```
R version 4.3.1 (2023-06-16) -- "Beagle Scouts"
Copyright (C) 2023 The R Foundation for Statistical Computing
Platform: aarch64-apple-darwin20 (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 1.79 (8238) aarch64-apple-darwin20]

[History restored from /Users/u1572955/.Rapp.history]

> setwd('/Users/u1572955/Dropbox/mexican-citizen-forecasting-2024/
replication')
> # Citizen forecasts of Mexican presidential elections, 2000-2024
> # Andreas Murr
> # Creates Figure 2 and Table 3
> # Writes functions mu.post.mvnorm() and alpha.post.dirichlet()
>
> # clear working memory
>
> rm(list = ls())
>
> # load packages
>
> library(MASS)
> library(MCMCpack)
Loading required package: coda
##
## Markov Chain Monte Carlo Package (MCMCpack)
## Copyright (C) 2003-2024 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)
##
Warning message:
package ‘MCMCpack’ was built under R version 4.3.3
```

```
>
> # write functions for posterior analysis
>
> mu.post.mvnorm = function(X, n.sim = 100000){
+   # assumes X is a matrix of vote share forecasts
+   # for p parties and n respondents
+   # creates draws from the posterior distribution of mu assuming a
normal inverse-wishart model
+   # where Sigma ~ Inv-Wishart_{nu0 = p+1} (I_p), see Gelman et al. 73 so
"that each of the correlations in Sigma has, marginally, a uniform
distribution"
+   # set priors
+   p = ncol(X)
+   n = nrow(X)
+   kappa0 = 1
+   Lambda0 = diag(p)
+   mu0 = rep(0, p)
+   nu0 = p + 1
+   # compute statistics
+   x.bar = apply(X, 2, mean)
+   S = cov(X) * (n - 1)
+   # compute posterior parameters
+   kappa1 = kappa0 + n
+   mu1 = kappa0 / kappa1 * mu0 + n / kappa1 * x.bar
+   nu1 = nu0 + n
+   Lambda1 = Lambda0 + S + (kappa0 * n / kappa1) * (x.bar -
mu0)%*%t(x.bar - mu0)
+   # simulate posterior mean
+   mu.draw = matrix(NA, nrow = n.sim, ncol = p)
+   # x.draw = matrix(NA, nrow = n.sim, ncol = p)
+   for (i in 1:n.sim){
+     S.draw = riwish(v = nu1, S = Lambda1)
+     mu.draw[i,] = mvrnorm(1, mu = mu1, Sigma = S.draw / kappa1)
+     # x.draw[i,] = mvrnorm(1, mu = mu.draw[i,], Sigma = S.draw)
+   }
+   # return results
+   mu.draw
+ }
>
> alpha.post.dirichlet = function(x, n.sim = 100000){
+   # assumes that x is a table of counts
+   # creates draws from posterior distribution of alpha of a dirichlet
distribution assuming a multinomial dirichlet model
+   # set prior
+   alpha = rep(1, length(x))
+   # simulate from posterior
+   r = rdirichlet(n = n.sim, alpha = x + alpha)
+   # return results
+   r
```

```
+ }
>
> # =============
> # = load data =
> # =============
>
> load("study-2-data.rdata")
>
> # ==========================================
> # = number of respondents mentioned in text =
> # ==========================================
>
> n.exp = c(nrow(X00), nrow(X06.1), nrow(X06.2), nrow(X06.3), nrow(X06.4),
nrow(X12), nrow(X18.1), nrow(X18.2))
>
> range(n.exp)
[1]  804 2273
>
> # ============
> # = figure 2 =
> # ============
>
> r = range(c(exp2000, exp2006, exp2012, exp2018, int2000, int2006,
int2012, int2018, v), na.rm = TRUE) + c(-2, 2)
>
> pdf("figure-2.pdf", width = 7, height = 8)
>
> par(mfcol = c(4, 2))
> par(mar = c(0, 0, 0, 0), oma = c(4, 4.5, 2, 1), las = 1)
>
> plot(exp2000, v[1,], xlim = r, ylim = r, xlab = "", ylab = "", axes = F,
pch = 16)
> abline(a = 0, b = 1, lty = 2)
> axis(2)
> box()
> mtext("2000", 2, 3.25, las = 3, font = 2)
> mtext("Actual", 2, 2, las = 3)
> mtext("Citizen forecasts", 3, .5, las = 1, font = 2)
> text(exp2000, v[1,], toupper(colnames(v)), pos = c(3, 4, 1))
>
> plot(exp2006, rep(v[2,], each = nrow(exp2006)), xlim = r, ylim = r, xlab
= "", ylab = "", axes = F, pch = 16)
> abline(a = 0, b = 1, lty = 2)
> axis(2)
> box()
> mtext("2006", 2, 3.25, las = 3, font = 2)
> mtext("Actual", 2, 2, las = 3)
> text(exp2006[3,], v[2,], toupper(colnames(v)), pos = c(3, 2, 1))
>
```

```
> plot(exp2012, rep(v[3,], each = nrow(exp2012)), xlim = r, ylim = r, xlab
= "", ylab = "", axes = F, pch = 16)
> abline(a = 0, b = 1, lty = 2)
> axis(2)
> box()
> mtext("2012", 2, 3.25, las = 3, font = 2)
> mtext("Actual", 2, 2, las = 3)
> text(exp2012, v[3,], toupper(colnames(v)), pos = c(2, 2, 2))
>
> plot(exp2018, rep(v[4,], each = nrow(exp2018)), xlim = r, ylim = r, xlab
= "", ylab = "", axes = F, pch = 16)
> abline(a = 0, b = 1, lty = 2)
> axis(1)
> axis(2)
> box()
> mtext("Predicted", 1, 2.5)
> mtext("2018", 2, 3.25, las = 3, font = 2)
> mtext("Actual", 2, 2, las = 3)
> text(exp2018[1,], v[4,], toupper(colnames(v)), pos = c(4, 2, 2, 4))
>
> # intentions
>
> plot(int2000, v[1,], xlim = r, ylim = r, xlab = "", ylab = "", axes = F,
pch = 16)
> abline(a = 0, b = 1, lty = 2)
> box()
> mtext("Vote intentions", 3, .5, las = 1, font = 2)
> text(int2000, v[1,], toupper(colnames(v)), pos = c(3, 4, 1))
>
> plot(int2006, rep(v[2,], each = nrow(int2006)), xlim = r, ylim = r, xlab
= "", ylab = "", axes = F, pch = 16)
> abline(a = 0, b = 1, lty = 2)
> box()
> text(int2006[3,], v[2,], toupper(colnames(v)), pos = c(3, 2, 1))
>
> plot(int2012, rep(v[3,], each = nrow(int2012)), xlim = r, ylim = r, xlab
= "", ylab = "", axes = F, pch = 16)
> abline(a = 0, b = 1, lty = 2)
> box()
> text(int2012, v[3,], toupper(colnames(v)), pos = c(2, 2, 4))
>
> plot(int2018, rep(v[4,], each = nrow(int2018)), xlim = r, ylim = r, xlab
= "", ylab = "", axes = F, pch = 16)
> abline(a = 0, b = 1, lty = 2)
> axis(1)
> box()
> mtext("Predicted", 1, 2.5)
> text(int2018[1,], v[4,], toupper(colnames(v)), pos = c(4, 2, 4, 4))
>
```

```
> dev.off()
null device
          1
>
> # ===========
> # = table 3 =
> # ===========
>
> # combine data
>
> X06 = rbind(X06.1, X06.2, X06.3, X06.4)
> X18 = rbind(X18.1, X18.2)
>
> Y06 = Y06.1 + Y06.2 + Y06.3 + Y06.4
> Y18 = Y18.1 + Y18.2
>
> # simulate from posterior
>
> set.seed(846128)
> mu00 = mu.post.mvnorm(X00, 100000)
> mu06 = mu.post.mvnorm(X06, 100000)
> mu12 = mu.post.mvnorm(X12, 100000)
> mu18 = mu.post.mvnorm(X18, 100000)
>
> alpha00 = alpha.post.dirichlet(Y00, 100000) * 100
> alpha06 = alpha.post.dirichlet(Y06, 100000) * 100
> alpha12 = alpha.post.dirichlet(Y12, 100000) * 100
> alpha18 = alpha.post.dirichlet(Y18[-2], 100000) * 100
>
> # compute rmse
>
> rmse.exp.00 = apply(mu00, 1, function(x){sqrt(mean((v[1,-4] - x)^2))})
> rmse.exp.06 = apply(mu06, 1, function(x){sqrt(mean((v[2,-4] - x)^2))})
> rmse.exp.12 = apply(mu12, 1, function(x){sqrt(mean((v[3,-4] - x)^2))})
> rmse.exp.18 = apply(mu18, 1, function(x){sqrt(mean((v[4,-2] - x)^2))})
>
> rmse.int.00 = apply(alpha00[,1:3], 1, function(x){sqrt(mean((x -
v[1,-4])^2))})
> rmse.int.06 = apply(alpha06[,1:3], 1, function(x){sqrt(mean((x -
v[2,-4])^2))})
> rmse.int.12 = apply(alpha12[,1:3], 1, function(x){sqrt(mean((x -
v[3,-4])^2))})
> rmse.int.18 = apply(alpha18[,1:3], 1, function(x){sqrt(mean((x -
v[4,-2])^2))})
>
> rmse.ran.00 = sqrt(mean((v[1,-4] - 1/3*100)^2))
> rmse.ran.06 = sqrt(mean((v[3,-4] - 1/3*100)^2))
> rmse.ran.12 = sqrt(mean((v[2,-4] - 1/3*100)^2))
> rmse.ran.18 = sqrt(mean((v[4,-2] - 1/3*100)^2))
```

```
>
> ran = c(round(rmse.ran.00, 1), round(rmse.ran.06, 1), round(rmse.ran.12,
1), round(rmse.ran.18, 1))
>
> exp.mean = c(mean(rmse.exp.00), mean(rmse.exp.06), mean(rmse.exp.12),
mean(rmse.exp.18))
> exp.quan = c(
+   paste("[", paste(formatC(quantile(rmse.exp.00, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = ""),
+   paste("[", paste(formatC(quantile(rmse.exp.06, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = ""),
+   paste("[", paste(formatC(quantile(rmse.exp.12, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = ""),
+   paste("[", paste(formatC(quantile(rmse.exp.18, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = "")
+   )
>
> int.mean = c(mean(rmse.int.00), mean(rmse.int.06), mean(rmse.int.12),
mean(rmse.int.18))
> int.quan = c(
+   paste("[", paste(formatC(quantile(rmse.int.00, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = ""),
+   paste("[", paste(formatC(quantile(rmse.int.06, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = ""),
+   paste("[", paste(formatC(quantile(rmse.int.12, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = ""),
+   paste("[", paste(formatC(quantile(rmse.int.18, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = "")
+   )
>
> # compute ratios
>
> r00.exp.ran = rmse.exp.00 / rmse.ran.00
> r06.exp.ran = rmse.exp.06 / rmse.ran.06
> r12.exp.ran = rmse.exp.12 / rmse.ran.12
> r18.exp.ran = rmse.exp.18 / rmse.ran.18
>
> r00.exp.int = rmse.exp.00 / rmse.int.00
> r06.exp.int = rmse.exp.06 / rmse.int.06
> r12.exp.int = rmse.exp.12 / rmse.int.12
> r18.exp.int = rmse.exp.18 / rmse.int.18
>
> r.exp.ran.m = c(mean(r00.exp.ran), mean(r06.exp.ran), mean(r12.exp.ran),
mean(r18.exp.ran))
> r.exp.int.m = c(mean(r00.exp.int), mean(r06.exp.int), mean(r12.exp.int),
mean(r18.exp.int))
>
> r.exp.ran.q = c(
+   paste("[", paste(formatC(quantile(r00.exp.ran, c(.025, .975)), format
```

```
= "f", 1), collapse = "; "), "]", sep = ""),
+   paste("[", paste(formatC(quantile(r06.exp.ran, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = ""),
+   paste("[", paste(formatC(quantile(r12.exp.ran, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = ""),
+   paste("[", paste(formatC(quantile(r18.exp.ran, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = "")
+ )
>
> r.exp.int.q = c(
+   paste("[", paste(formatC(quantile(r00.exp.int, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = ""),
+   paste("[", paste(formatC(quantile(r06.exp.int, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = ""),
+   paste("[", paste(formatC(quantile(r12.exp.int, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = ""),
+   paste("[", paste(formatC(quantile(r18.exp.int, c(.025, .975)), format
= "f", 1), collapse = "; "), "]", sep = "")
+ )
>
> # create table
>
> o = matrix(NA, nrow = 5*2, ncol = 5)
> o[seq(1, 7, 2),1] = formatC(exp.mean, format = "f", 1)
> o[seq(2, 8, 2),1] = exp.quan
> o[seq(1, 7, 2),2] = ran
> o[seq(2, 8, 2),2] = ""
> o[seq(1, 7, 2),3] = formatC(r.exp.ran.m, format = "f", 1)
> o[seq(2, 8, 2),3] = r.exp.ran.q
> o[seq(1, 7, 2),4] = formatC(int.mean, format = "f", 1)
> o[seq(2, 8, 2),4] = int.quan
> o[seq(1, 7, 2),5] = formatC(r.exp.int.m, format = "f", 1)
> o[seq(2, 8, 2),5] = r.exp.int.q
> o[9,1] = formatC(mean(c(rmse.exp.00, rmse.exp.06, rmse.exp.12,
rmse.exp.18)), format = "f", 1)
> o[9,2] = formatC(mean(ran), format = "f", 1)
> o[9,3] = formatC(mean(r.exp.ran.m), format = "f", 1)
> o[9,4] = formatC(mean(c(rmse.int.00, rmse.int.06, rmse.int.12,
rmse.int.18)), format = "f", 1)
> o[9,5] = formatC(mean(r.exp.int.m), format = "f", 1)
> o[10,1] = paste("[", paste(formatC(quantile(apply(cbind(rmse.exp.00,
rmse.exp.06, rmse.exp.12, rmse.exp.18), 1, mean), c(0.025, 0.975)), format
= "f", 1), collapse = "; "), "]", sep = "")
> o[10,2] = ""
> o[10,3] = paste("[", paste(formatC(quantile(apply(cbind(r00.exp.ran,
r06.exp.ran, r12.exp.ran, r18.exp.ran), 1, mean), c(0.025, 0.975)), format
= "f", 1), collapse = "; "), "]", sep = "")
> o[10,4] = paste("[", paste(formatC(quantile(apply(cbind(rmse.int.00,
rmse.int.06, rmse.int.12, rmse.int.18), 1, mean), c(0.025, 0.975)), format
```

```
= "f", 1), collapse = "; "), "]", sep = "")
> o[10,5] = paste("[", paste(formatC(quantile(apply(cbind(r00.exp.int,
r06.exp.int, r12.exp.int, r18.exp.int), 1, mean), c(0.025, 0.975)), format
= "f", 1), collapse = "; "), "]", sep = "")
>
> colnames(o) = c("CF", "RG", "CF/RG", "VI", "CF/VI")
> rownames(o) = rep("", 10)
> rownames(o)[seq(1, 10, 2)] = c(seq(2000, 2018, 6), "Overall")
> noquote(o)
        CF            RG   CF/RG       VI          CF/VI
2000    5.4           11.3 0.5         2.6         2.3
        [4.9; 5.9]         [0.4; 0.5] [1.2; 4.0] [1.3; 4.4]
2006    2.3           5.4  0.4         1.8         1.3
        [2.0; 2.6]         [0.4; 0.5] [1.3; 2.2] [1.0; 1.8]
2012    2.8           6.6  0.4         3.4         0.9
        [2.4; 3.2]         [0.4; 0.5] [1.3; 5.6] [0.5; 2.1]
2018    10.9          16.7 0.7         7.4         1.5
        [10.5; 11.3]       [0.6; 0.7] [5.8; 9.0] [1.2; 1.9]
Overall 5.3           10.0 0.5         3.8         1.5
        [5.1; 5.6]         [0.5; 0.5] [3.0; 4.6] [1.2; 2.2]
>
> # ====================
> # = end source code =
> # ====================
>
```