Homework 04

Minji Jeong

```
df <- read.csv("https://faculty.washington.edu/cadolph/mle/nes92con.csv")

# Get nice colors (optional)
col <- brewer.pal(5, "Set1")
blue <- col[2]
orange <- col[5]
```

Answer a.

```
llk.oprobit4 <- function(param, x, y) {
  # preliminaries
  os <- rep(1, nrow(x))
  x  <- cbind(os, x)
  b  <- param[1:ncol(x)]
  t2 <- param[(ncol(x)+1)]
  t3 <- param[(ncol(x)+2)]

  # probabilities and penalty function
  xb <- x%*%b
  p1 <- log(pnorm(-xb))
  if (t2<=0)  p2 <- -(abs(t2)*10000)    # penalty function to keep t2>0
  else p2 <- log(pnorm(t2-xb)-pnorm(-xb))
  if (t3<=t2) p3 <- -((t2-t3)*10000)    # penalty to keep t3>t2
  else p3 <- log(pnorm(t3-xb)-pnorm(t2-xb))
  p4 <- log(1-pnorm(t3-xb))

  # -1 * log likelihood (optim is a minimizer)
  -sum(cbind(y==0,y==1,y==2,y==3) * cbind(p1,p2,p3,p4))
}

# In case you kept cbind(y==1,y==2,y==3,y==4)
# change the data so that it matches with your function
# df <- df |> mutate(bushapp = bushapp +1)
```

```
m1 <- bushapp ~ milforce + rbdist + econ + partyid + yrsofed

m1df <- extractdata(m1, df, na.rm=TRUE)
m1df |> dim()
```

```
## [1] 558   6
```

```
y <-  m1df[,1]
x1 <- m1df[,2:ncol(m1df)] |> as.matrix()

ls.result <- lm(y~x1)  # use ls estimates as starting values
stval <- c(coef(ls.result),1,2)    # initial guesses
oprobit.m1 <- optim(stval,llk.oprobit4,method="BFGS",hessian=T,y=y,x=x1)
pe.m1 <- oprobit.m1$par   # point estimates
vc.m1 <- solve(oprobit.m1$hessian)  # var-cov matrix
se.m1 <- sqrt(diag(vc.m1))    # standard errors
ll.m1 <- -oprobit.m1$value  # likelihood at maximum
```

```
tibble(term = c(ls.result$coefficients |> names(),"t2", "t3"),
       estimate.optim = pe.m1,
       std.error.optim = se.m1) |> kable(digits = 2)
```

term	estimate.optim	std.error.optim
(Intercept)	4.08	0.43
x1milforce	-0.33	0.06
x1rbdist	-0.21	0.04
x1econ	-0.35	0.06
x1partyid	0.26	0.03
x1yrsofed	-0.04	0.02
t2	0.86	0.07
t3	2.16	0.10

```
tibble(logll.optim = ll.m1) |> kable(digits = 2)
```

logll.optim
-587.93

Answer b.

```
sims <- 10000
simbetas <- mvrnorm(sims, pe.m1, vc.m1)       # draw parameters, using MASS::mvrnorm
```

```
# Standard way
# Create example counterfactuals
xhyp <- cfMake(m1, m1df, nscen=10)
nscen<-1:10

for (i in 1:5) {
  xhyp <- cfName(xhyp, paste0("Str.Dem, Milforce = ", i), scen=i)
  xhyp <- cfChange(xhyp, "milforce", x=i, scen=i)
  xhyp <- cfChange(xhyp, "rbdist", x=mean(m1df$rbdist), scen=i)
  xhyp <- cfChange(xhyp, "econ", x=mean(m1df$econ), scen=i)
  xhyp <- cfChange(xhyp, "yrsofed", x=mean(m1df$yrsofed), scen=i)
  xhyp <- cfChange(xhyp, "partyid", x=-3, scen=i)
```

```
}

for (i in 6:10) {
  xhyp <- cfName(xhyp, paste0("Str.Rep, Milforce = ", i-5), scen=i)
  xhyp <- cfChange(xhyp, "milforce", x=i-5, scen=i)
  xhyp <- cfChange(xhyp, "rbdist", x=mean(m1df$rbdist), scen=i)
  xhyp <- cfChange(xhyp, "econ", x=mean(m1df$econ), scen=i)
  xhyp <- cfChange(xhyp, "yrsofed", x=mean(m1df$yrsofed), scen=i)
  xhyp <- cfChange(xhyp, "partyid", x=3, scen=i)
}
```

```
# Using custom function
# Create example counterfactuals
xhyp <- cfMake(m1, m1df, nscen=10)

fun.pty.mf <- function(i, pty, pty.n, mf){
name.cf <- paste0(pty, ", Milforce=", mf)
xhyp <- cfName(xhyp,name.cf, scen=i)
xhyp <- cfChange(xhyp, "partyid",  x=pty.n, scen=i)
xhyp <- cfChange(xhyp, "milforce", x=mf,  scen=i)
}

i = 1:10
pty = rep(c("Str.Dem", "Str.Rep"), each=5)
pty.n = rep(c(-3, 3), each=5)
mf = rep(c(1:5), times=2)

xhyp <-fun.pty.mf(i, pty, pty.n, mf)
```

```
# Simulate expected probabilities (all four categories)
oprobit.ev.m1 <- oprobitsimev(xhyp, simbetas, cat=4)

tibble("Scenarios" = row.names(xhyp$x),
       "Strongly Disapproved" = oprobit.ev.m1$pe[,1],
       "Disapproved" = oprobit.ev.m1$pe[,2],
       "Approved" = oprobit.ev.m1$pe[,3],
       "Strongly Approved"  = oprobit.ev.m1$pe[,4])%>%
  kable(digits = 2) %>% kable_styling(full_width = FALSE, position = "center", font_size = 8)
```

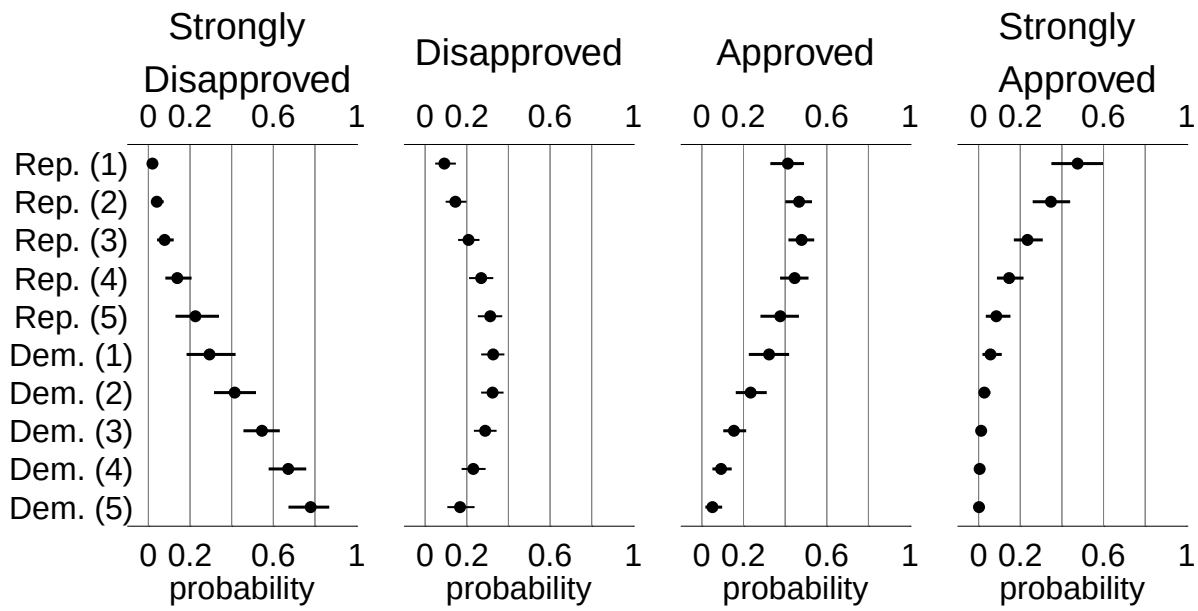
Scenarios	Strongly Disapproved	Disapproved	Approved	Strongly Approved
Str.Dem, Milforce=1	0.29	0.33	0.32	0.06
Str.Dem, Milforce=2	0.41	0.32	0.23	0.03
Str.Dem, Milforce=3	0.55	0.29	0.15	0.01
Str.Dem, Milforce=4	0.67	0.23	0.09	0.01
Str.Dem, Milforce=5	0.78	0.17	0.05	0.00
Str.Rep, Milforce=1	0.02	0.09	0.41	0.48
Str.Rep, Milforce=2	0.04	0.15	0.47	0.35
Str.Rep, Milforce=3	0.08	0.21	0.48	0.23
Str.Rep, Milforce=4	0.14	0.27	0.45	0.15
Str.Rep, Milforce=5	0.23	0.31	0.38	0.08

Answer c.

In the plot below, "Rep. (1)" means strong Republican who strongly support military force (Milforce=1).

```
sorted <- order(oprobit.ev.m1$pe[,1])

# Labels for plot
Letterlabel <- c(
  "Rep. (1)",  "Rep. (2)",  "Rep. (3)",  "Rep. (4)",  "Rep. (5)",
  "Dem. (1)",  "Dem. (2)",  "Dem. (3)",  "Dem. (4)",  "Dem. (5)")

trace1 <- ropeladder(x = oprobit.ev.m1$pe[sorted,1],
                     lower = oprobit.ev.m1$lower[sorted,1],
                     upper = oprobit.ev.m1$upper[sorted,1],
                     #labels = scenNames[sorted],
                     labels = Letterlabel,
                     size=0.5,
                     lex=1.5,
                     lineend="square",
                     plot=1
                     )

trace2 <- ropeladder(x = oprobit.ev.m1$pe[sorted,2],
                     lower = oprobit.ev.m1$lower[sorted,2],
                     upper = oprobit.ev.m1$upper[sorted,2],
                     size=0.5,
                     lineend="square",
                     plot=2
                     )

trace3 <- ropeladder(x = oprobit.ev.m1$pe[sorted,3],
                     lower = oprobit.ev.m1$lower[sorted,3],
                     upper = oprobit.ev.m1$upper[sorted,3],
                     #labels = scenNames[sorted],
                     size=0.5,
                     lex=1.5,
                     lineend="square",
                     plot=3
                     )

trace4 <- ropeladder(x = oprobit.ev.m1$pe[sorted,4],
                     lower = oprobit.ev.m1$lower[sorted,4],
                     upper = oprobit.ev.m1$upper[sorted,4],
                     size=0.5,
                     lex=1.5,
                     lineend="square",
                     plot=4
                     )

vertmark <- linesTile(x=c(0,0), y=c(0,1), plot=1,2)

tile(trace1,
     trace2,
     trace3,
```

```
    trace4,

    limits = c(-0.1,1),
    gridlines = list(type="xt"),
    topaxis=list(add=TRUE, at=seq(from=0, to=1.0, by=0.2)),
    xaxis=list(at=seq(from=0, to=1.0, by=0.2)),
    xaxistitle=list(labels="probability"),
    plottitle=list(labels=c("Strongly \n Disapproved",
                            "Disapproved",
                           "Approved",
                        "Strongly \n Approved")),
    width=list(spacer=2.5),
    height = list(plottitle=0.5,xaxistitle=2)
    )
```

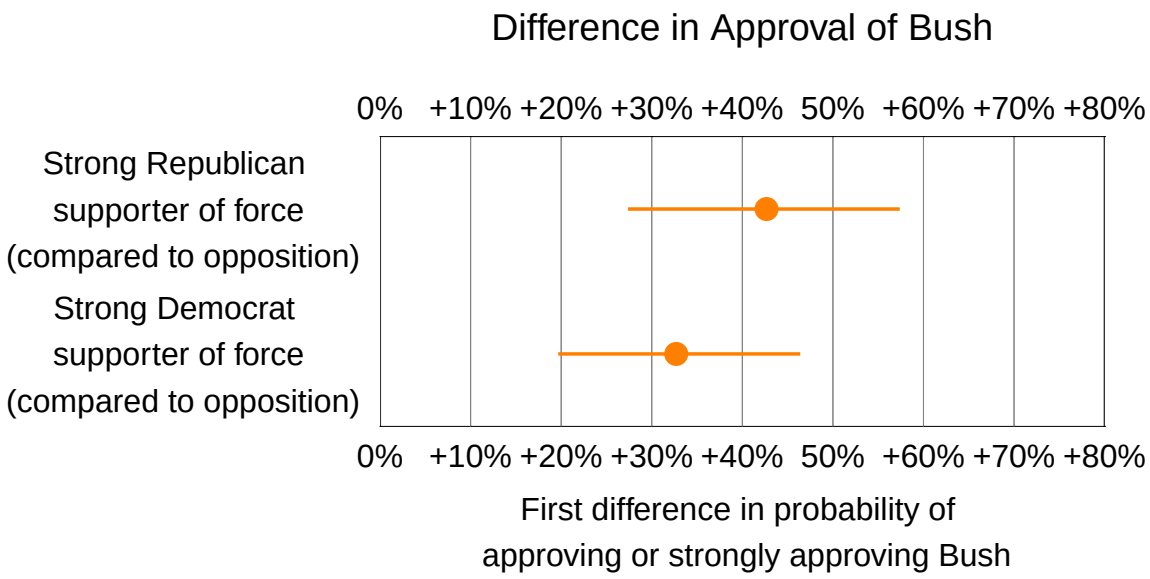


Answer d.

```
xhyp <- cfMake(m1, df, nscen=2)

xhyp <- cfName(xhyp,"Strong Democrat \n supporter of force \n (compared to opposition)",
               scen=1)
xhyp <- cfChange(xhyp, "partyid",  x=-3, xpre=-3, scen=1)
xhyp <- cfChange(xhyp, "milforce", x=1,  xpre=5,  scen=1)
```

```
xhyp <- cfName(xhyp,"Strong Republican \n supporter of force \n (compared to opposition)",
               scen=2)
xhyp <- cfChange(xhyp, "partyid",  x=3, xpre=3, scen=2)
xhyp <- cfChange(xhyp, "milforce", x=1, xpre=5, scen=2)

## Simulate first differences (two categories)
oprobit.fd.m1c <- oprobitsimfd(xhyp, simbetas, cat=4,
                               recode=list(c(1,2), c(3,4)) )
```

```
sortedc <- rev(order(oprobit.fd.m1c$pe[,2]))
scenNames <- row.names(xhyp$x)

trace1c <- ropeladder(x = oprobit.fd.m1c$pe[sortedc,2],
                      lower = oprobit.fd.m1c$lower[sortedc,2],
                      upper = oprobit.fd.m1c$upper[sortedc,2],
                      labels = scenNames[sortedc],
                      col=orange,
                      size=1,
                      lex=1.75,
                      lineend="square",
                      entryheight=0.40,
                      subentryheight=.8,
                      plot=1
                      )


vert.left <- linesTile(x=c(0,0), y=c(0,1), plot=1)
vert.right <- linesTile(x=c(0.8,0.8), y=c(0,1), plot=1)


tile(trace1c,
     vert.left,
     vert.right,
     limits=c(0,0.8),
     gridlines=list(type="xt"),
     topaxis=list(add=TRUE, at=seq(from=0, to=0.8, by=0.1),
                  labels=c("0%", "+10%", "+20%", "+30%", "+40%",
                           "50%", "+60%", "+70%", "+80%", "+40%")),
     xaxis=list(at=seq(from=0, to=0.8, by=0.1),
                  labels=c("0%", "+10%", "+20%", "+30%", "+40%",
                           "50%", "+60%", "+70%", "+80%", "+40%")),

     xaxistitle=list(labels="First difference in probability of \n approving or strongly approving Bush
     plottitle=list(labels="Difference in Approval of Bush"),

     width=list(plot=2),
     height=list(plottitle=1,xaxistitle=0.1)
     )
```



Answer e.

1. Overall, strong Republicans are more likely to approve of Bush than strong Democrats.
2. Regardless of the party you support, those who are in favor of the use of force are more likely to approve or strongly approve of Bush compared to those who oppose to the use of force.
3. The increase in probability of either approving or strongly approving of Bush given a shift from strong opposition to the use of force to strong support, for a respondent who is a strong Democrat and otherwise average, is about 0.33 on average.
4. The increase in probability of either approving or strongly approving of Bush given a shift from strong opposition to the use of force to strong support for a respondent who is a strong Republican and otherwise average is 0.43 on average.

Answer f.

```
df <-
  df |>
  mutate(econ3 = case_when(econ <= 2 ~ 1,
                           econ == 3 ~ 2,
                           TRUE ~ 3))

m2 <- econ3 ~ partyid
```

```
m2df <- extractdata(m2,df,na.rm=TRUE)
y2 <- m2df[,1]
x2 <- m2df[,2:ncol(m2df)] |> as.matrix()

llk.oprobit3 <- function(param, x, y) {
  os <- rep(1, nrow(x))
  x <- cbind(os, x)
  b <- param[1:ncol(x)]
  t2 <- param[(ncol(x)+1)]

  xb <- x%*%b
  p1 <- log(pnorm(-xb))
  if (t2<=0)  p2 <- -(abs(t2)*10000)
  else p2 <- log(pnorm(t2-xb)-pnorm(-xb))
  p3 <- log(1-pnorm(t2-xb))

  -sum(cbind(y==1,y==2,y==3) * cbind(p1,p2,p3))
}
```

```
ls.result2 <- lm(y2~x2)  # use ls estimates as starting values
stval2 <- c(coef(ls.result2),1)    # initial guesses
oprobit.m2 <- optim(stval2,llk.oprobit3,method="BFGS",hessian=T,y=y2,x=x2)
pe.m2 <- oprobit.m2$par   # point estimates
vc.m2 <- solve(oprobit.m2$hessian)  # var-cov matrix
se.m2 <- sqrt(diag(vc.m2))    # standard errors
ll.m2 <- -oprobit.m2$value  # likelihood at maximum
```

```
tibble(term = c("intercept","partyid", "t2"),
       estimate.optim = pe.m2,
       std.error.optim = se.m2) |>
  pander(digits = 2)
```

term	estimate.optim	std.error.optim
intercept	1.7	0.081
partyid	-0.16	0.024
t2	1.2	0.08

```
tibble(logll.optim = ll.m2) |>
  pander(digits = 2)
```

logll.optim
-525

Bonus

```
param=stval
x=x1
```

```
y=y

llk.oprobit.gen<-function(param, x, y){
      if (min(y) == 0) {y <- y + 1}
      n<-length(unique(y))
      os<-rep(1, nrow(x))
      x<-cbind(os, x)
      b<-param[1:ncol(x)]
      xb<-x%*%b

      tau <- c(0, param[(ncol(x)+1):length(param)])

      prob <-  matrix(0, nrow = nrow(x), ncol = n)
      prob[,1]<-log(pnorm(-xb))

      for (i in 2:(n-1)){
        if (tau[i]<=tau[i-1])
          prob[,i]<- -(abs(tau[i]*10000))
        else
          prob[,i]<- log(pnorm(tau[i]-xb)-pnorm(tau[i-1]-xb))
        }

      prob[,n]<- log(1-pnorm(tau[n-1]-xb))

      q<-NULL
      for (i in 1:n){
        q<-cbind(q, y == i)
        }
      -sum(q*prob)
    }
```

Now, check if the new function works.

```
test.m1 <- optim(stval,llk.oprobit.gen,method="BFGS",hessian=T,y=y,x=x1)
test.m2 <- optim(stval2,llk.oprobit.gen,method="BFGS",hessian=T,y=y2,x=x2)


test.m1$par == oprobit.m1$par
```

```
## (Intercept)  x1milforce    x1rbdist      x1econ   x1partyid   x1yrsofed
##        TRUE        TRUE        TRUE        TRUE        TRUE        TRUE
##
##        TRUE        TRUE
```

```
test.m1$value == oprobit.m1$value
```

```
## [1] TRUE
```

```
test.m2$par == oprobit.m2$par
```

```
## (Intercept)          x2
##        TRUE        TRUE        TRUE
```

```
test.m2$value == oprobit.m2$value
```

```
## [1] TRUE
```

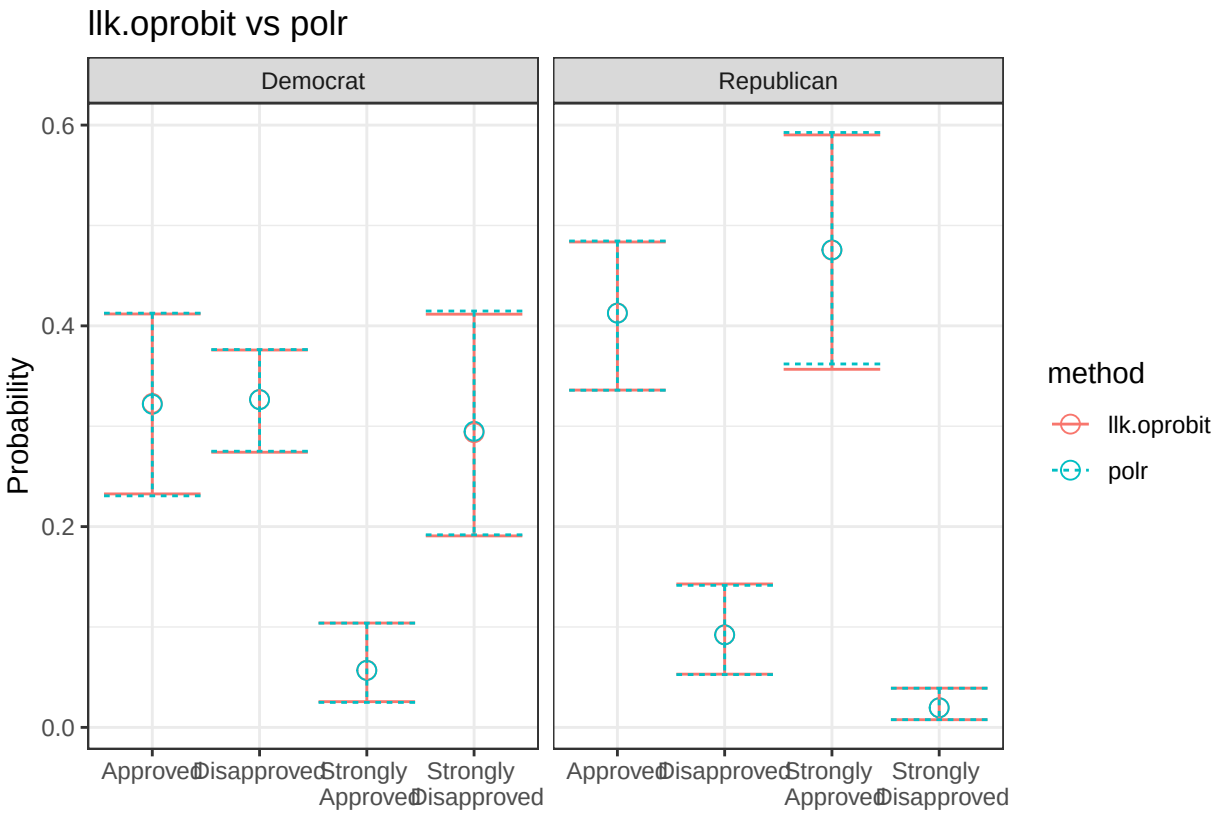
It works!

Answer g.

```
df <-
  df |>
  mutate(bushapp.f = as.factor(bushapp))

m3 <- bushapp.f ~ milforce + rbdist + econ + partyid + yrsofed
m3df <- extractdata(m3, df, na.rm=TRUE)
polr.m3 <- polr(m3,
                data=df,
                method="probit",
                na.action=na.omit,
                Hess = TRUE)

polr.pe.m3 <- c(polr.m3$coefficients, polr.m3$zeta)
polr.vc.m3 <- polr.m3 |> vcov()

simbetas.polr <- mvrnorm(sims, polr.pe.m3, polr.vc.m3)

oprobit.ev.polr.m3c <- oprobitsimev(xhyp, simbetas.polr, cat=4,
                                    constant = NA)

oprobit.ev.m1c <- oprobitsimev(xhyp, simbetas, cat=4)
```

```
glm.m3c <- oprobit.ev.polr.m3c |> as.data.frame() |> t()
ev.m1c  <- oprobit.ev.m1c |> as.data.frame() |> t()

check.df <-
  tibble(typ = rep(c("pe", "lower", "upper"), each=4) |> rep(times=4),
         party = rep(c("Democrat","Republican"), each=12) |> rep(times=2),
         ans = rep(c("Strongly \n Disapproved",
                     "Disapproved",
                     "Approved",
                     "Strongly \n Approved"), times=12),
         method = rep(c("llk.oprobit", "polr"), each=24),
         dta = c(ev.m1c[,1], ev.m1c[,2], glm.m3c[,1], glm.m3c[,2])
         ) |>
  spread(typ, dta)


check.df |>
  ggplot(aes(x=ans, y=pe, color=method, linetype=method) )+
  geom_point(size=3, shape=1 )+
  geom_errorbar(aes(x=ans, ymin=lower, ymax=upper))+
```

```
  facet_grid(~party)+
  labs(title = "llk.oprobit vs polr",
       x = "",
       y = "Probability"
       )+
  theme_bw()
```



As the plot above shows, we can produce almost same result using `polr`.

```
# You can also show the result as table
# check.df |> kable(digits = 2) |> kable_styling()
```