```
R version 4.0.2 (2020-06-22) -- "Taking Off Again"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin17.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[R.app GUI 1.72 (7847) x86_64-apple-darwin17.0]

[Workspace restored from /Users/andreasmurr/.RData]
[History restored from /Users/andreasmurr/.Rapp.history]

> setwd('/Users/andreasmurr/Dropbox/PostModernBayes/Replication')
> # Computing quantities of interest and their uncertainty using Bayesian simulation
> # Andreas Murr, Richard Traunmueller, and Jeff Gill
> # Creates Figures 1 and 2 based on candidates ratings data
>
> # clear working memory
>
>   rm(list=ls())
>
> # load packages
>
>  library(foreign)
>  library(MCMCpack)
Loading required package: coda
Loading required package: MASS
##
## Markov Chain Monte Carlo Package (MCMCpack)
## Copyright (C) 2003-2022 Andrew D. Martin, Kevin M. Quinn, and Jong Hee Park
##
## Support provided by the U.S. National Science Foundation
## (Grants SES-0350646 and SES-0350613)
##
>
> # load data
>
>  data = read.dta("./Hainmueller2014/candidate.dta")
>
> # fit model (seed defaults to 12345)
>
>  m = MCMChregress(fixed=rating ~ 1 + atmilitary + atreligion + ated + atprof + atmale +
atinc + atrace + atage, random=~1, group="resID", r=1, R=1, data=data, mcmc = 30000)

Running the Gibbs sampler. It may be long, keep cool :)

**********:10.0%
**********:20.0%
**********:30.0%
```

```
**********:40.0%
**********:50.0%
**********:60.0%
**********:70.0%
**********:80.0%
**********:90.0%
**********:100.0%
>
> # numerical example in text (estimate level / coefficient estimates / second paragraph)
>
>  round(mean(m$mcmc[,"beta.atreligionCatholic"])*100, 1)
[1] -0.5
>  round(mean(m$mcmc[,"beta.atreligionCatholic"]<0)*100)
[1] 95
>
> # vertical axis of figures 1 and 2

>
>  ## levels of each variable (categories)
>  l.age = levels(data$atage)
>  l.rac = levels(data$atrace)
>  l.inc = levels(data$atinc)
>  l.gen = levels(data$atmale)
>  l.pro = levels(data$atprof)
>  l.col = levels(data$ated)
>  l.rel = levels(data$atreligion)
>  l.mil = levels(data$atmilitary)
>  ## reversed levels of variables
>  l = c(rev(l.age), rev(l.rac), rev(l.inc), rev(l.gen), rev(l.pro), rev(l.col), rev(l.rel),
rev(l.mil))
>  ## number of levels (catogeries) of variable
>  k.age = length(l.age)
>  k.rac = length(l.rac)
>  k.inc = length(l.inc)
>  k.gen = length(l.gen)
>  k.pro = length(l.pro)
>  k.col = length(l.col)
>  k.rel = length(l.rel)
>  k.mil = length(l.mil)
>  ## position of levels (category labels)
>  y.age.cat = 1:k.age
>  y.rac.cat = (max(y.age.cat)+3):(max(y.age.cat)+3+k.rac-1)
>  y.inc.cat = (max(y.rac.cat)+3):(max(y.rac.cat)+3+k.inc-1)
>  y.gen.cat = (max(y.inc.cat)+3):(max(y.inc.cat)+3+k.gen-1)
>  y.pro.cat = (max(y.gen.cat)+3):(max(y.gen.cat)+3+k.pro-1)
>  y.col.cat = (max(y.pro.cat)+3):(max(y.pro.cat)+3+k.col-1)
>  y.rel.cat = (max(y.col.cat)+3):(max(y.col.cat)+3+k.rel-1)
>  y.mil.cat = (max(y.rel.cat)+3):(max(y.rel.cat)+3+k.mil-1)
>  y.cat = c(y.age.cat, y.rac.cat, y.inc.cat, y.gen.cat, y.pro.cat, y.col.cat, y.rel.cat,
y.mil.cat)
>  ## position of variable labels
>  y.age.var = max(y.age.cat)+1
>  y.rac.var = max(y.rac.cat)+1
>  y.inc.var = max(y.inc.cat)+1
>  y.gen.var = max(y.gen.cat)+1
>  y.pro.var = max(y.pro.cat)+1
>  y.col.var = max(y.col.cat)+1
>  y.rel.var = max(y.rel.cat)+1
>  y.mil.var = max(y.mil.cat)+1
>  y.var = c(y.age.var, y.rac.var, y.inc.var, y.gen.var, y.pro.var, y.col.var, y.rel.var,
y.mil.var)
```

```
>  ## position of reference points
>  y.age.ref = max(y.age.cat)
>  y.rac.ref = max(y.rac.cat)
>  y.inc.ref = max(y.inc.cat)
>  y.gen.ref = max(y.gen.cat)
>  y.pro.ref = max(y.pro.cat)
>  y.col.ref = max(y.col.cat)
>  y.rel.ref = max(y.rel.cat)
>  y.mil.ref = max(y.mil.cat)
>  y.ref = c(y.age.ref, y.rac.ref, y.inc.ref, y.gen.ref, y.pro.ref, y.col.ref, y.rel.ref,
y.mil.ref)
>  ## position of distributions points
>  y.age.dis = y.age.cat[-length(y.age.cat)]
>  y.rac.dis = y.rac.cat[-length(y.rac.cat)]
>  y.inc.dis = y.inc.cat[-length(y.inc.cat)]
>  y.gen.dis = y.gen.cat[-length(y.gen.cat)]
>  y.pro.dis = y.pro.cat[-length(y.pro.cat)]
>  y.col.dis = y.col.cat[-length(y.col.cat)]
>  y.rel.dis = y.rel.cat[-length(y.rel.cat)]
>  y.mil.dis = y.mil.cat[-length(y.mil.cat)]
>  y.dis = c(y.age.dis, y.rac.dis, y.inc.dis, y.gen.dis, y.pro.dis, y.col.dis, y.rel.dis,
y.mil.dis)
>
> # ============
> # = figure 1 =
> # ============
>
> # preliminaries
>
>   # omit intercepts and sigmas and reverse order of other coefficients
>   m.coef = m$mcmc[,33:2]
>   # set scaling parameter for densities
>   # scale = 8
>   scale = 50
>   # compute density for each coefficient
>   den = apply(m.coef, 2, density)
>   # set range of y values
>   y.ran = c(1, max(den[ncol(m.coef)][[1]]$y) / scale + max(y.dis))
>   # set range of x values
>   x.ran = range(c(-.10, 0, apply(m.coef, 2, range), .15))
>
> # plot coefficients
>
>  cairo_pdf("figure-1.pdf", width=5, height=6, pointsize=10)
>   # set graphical parameters
>   par(mar=c(0,0,0,0), family="Gill Sans", mgp=c(1.5, .5, 0), oma=c(4, 6, 0, 1))
>   # create plot window
>   plot(x.ran, y.ran, type="n", axes = F)
>   # plot horizontal reference lines
>   abline(h=y.cat, col="grey", lwd=.5)
>   # plot vertical reference line
>   lines(x=c(0,0), y=y.ran+c(-3,0), lty=1)
>   # abline(v=0, lty=1)
>   # plot reference points
>   points(rep(0, length(y.ref)), y.ref, pch=16, cex=.5)
>   # plot density of each coefficient
>   for(i in ncol(m.coef):1){
+     den.adj = den[[i]]
+     den.adj$y = den.adj$y / scale + y.dis[i]
+     polygon(x=c(den.adj$x, rev(den.adj$x)), y=c(den.adj$y, rep(y.dis[i],
length(den.adj$y))), col=gray(.5, .5), border=NA)
```

```
+     lines(den.adj, lwd=.5)
+   }
>   # draw horizontal axis
>   axis(side=1, at=c(-.10,-.05,0,.05,.10,.15), lty=0)
>   # annotate horizontal axis
>  mtext(side=1, line=2.5, text="Change: Candidate Rating (0 'never support' - 1 'always
support')")
>   # draw vertical axis
>   axis(2, at=y.cat, las=1, lty=0, labels=F)
>   # annotate vertical axis
>   ## categories
>   mtext(side=2, at=y.cat, text=l, cex=.75, adj=0, line=5.5, las=1)
>   ## variables
>   mtext(side=2, at=y.var, text=paste(c("Age", "Race/Ethnicity", "Income", "Gender",
"Profession", "College", "Religion", "Military Service"), ":", sep=""), cex=.75, adj=0,
line=6, las=1)
>   # close graphical window
>   dev.off()
null device
          1
>
> # ============
> # = figure 2 =
> # ============
>
> # preliminaries
>
>   # compute ranks at each iteration
>   r = apply(abs(m.coef), 1, rank)
>   # compute probability of rank
>   p = apply(r, 1, function(x){prop.table(table(factor(x, levels=1:ncol(m.coef))))})
>
> # numerical example in text (estimate level / relative explanatory variable importance /
second paragraph)
>
>  colnames(p)[apply(p, 1, which.max)[30:32]]
[1] "beta.atedState university"      "beta.atedSmall college"
[3] "beta.atedIvy League university"
>  p[30:32,colnames(p)[apply(p, 1, which.max)[30:32]]]
   beta.atedState university beta.atedSmall college beta.atedIvy League university
30                0.7393333             0.2606667                              0
31                0.2606667             0.7393333                              0
32                0.0000000             0.0000000                              1
>
> # plot variable importance
>
>  cairo_pdf("figure-2.pdf", width=5, height=6, pointsize=10)
> # cairo_pdf("../Graphics/plot-hainmueller-rank.pdf", width=3, height=4, pointsize=10)
>   # set graphical parameters
>   par(mar=c(0,0,0,0), family="Gill Sans", mgp=c(1.5, .5, 0), oma=c(4, 6, 0, 1))
>   # create plot window
>   plot(c(1, ncol(m.coef)), y.ran, type="n", axes=F, xlab="", ylab="")
>   # plot probabilities of rank
>   for(i in 1:ncol(m.coef)){
+     x = c(sapply(1:ncol(m.coef), function(x) c(rep(NA,4), rep(x, 4)+c(-.5, .5, .5, -.5))))
[-c(1:4)]
+     y = rep(c(rep(NA, 4), rep(y.dis[i] + c(-.5,.5), each=2)), ncol(m.coef))[-c(1:4)]
+     polygon(x, y, col=gray(1-p[,i]), border=NA)
+   }
>   # draw horizontal axis
>   axis(1, at=ncol(m.coef)+1-rev(c(1, seq(5, ncol(m.coef), 5))), labels=rev(c(1, seq(5,
```

```
ncol(m.coef), 5))), lty=0)
>   # annotate horizontal axis
>   mtext(side=1, line=2, text="Rank of effect size", outer=T)
>   # draw vertical axis
> axis(2, at=y.cat, las=1, lty=0, labels=F)
>   # annotate vertical axis
>   ## categories
>   mtext(side=2, at=y.cat, text=l, cex=.75, adj=0, line=5.5, las=1)
>   ## variables
>   mtext(side=2, at=y.var, text=paste(c("Age", "Race/Ethnicity", "Income", "Gender",
"Profession", "College", "Religion", "Military Service"), ":", sep=""), cex=.75, adj=0,
line=6, las=1)
>   # create legend
>   x = c(sapply(1:ncol(m.coef), function(x) c(rep(NA,4), rep(x, 4)+c(-.5, .5, .5, -.5))))[-
c(1:4)]
>   y = rep(c(rep(NA, 4), rep(y.ran[2]-1 + c(-.5,.5), each=2)), ncol(m.coef))[-c(1:4)]
>   polygon(x, y, col=gray(1-(1:ncol(m.coef))/ncol(m.coef)), border=NA)
>   mtext(side=2, las=1, at=y.ran[2]-1, line=0, text="0")
>   mtext(side=4, las=1, at=y.ran[2]-1, line=0, text="1")
>   mtext(side=3, las=1, line=-1, text="Posterior probability", outer=T)
>   # close graphical window
>   dev.off()
null device
          1
>
> # ====================
> # = end source code =
> # ====================
>
```