    2  2008 0.9937384  0.019688926 -0.5219804      1
## 19    2  2009 0.7161956 -0.479984118 -0.5219804      1
## 20    2  2010 0.9087623 -1.330193938 -0.5219804      1
## 21    3  2001 0.7843109  1.024680339  1.6599535      1
## 22    3  2002 0.4214186 -0.115317909  1.6599535      1
## 23    3  2003 0.4690896 -0.019207839  1.6599535      1
## 24    3  2004 0.9826783 -0.331310319  1.6599535      1
## 25    3  2005 0.5202461  0.017149152  1.6599535      1
## 26    3  2006 0.5706091  0.126947664  1.6599535      1
## 27    3  2007 0.9412412  0.091350105  1.6599535      1
## 28    3  2008 0.4746049  1.464210259  1.6599535      1
## 29    3  2009 0.8198836  0.362756720  1.6599535      1
```

```
## 30    3  2010 0.8814067  0.972633642  1.6599535      1
## 31    4  2001 0.9003148 -1.579999835  1.0509941      0
## 32    4  2002 0.8791843 -0.901930197  1.0509941      1
## 33    4  2003 0.4857555  0.810361728  1.0509941      0
## 34    4  2004 0.4698130  1.791758971  1.0509941      1
## 35    4  2005 0.4790621 -1.400571271  1.0509941      1
## 36    4  2006 0.7373146  0.460122839  1.0509941      1
## 37    4  2007 0.6444353 -1.540631777  1.0509941      0
## 38    4  2008 0.2567840  1.467108337  1.0509941      1
## 39    4  2009 0.8160949  1.803468787  1.0509941      1
## 40    4  2010 0.7061629 -0.110921618  1.0509941      1
## 41    5  2001 0.7901509  0.871505733 -0.9855537      1
## 42    5  2002 0.9402003 -1.249436502 -0.9855537      0
## 43    5  2003 0.3245155 -0.163727437 -0.9855537      1
## 44    5  2004 0.8139511  0.454186636 -0.9855537      0
## 45    5  2005 0.8332164 -0.103368956 -0.9855537      0
## 46    5  2006 0.4175290 -1.297319707 -0.9855537      1
## 47    5  2007 0.4800863  0.966208362 -0.9855537      1
## 48    5  2008 0.8874108 -0.009493285 -0.9855537      0
## 49    5  2009 0.4135354 -1.115040401 -0.9855537      1
## 50    5  2010 0.8308159 -0.078702741 -0.9855537      0
```

Fit model

```
reff_mod <- glmmTMB(status ~ age + wealthindex + (1|hhid)
    , data = sim_df
    , family = binomial(link = "logit")
)
```

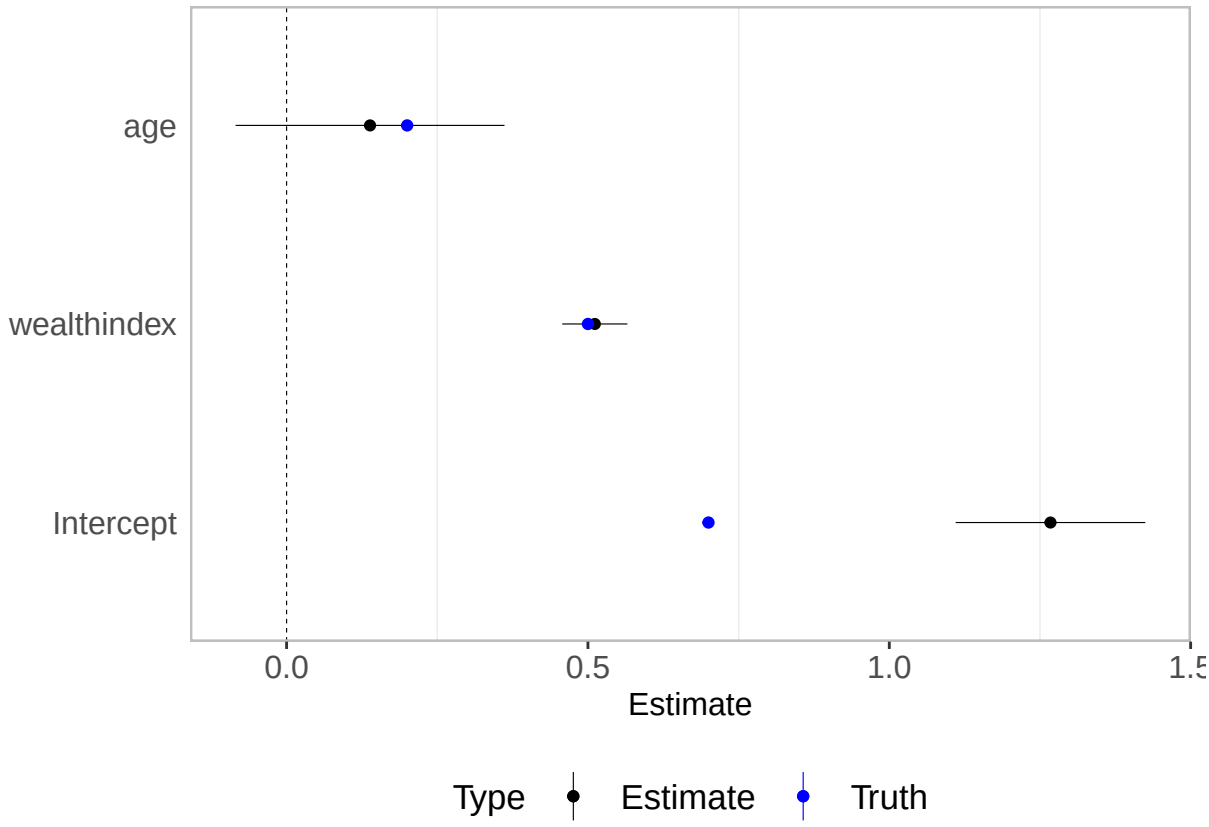
```
## Tidy coef estimates
reff_coef_df <- (broom.mixed::tidy(reff_mod, conf.int=TRUE)
    %>% mutate(term = gsub("\\(|\\)", "", term))
    %>% filter(effect=="fixed")
)
```

```
## Registered S3 method overwritten by 'broom.mixed':
##   method      from
##   tidy.gamlss broom
```

```
print(reff_coef_df)
```

```
## # A tibble: 3 x 10
##   effect component group term     estimate std.error statistic  p.value conf.low
##   <chr>  <chr>     <chr> <chr>       <dbl>     <dbl>     <dbl>    <dbl>    <dbl>
## 1 fixed  cond      <NA>  Interce~    1.27     0.0803     15.8  3.53e-56   1.11
## 2 fixed  cond      <NA>  age         0.138    0.114       1.22 2.24e- 1  -0.0846
## 3 fixed  cond      <NA>  wealthi~    0.511    0.0276     18.5  1.03e-76   0.457
## # ... with 1 more variable: conf.high <dbl>
```

```
reff_coef_plot <- (plotEsize(reff_coef_df)
    + geom_point(data=true_beta_df, aes(x=term, y=estimate, colour="Truth"))
    + labs(colour="Type")
    + scale_colour_manual(values=c("black", "blue"))
)
print(reff_coef_plot)
```



Variable effect plots

- Age

```
## varpred way
reff_vareff_age <- varpred(reff_mod, "age", isolate=FALSE, modelname="varpred")

## Pop. average
reff_vareff_age_pop <- varpred(reff_mod, "age", isolate=TRUE, pop.ave=TRUE, modelname = "Pop. ave")

## Bias adjust
reff_vareff_age_adjust <- varpred(reff_mod, "age", isolate=TRUE, bias.adjust=TRUE, modelname = "Bias ad

vareff_age <- reff_vareff_age
vareff_age$preds <- do.call("rbind", list(vareff_age$preds, reff_vareff_age_pop$preds, reff_vareff_age_
age_plot <- (plot(vareff_age)
    + labs(y="Prob. of improved \n service", colour="Model")
    + geom_hline(yintercept=true_prop_reff, lty=2, colour="grey")
    + scale_colour_manual(values=c("black", "blue", "red"))
    + theme(legend.position="bottom")
)
```

```
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
```

- Wealth index

```
# Wealth index
## varpred
reff_vareff_wealthindex <- varpred(reff_mod, "wealthindex", isolate=TRUE, modelname="varpred")
```

```
## Pop. average
reff_vareff_wealthindex_pop <- varpred(reff_mod, "wealthindex", isolate=TRUE, pop.ave=TRUE, modelname="

## Bias adjust
reff_vareff_wealthindex_adjust <- varpred(reff_mod, "wealthindex", isolate=TRUE, bias.adjust=TRUE, model

vareff_wealthindex <- reff_vareff_wealthindex
vareff_wealthindex$preds <- do.call("rbind", list(vareff_wealthindex$preds, reff_vareff_wealthindex_pop$
wealthindex_plot <- (plot(vareff_wealthindex)
    + labs(y="", colour="Model")
    + geom_hline(yintercept=true_prop_reff, lty=2, colour="grey")
    + scale_colour_manual(values=c("black", "blue", "red"))
    + theme(legend.position="bottom")
)
```

```
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
```

```
ggarrange(age_plot, wealthindex_plot, common.legend=TRUE)
```

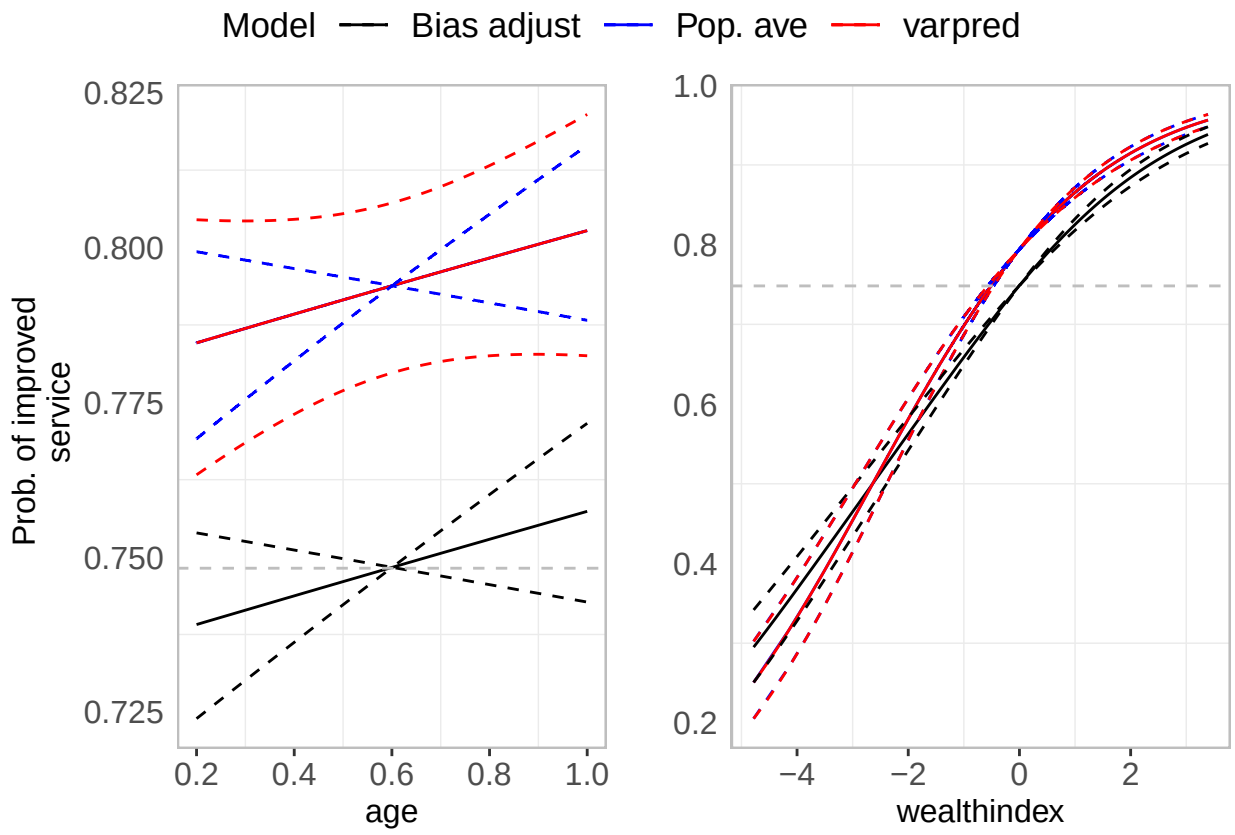


Figure 2: A comparison of population averaged, varpred-based and bias-adjusted predictions. For each of the predictors, the population averaging and varpred gives similar estimates, and slightly off the truth. However, when we apply bias-adjustment, the estimates are very precise (i.e., close to the truth).