05b Model Results

Author names redacted

2020-12-07

Contents

1 Load data

Load the newly created data with all relevant covariates and subset to the European sample

```
df_full <- readRDS(paste0(here::here(), '/data/grayzone_model.rds'))
```

2 Fix polr functions

We use the polr function from the MASS package to compute an ordered probit. The base version of the function in the R package contains an error that does not take the log of differences in the reposed zetas which results in an optimization error where vmmin is infinite. A fixed version of the function was created and is loaded here. For this reason, the polr function is not loaded from the MASS package, but instead from the function below. All secondary functions from the MASS package are compatible with the output of the revised function.

```
# file MASS/R/polr.R
# copyright (C) 1994-2008 W. N. Venables and B. D. Ripley
# Use of transformed intercepts contributed by David Firth
#
#  This program is free software; you can redistribute it and/or modify
#  it under the terms of the GNU General Public License as published by
#  the Free Software Foundation; either version 2 or 3 of the License
```

```
#  (at your option).
#
#  This program is distributed in the hope that it will be useful,
#  but WITHOUT ANY WARRANTY; without even the implied warranty of
#  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
#  GNU General Public License for more details.
#
#  A copy of the GNU General Public License is available at
#  http://www.r-project.org/Licenses/
#
library(MASS)

polr <- function(formula, data, weights, start, ..., subset,
                 na.action, contrasts = NULL, Hess = FALSE,
                 model = TRUE,
                 method = c("logistic", "probit", "cloglog", "cauchit"))
{
    logit <- function(p) log(p/(1 - p))

    fmin <- function(beta) {
        theta <- beta[pc + 1L:q]
        gamm <- c(-Inf, cumsum(c(theta[1L], exp(theta[-1L]))), Inf)
        eta <- offset
        if (pc > 0)
            eta <- eta + drop(x %*% beta[1L:pc])
        pr <- pfun(gamm[y + 1] - eta) - pfun(gamm[y] - eta)
        if (all(pr > 0))
            -sum(wt * log(pr))
        else Inf
    }

    gmin <- function(beta)
    {
        jacobian <- function(theta) { ## dgamma by dtheta matrix
            k <- length(theta)
            etheta <- exp(theta)
            mat <- matrix(0 , k, k)
            mat[, 1] <- rep(1, k)
            for (i in 2:k) mat[i:k, i] <- etheta[i]
            mat
        }
        theta <- beta[pc + 1L:q]
        gamm <- c(-Inf, cumsum(c(theta[1L], exp(theta[-1L]))), Inf)
        eta <- offset
        if(pc > 0) eta <- eta + drop(x %*% beta[1L:pc])
        pr <- pfun(gamm[y+1] - eta) - pfun(gamm[y] - eta)
        p1 <- dfun(gamm[y+1] - eta)
        p2 <- dfun(gamm[y] - eta)
        g1 <- if(pc > 0) t(x) %*% (wt*(p1 - p2)/pr) else numeric(0)
        xx <- .polrY1*p1 - .polrY2*p2
        g2 <- - t(xx) %*% (wt/pr)
        g2 <- t(g2) %*% jacobian(theta)
        if(all(pr > 0)) c(g1, g2) else rep(NA, pc+q)
```

```
    }

    m <- match.call(expand.dots = FALSE)
    method <- match.arg(method)
    pfun <- switch(method, logistic = plogis, probit = pnorm,
                   cloglog = pgumbel, cauchit = pcauchy)
    dfun <- switch(method, logistic = dlogis, probit = dnorm,
                   cloglog = dgumbel, cauchit = dcauchy)
    if(is.matrix(eval.parent(m$data)))
        m$data <- as.data.frame(data)
    m$start <- m$Hess <- m$method <- m$model <- m$... <- NULL
    m[[1L]] <- as.name("model.frame")
    m <- eval.parent(m)
    Terms <- attr(m, "terms")
    x <- model.matrix(Terms, m, contrasts)
    xint <- match("(Intercept)", colnames(x), nomatch=0L)
    n <- nrow(x)
    pc <- ncol(x)
    cons <- attr(x, "contrasts") # will get dropped by subsetting
    if(xint > 0) {
        x <- x[, -xint, drop=FALSE]
        pc <- pc - 1
    } else warning("an intercept is needed and assumed")
    wt <- model.weights(m)
    if(!length(wt)) wt <- rep(1, n)
    offset <- model.offset(m)
    if(length(offset) <= 1) offset <- rep(0, n)
    y <- model.response(m)
    if(!is.factor(y)) stop("response must be a factor")
    lev <- levels(y)
    if(length(lev) <= 2) stop("response must have 3 or more levels")
    y <- unclass(y)
    q <- length(lev) - 1
    Y <- matrix(0, n, q)
    .polrY1 <- col(Y) == y
    .polrY2 <- col(Y) == y - 1
    if(missing(start)) {
      # try something that should always work -tjb
      u <- as.integer(table(y))
      u <- (cumsum(u)/sum(u))[1:q]
      zetas <-
         switch(method,
                "logistic"= qlogis(u),
                "probit"=   qnorm(u),
                "cauchit"=  qcauchy(u),
                "cloglog"=  -log(-log(u)) )
      s0 <- c(rep(0,pc),zetas[1],log(diff(zetas)))

##         # try logistic/probit regression on 'middle' cut
##         q1 <- length(lev) %/% 2
##         y1 <- (y > q1)
##         X <- cbind(Intercept = rep(1, n), x)
##         fit <-
```

```
##             switch(method,
##                    "logistic"= glm.fit(X, y1, wt, family = binomial(), offset = offset),
##                    "probit" = glm.fit(X, y1, wt, family = binomial("probit"), offset = offset),
##                    ## this is deliberate, a better starting point
##                    "cloglog" = glm.fit(X, y1, wt, family = binomial("probit"), offset = offset),
##                    "cauchit" = glm.fit(X, y1, wt, family = binomial("cauchit"), offset = offset))
##         if(!fit$converged)
##             stop("attempt to find suitable starting values failed")
##         coefs <- fit$coefficients
##         if(any(is.na(coefs))) {
##             warning("design appears to be rank-deficient, so dropping some coefs")
##             keep <- names(coefs)[!is.na(coefs)]
##             coefs <- coefs[keep]
##             x <- x[, keep[-1L], drop = FALSE]
##             pc <- ncol(x)
##           }
##         spacing <- logit((1L:q)/(q+1)) # just a guess
##         if(method != "logistic") spacing <- spacing/1.7
##         gammas <- -coefs[1L] + spacing - spacing[q1]
##         thetas <- c(gammas[1L], log(diff(gammas)))
##         s0 <- c(coefs[-1L], thetas)
    } else if(length(start) != pc + q)
    stop("'start' is not of the correct length")
    else {
        s0 <- if(pc > 0) c(start[seq_len(pc+1)], log(diff(start[-seq_len(pc)])))
        else c(start[1L], log(diff(start)))
      }
    res <- optim(s0, fmin, gmin, method="BFGS", hessian = Hess, ...)
    beta <- res$par[seq_len(pc)]
    theta <- res$par[pc + 1L:q]
    zeta <- cumsum(c(theta[1L],exp(theta[-1L])))
    deviance <- 2 * res$value
    niter <- c(f.evals=res$counts[1L], g.evals=res$counts[2L])
    names(zeta) <- paste(lev[-length(lev)], lev[-1L], sep="|")
    if(pc > 0) {
        names(beta) <- colnames(x)
        eta <- offset + drop(x %*% beta)
    } else eta <- offset + rep(0, n)

    cumpr <- matrix(pfun(matrix(zeta, n, q, byrow=TRUE) - eta), , q)
    fitted <- t(apply(cumpr, 1L, function(x) diff(c(0, x, 1))))
    dimnames(fitted) <- list(row.names(m), lev)
    fit <- list(coefficients = beta, zeta = zeta, deviance = deviance,
                fitted.values = fitted, lev = lev, terms = Terms,
                df.residual = sum(wt) - pc - q, edf = pc + q, n = sum(wt),
                nobs = sum(wt),
                call = match.call(), method = method,
        convergence = res$convergence, niter = niter, lp = eta)
    if(Hess) {
        dn <- c(names(beta), names(zeta))
        H <- res$hessian
        dimnames(H) <- list(dn, dn)
        fit$Hessian <- H
```

```
    }
    if(model) fit$model <- m
    fit$na.action <- attr(m, "na.action")
    fit$contrasts <- cons
    fit$xlevels <- .getXlevels(Terms, m)
    class(fit) <- "polr"
    fit
}
```

3 European sample

Create year dummy columns for the year fixed effects

```
df <- df_full %>%
  dplyr::filter(continent == "Europe")

df <- fastDummies::dummy_cols(df, select_columns = 'year') %>%
  dplyr::mutate_each(dplyr::funs(factor(.)), dplyr::starts_with("year_"))
```

3.1 Model 1

For the baseline model we just look at the relationship between the intensity of Russia intervention and whether the target is a NATO member as well as logged minimum distance from Russia. These are our two independent variables of interest. We use year fixed effects and cluster standard errors at the country level.

```
# Select model 1 variables
df_m1 <- df %>%
  dplyr::select(intensity, NATOmem_MEM, lnmindistkm_rus, dplyr::starts_with("year_"))

# Model
m1 <- polr(intensity ~ .,
           data = df_m1,
           method = "probit",
           Hess = TRUE)

# Country-clustered SE
m1 <- lmtest::coeftest(m1, vcov = sandwich::vcovCL(m1, factor(df$cname1)))
```

3.2 Model 2

Independent variables are NATO membership and logged minimum distance from Russia. Controls include logged GDP per capita, logged population, democracy, and nuclear status. We use year fixed effects.

```
# Select model 2 variables
df_m2 <- df %>%
  dplyr::select(intensity, NATOmem_MEM, lnmindistkm_rus, demo1, nuclear1, gdppc1_2010const, lnpop1, dply
  dplyr::mutate(gdppc = gdppc1_2010const/1000) %>%
  dplyr::select(-gdppc1_2010const)

# Model
m2 <- polr(intensity ~ .,
```

```
                  data = df_m2,
                  method = "probit",
                  Hess = TRUE)

# Country-clustered SE
m2 <- lmtest::coeftest(m2, vcov = sandwich::vcovCL(m2, factor(df$cname1)))
```

3.3 Model 3

For the final model, we include controls for CINC ratio, democracy, nuclear status, and civil war with year fixed effects and country-clustered standard errors.

```
# Select model 3 variables
df_m3 <- df %>%
  dplyr::select(intensity, NATOmem_MEM, lnmindistkm_rus, demo1, nuclear1, gdppc1_2010const, lnpop1, mil
  dplyr::mutate(gdppc = gdppc1_2010const/1000) %>%
  dplyr::mutate(milex = milex_sipri/1000) %>%
  dplyr::select(-gdppc1_2010const, -milex_sipri)

# Model
m3 <- polr(intensity ~ .,
           data = df_m3,
           method = "probit",
           Hess = TRUE)

# Country-clustered SE
m3 <- lmtest::coeftest(m3, vcov = sandwich::vcovCL(m3, factor(df$cname1)))
```

4 Relevant states sample

We run the same models as above on a different sample. Here, we limit sample to European states that meet any 1 of the following criteria:

1. **History of conflict** – European states that have had a MID or ICB incident with Russia/the Soviet Union from 1945-1994.
2. **Former Soviet Union/Warsaw Pact** – European states that were formerly members of either the Soviet Union or Warsaw Pact.
3. **Contiguity** – European states that are contiguous with Russia.

4.1 Model 4

Replicate model 1 with the new sample.

```
# Use same variables as model 1 but subset to
df_m4 <- df %>%
  dplyr::filter(relevant_conserv == 1)

df_m4vars <- df_m4 %>%
  dplyr::select(intensity, NATOmem_MEM, lnmindistkm_rus, dplyr::starts_with("year_"))
```

```r
# Model
m4 <- polr(intensity ~ .,
           data = df_m4vars,
           method = "probit",
           Hess = TRUE)

# Country-clustered SE
m4 <- lmtest::coeftest(m4, vcov = sandwich::vcovCL(m4, factor(df_m4$cname1)))
```

4.2 Model 5

Replicate model 2

```r
# Use same variables as model 2 but subset to
df_m5 <- df %>%
  dplyr::filter(relevant_conserv == 1)

df_m5vars <- df_m5 %>%
  dplyr::select(intensity, NATOmem_MEM, lnmindistkm_rus, demo1, nuclear1, gdppc1_2010const, lnpop1, dply
  dplyr::mutate(gdppc = gdppc1_2010const/1000) %>%
  dplyr::select(-gdppc1_2010const)

# Model
m5 <- polr(intensity ~ .,
                 data = df_m5vars,
                 method = "probit",
                 Hess = TRUE)

# Country-clustered SE
m5 <- lmtest::coeftest(m5, vcov = sandwich::vcovCL(m5, factor(df_m5$cname1)))
```

4.3 Model 6

Replicate model 3 on the new sample

```r
# Use same variables as model 3 but subset to
df_m6 <- df %>%
  dplyr::filter(relevant_conserv == 1)

df_m6vars <- df_m6 %>%
  dplyr::select(intensity, NATOmem_MEM, lnmindistkm_rus, demo1, nuclear1, gdppc1_2010const, lnpop1, mil
  dplyr::mutate(gdppc = gdppc1_2010const/1000) %>%
  dplyr::mutate(milex = milex_sipri/1000) %>%
  dplyr::select(-gdppc1_2010const, -milex_sipri)

# Model
m6 <- polr(intensity ~ .,
           data = df_m6vars,
           method = "probit",
           Hess = TRUE)

# Country-clustered SE
```

```r
m6 <- lmtest::coeftest(m6, vcov = sandwich::vcovCL(m6, factor(df_m6$cname1)))
```

5 Compiled results

Compiled results of the 6 models are shown below:

```r
# Make list of models
models <- list(m1, m2, m3, m4, m5, m6)

# Main coefficients
texreg::texreg(models,
               stars = c(0.01, 0.05, 0.1),
               omit.coef = "(y)",
               custom.coef.names = c("NATO member",
                                     "Russia distance",
                                     "Democracy",
                                     "Nuclear power",
                                     "Population",
                                     "GDP per cap",
                                     "Mil. spending"),
               groups = (list("Independent Variables" = 1:2, "Controls" = 3:7)),
               custom.gof.rows = list("Observations" = c("1,000", "921", "891", "376", "373", "346")),
               custom.header = list("Full sample" = 1:3, "Relevant states sample" = 4:6),
               custom.note = "All models include year-fixed effects with country-clustered standard err
               label = "table:model",
               float.pos = "h",
               caption = "Intensity of Russian Intervention: Ordered Probit Results",
#               scalebox = 0.9,
               use.packages = FALSE,
               file = paste0(here::here(), "/paper/figures/","model.tex"))

# Prep odds ratio
m1_tr <- texreg::extract(m1)
m1_tr@coef <- exp(m1_tr@coef)

m2_tr <- texreg::extract(m2)
m2_tr@coef <- exp(m2_tr@coef)

m3_tr <- texreg::extract(m3)
m3_tr@coef <- exp(m3_tr@coef)

m4_tr <- texreg::extract(m4)
m4_tr@coef <- exp(m4_tr@coef)

m5_tr <- texreg::extract(m5)
m5_tr@coef <- exp(m5_tr@coef)

m6_tr <- texreg::extract(m6)
m6_tr@coef <- exp(m6_tr@coef)

models_tr <- list(m1_tr, m2_tr, m3_tr, m4_tr, m5_tr, m6_tr)
```

```r
# Odds ratio
texreg::texreg(models_tr,
                  omit.coef = "(y)",
                  override.coef = list(exp(coef(m1)),
                                       exp(coef(m2)),
                                       exp(coef(m3)),
                                       exp(coef(m4)),
                                       exp(coef(m5)),
                                       exp(coef(m6))),
                  ci.force = TRUE,
                  ci.force.level = 0.9,
                  ci.test = 1,
                  digits = 2,
                  custom.coef.names = c("NATO member",
                                        "Russia distance",
                                        "Democracy",
                                        "Nuclear power",
                                        "Population",
                                        "GDP per cap",
                                        "Mil. spending"),
                  groups = (list("Independent Variables" = 1:2, "Controls" = 3:7)),
                  custom.gof.rows = list("Observations" = c("1,000", "921", "891", "376", "373", "346")),
                  custom.header = list("Full sample" = 1:3, "Relevant states sample" = 4:6),
                  custom.note = "All models are ordered probits and include year-fixed effects with cou
                  float.pos = "h!",
                  caption = "Intensity of Russian Intervention: Odds Ratios",
                  file = paste0(here::here(), "/paper/figures/","model_or.tex"))
```