PAIRS ESM Networks

Emorie D Beck

April 1, 2017

Contents

1 Workspace

1.1 Packages

```
library(lavaan)
library(qgraph)
library(igraph)
library(GPArotation)
library(mlVAR)
library(graphicalVAR)
library(knitr)
library(gridExtra)
library(Rmisc)
library(psych)
library(rlist)
library(stargazer)
library(data.table)
library(tidyverse)
library(pander)
library(RColorBrewer)
library(animation)
```

1.2 Local Functions

```
meanSD_r2z2r <- function(x) {
  z <- fisherz(x)
  z[is.infinite(z)] <- NA
  x_bar <- mean(z, na.rm = T)
  x_sd <- sd(z, na.rm = T)
  r_bar <- fisherz2r(x_bar)
  r_sd <- fisherz2r(x_sd)
  return(c(r_bar, r_sd))
}
```

2 Prepare Data

The data include three waves of experience sampling method data from the Personality and Intimate Relationship Study. Data were previously cleaned to remove data points that did not meet inclusion criteria. ##Load Data

```
wave1_all <- tbl_df(read.csv("~/Box Sync/network/PAIRS/Wave 1/esm_w1_RENAMED.csv"))
wave4_all <- tbl_df(read.csv("~/Box Sync/network/PAIRS/Wave 4/esm_w4_RENAMED_all.csv"))
wave7_all <- tbl_df(read.csv("~/Box Sync/network/PAIRS/Wave 7/esm_w7_RENAMED_all.csv"))
```

2.1 Clean Data

Because the data sets include data that are not being used in this study, we extract the relevant columns (Subject ID, frequency, hour block, day of study, measurement point, and personality items) from the original data frames. Next, we rename the columns for later ease of use and visualization. Finally, because of the small sample size for waves 4 and 7, we merge those data sets.

```
#Getting necessary columns
#Keeping subject ID and all esm.BFI items
w1 <- dplyr::select(wave1_all, esm.IDnum.w1, esm.PRO01.w1, esm.PRO03.w1,
                    esm.PRO04.w1, esm.PRO05.w1, dplyr::matches("BFI"),
                    -dplyr::contains(".1."))
w4 <- dplyr::select(wave4_all, esm.IDnum.w4, esm.PRO01.w4, esm.PRO03.w4,
```

```
                    esm.PRO04.w4, esm.PRO05.w4, matches("BFI"))
w7 <- dplyr::select(wave7_all, esm.IDnum.w7, esm.PRO01.w7, esm.PRO03.w7,
                    esm.PRO04.w7, esm.PRO05.w7,  matches("BFI"))
w7 <- w7[,!(colnames(w7) %in% c("esm.BFI20.w7", "esm.BFI12.w7"))]

# column names for w1
varnames <- c("SID", "freq", "hourBlock", "day", "beepvar",
              "A_rude", "E_quiet", "C_lazy",
              "N_relaxed", "N_depressed", "E_outgoing",
              "A_kind", "C_reliable", "N_worried")

# column names for w4 and w7
varnames_w47 <- c("SID", "freq", "hourBlock", "day", "beepvar",
              "E_outgoing","E_quiet",
              "C_lazy","C_reliable",
              "N_worried","N_relaxed",
              "N_depressed", "A_rude",
              "A_kind")

# short column names (for plots)
varnames2 <- c("rude", "quiet", "lazy",
              "relaxed", "depressed", "outgoing",
              "kind", "reliable", "worried")


# rename columns
colnames(w1) <- varnames
colnames(w4) <- varnames_w47
colnames(w7) <- varnames_w47

# change subject IDs to factor
w1$SID <- factor(w1$SID)
w4$SID <- factor(w4$SID)
w7$SID <- factor(w7$SID)


# reorder w4 and w7 columns to match w1
w4 <- w4[,c(varnames,setdiff(names(w4), varnames))]
w7 <- w7[,c(varnames,setdiff(names(w7), varnames))]

# create wave variable before combining data sets.
w4$Wave <- "4"
w7$Wave <- "7"
# merge wave 4 and 7 data sets
w2 <- merge(w4, w7, all = T)
```

Variable	New Name	Description
esm.IDnum.w1	SID	numeric variable; identification number
esm.BFI37.w1	A_rude	agreeablness, negative; “During the last hour, how rude were you?” Likert scale from 1 to 5; 1 = Not a lot, 3 = Somewhat, 5 = Very
esm.BFI21.w1	E_quiet	extraversion, negative; “During the last hour, how quiet were you?” Likert scale from 1 to 5; 1 = Not a lot, 3 = Somewhat, 5 = Very
esm.BFI23.w1	C_lazy	conscientiousness, negative; “During the last hour, how lazy were you?” Likert scale from 1 to 5; 1 = Not a lot, 3 = Somewhat, 5 = Very
esm.BFI09.w1	N_relaxed	neuroticism, positive; “During the last hour, how relaxed were you?” Likert scale from 1 to 5; 1 = Not a lot, 3 = Somewhat, 5 = Very
esm.BFI04.w1	N_depressed	neuroticism, positive; “During the last hour, did you feel ‘depressed, blue’?” Likert scale from 1 to 5; 1 = Not a lot, 3 = Somewhat, 5 = Very
esm.BFI36.w1	E_outgoing	extraversion, positive; “During the last hour, how ‘outgoing, sociable’ were you?” Likert scale from 1 to 5; 1 = Not a lot, 3 = Somewhat, 5 = Very

Variable	New Name	Description
esm.BFI32.w1	A_kind	agreeablness, positive; "During the last hour, how 'considerate, kind' were you?" Likert scale from 1 to 5; 1 = Not a lot, 3 = Somewhat, 5 = Very
esm.BFI13.w1	C_reliable	conscientiousness, positive; "During the last hour, how reliable were you?" Likert scale from 1 to 5; 1 = Not a lot, 3 = Somewhat, 5 = Very
esm.BFI19.w1	N_worried	neuroticism, positive; "During the last hour, how worried were you?" Likert scale from 1 to 5; 1 = Not a lot, 3 = Somewhat, 5 = Very

2.2 Screen Participants

To be able to construct individual networks for participants, we ideally need approximately 50 measurement points. However, for current purposes, we will keep all participants who have at least 20 responses, lest we eliminate a large portion of our subjects.

```
# get frequency count of subject's responses
problem_w1 <- plyr::count(w1$SID)
problem_w2 <- plyr::count(w2$SID)

# extract subject IDs of those with < 10 measurement points
problem_w1  <- as.character(problem_w1$x[which(problem_w1$freq < 10)])
problem_w2 <- as.character(problem_w2$x[which(problem_w2$freq < 10)])

# save excluded subjects to a separate data frame
excluded <- w1[which(w1$SID %in% problem_w1),]
excluded_w2 <- w2[which(w2$SID %in% problem_w2),]

# remove excluded subjects
w1  <- w1[!(w1$SID %in% problem_w1),]
w2 <- w2[!(w2$SID %in% problem_w2),]
```

2.3 Missing Data Handling

Participants in the study only answered Agreeableness items if they indicated they were interacting with another person during the hour block previous to responding. To retain those measurement points for use in models later, we fill in gaps using within-person means of Agreeabless items.

```
for (i in unique(w1$SID)){
  mean_A_rude <- mean(w1$A_rude[w1$SID == i], na.rm = T)
  w1$A_rude[is.na(w1$A_rude) & w1$SID == i] <- mean_A_rude
  mean_A_kind <- mean(w1$A_kind[w1$SID == i], na.rm = T)
  w1$A_kind[is.na(w1$A_kind) & w1$SID == i] <- mean_A_kind
}

for (i in unique(w2$SID)){
  mean_A_rude <- mean(w2$A_rude[w2$SID == i], na.rm = T)
  w2$A_rude[is.na(w2$A_rude) & w2$SID == i] <- mean_A_rude
  mean_A_kind <- mean(w2$A_kind[w2$SID == i], na.rm = T)
  w2$A_kind[is.na(w2$A_kind) & w2$SID == i] <- mean_A_kind
}
```

2.4 Response Order and Centering

Below, we within-person center both data sets to aid in interpretation.

```
# retain cases where all personality data are retained
w1_com <- w1[complete.cases(w1[,c(6:14)]),]
w2_com <- w2[complete.cases(w2[,c(6:14)]),]
```

```r
# for waves 4 and 7, create a variable that combines wave and day of study
w2_com$waveDay <- paste(w2_com$Wave, w2_com$day, sep = ".")

# reorder data sets by SID, day, and block
w1_com  <- w1_com[order(w1_com$SID, w1_com$day, w1_com$hourBlock),]
w2_com <- w2_com[order(w2_com$SID, w2_com$waveDay, w2_com$hourBlock),]
```

```r
w1_com <- w1_com %>%
  group_by(SID) %>%
  arrange(day, hourBlock) %>%
  mutate(beepvar3 = seq(1, n(), 1))

w2_com <- w2_com %>%
  group_by(SID) %>%
  arrange(waveDay, hourBlock) %>%
  mutate(beepvar3 = seq(1, n(), 1))

w1_centered <- data.frame(
  ddply(w1_com[,-c(2:5,15)], .(SID),
        colwise(function(x) x-mean(x, na.rm = T))),  w1_com[,c(2:5,15)])
w2_centered <- data.frame(
  ddply(w2_com[,-c(2:5,15:17)], .(SID),
        colwise(function(x) x-mean(x, na.rm = T))), w2_com[,c(2:5, 15:17)])

# Make numeric subject IDs for each df because mlVAR won't run for factors #
w1_com$SID2 <- as.numeric(w1_com$SID)
w2_com$SID2 <- as.numeric(w2_com$SID)
w1_centered$SID2 <- as.numeric(w1_centered$SID)
w2_centered$SID2 <- as.numeric(w2_centered$SID)

# calculate individual for each variable
w1_com <- w1_com %>%
  group_by(SID) %>%
  mutate_each_(funs(comp = mean), vars = colnames(w1_com)[6:14]) %>%
  ungroup()
w2_com <- w2_com %>%
  group_by(SID) %>%
  mutate_each_(funs(comp = mean), vars = colnames(w2_com)[6:14]) %>%
  ungroup()

# calculate overaall means for all variables
means_w1 <- w1_com %>%
  dplyr::select(17:25) %>%
  summarise_each(funs = funs(mean = mean))
means_w2 <- w2_com %>%
  dplyr::select(19:27) %>%
  summarise_each(funs = funs(mean = mean))


# save final data frames to files
write.csv(w1_com, "~/Box Sync/network/PAIRS/Wave 1/esm_w1_networks.csv", row.names = F)
write.csv(w2_com, "~/Box Sync/network/PAIRS/Wave 4 + 7/esm_w47_networks.csv", row.names = F)
```

3 Population Level Analyses

Bringmann et al. (2013) developed a technique for assessing ESM data using a network approach. This approach utilizes a series of univariate multilvel vector autoregressive models (mlVAR) in which all items are entered simultaneously as predictors and individually as outcomes. Below we run the models for the nine personality items at each wave. The `mlVAR()` function automatically within-centers data, so we will enter the raw data into the models. Because of the number of observations we have for each usbject we use a lag 1 factorization. Because we have more than 6 predictors, we use orthogonal estimation of temporal and contemporaneous effects.

3.1 Univariate Multilevel Autoregressive Models

3.1.1 Wave 1

```
# raw data #
# not affected by using only complete cases of the data #
fit1_w1 <-
  mlVAR_test(w1_com,
        vars = colnames(w1_com)[6:14], #4:18
        idvar = "SID2",
        lags = 1,
        #dayvar = "day",
        beepvar = "beepvar3",
        temporal = "orthogonal",
        contemporaneous = "orthogonal",
        verbose = TRUE,
        scale = FALSE)
```

3.1.2 Wave 2

```
# raw data #
fit1_w2 <-
  mlVAR_test(w2_com,
        vars = colnames(w2_com)[6:14], #4:16
        idvar = "SID2",
        lags = 1,
        #dayvar = "day",
        beepvar = "beepvar3",
        temporal = "orthogonal",
        contemporaneous = "orthogonal",
        verbose = TRUE,
        scale = FALSE)
```

3.2 Table Results

mlVAR involves running a series of multilevel models, we include all of the results for each wave in a table.

3.2.1 Wave 1

```
augData <- fit1_w1$data
sjt.lmer(fit1_w1$output$A_rude, fit1_w1$output$E_quiet, fit1_w1$output$C_lazy,
         fit1_w1$output$N_relaxed, fit1_w1$output$N_depressed, fit1_w1$output$E_outgoing,
         fit1_w1$output$A_kind, fit1_w1$output$C_reliable, fit1_w1$output$N_worried,
         file = "~/Box Sync/network/PAIRS/Wave 1/mlVAR_raw.html",
         p.numeric = FALSE,
         show.ci = FALSE,
         sep.column = FALSE,
         pred.labels = c(paste(varnames2, "within"), paste(varnames2, "between")))


library(stargazer)
library(XML)

table <- readHTMLTable("~/Box Sync/network/PAIRS/Wave 1/mlVAR_raw.html", header=T)
table <- as.data.frame(table)
table <- table[-1, ]
colnames(table) <- c("", rep(c("B"), times = 9))
table[,1] <- as.character(table[,1])
table[c(22:27),1] <- c("$\\sigma^2$", "$\\pi_{00, SID2}$", "$\\N_{SID2}$",
                       "$\\ICC_{SID2}$", "Observations",
```

```
                      paste("$\\R^{2}", "\\Omega_{0}^2$"))
stargazer(table, summary=F, rownames=F, float.env = "sidewaystable",
          font.size = "footnotesize")
```

3.2.2 Wave 2

```
augData <- fit1_w2$data
sjt.lmer(fit1_w2$output$A_rude, fit1_w2$output$E_quiet, fit1_w2$output$C_lazy,
         fit1_w2$output$N_relaxed, fit1_w2$output$N_depressed, fit1_w2$output$E_outgoing,
         fit1_w2$output$A_kind, fit1_w2$output$C_reliable, fit1_w2$output$N_worried,
         file = "~/Box Sync/network/PAIRS/Wave 4 + 7/mlVAR_raw.html",
         p.numeric = FALSE,
         sep.column = FALSE,
         show.ci = FALSE,
         pred.labels = c(paste(varnames2, "within"), paste(varnames2, "between")))


library(stargazer)
library(XML)

table <- readHTMLTable("~/Box Sync/network/PAIRS/Wave 4 + 7/mlVAR_raw.html", header=T)
table <- as.matrix(as.data.frame(table))
table <- rbind(c("", rep(c("B"), times = 9)), table)
table <- as.data.frame(table)
colnames(table) <- c("", varnames2)
table[,1] <- as.character(table[,1])
table[c(23:28),1] <- c("$\\sigma^2$", "$\\pi_{00, SID2}$", "$\\N_{SID2}$",
                       "$\\ICC_{SID2}$", "Observations",
                       paste("$\\R^{2}", "\\Omega_{0}^2$"))
stargazer(table, summary=F, rownames=F, float.env = "sidewaystable",
          font.size = "footnotesize")
```

Save summaries of the models

```
#model summary
sum_fit1_w1              <- summary(fit1_w1)
sum_fit1_w2             <- summary(fit1_w2)
```

3.3 Plots

The graphs below show the raw, directed network the estimated, directed network using a univariate multilevel vector autoregressive models as (1) temporal and (2) contemporaneous netowrks

```
# load centrality data for plotting
load("~/Box Sync/network/PAIRS/manuscript/R/centrality_pop.RData")

#load communities data for plotting
load("~/Box Sync/network/PAIRS/manuscript/R/communities.RData")

border_colors <- brewer.pal(11,"PRGn")[1:3]

######## temporal ########
# calculate quantiles of inStrength
temporal_cutoffs_w1 <-
  quantile(temporal_centrality_w1_raw$node.centrality$InStrength,
           probs = c(1/3,2/3))
temporal_cutoffs_w2 <-
  quantile(temporal_centrality_w2_raw$node.centrality$InStrength,
           probs = c(1/3,2/3))

# assign colors to variables based on quantile
temporal_centrality_colors_w1 <-
  ifelse(temporal_centrality_w1_raw$node.centrality$InStrength <= temporal_cutoffs_w1[1],
         border_colors[1],
```

Table 2:

	rude	**quiet**	**lazy**	**relaxed**	**depressed**	**outgoing**	**kind**	**reliable**	**worried**
	b	b	b	b	b	b	b	b	b
Fixed Parts									
(Intercept)	1.65 ***	3.61 ***	1.49 **	2.73 ***	0.12	2.90 ***	-0.59	2.84 ***	2.94 ***
rude within	0.03 *	0.01	0.01	-0.05 *	0.02	-0.02	-0.01	-0.02	0.03
quiet within	0.01	0.00	0.00	0.01	-0.01	-0.00	-0.00	0.01	0.00
lazy within	-0.00	0.03 **	0.06 ***	-0.01	0.02 *	-0.03 **	0.01	-0.02 *	0.01
relaxed within	0.00	-0.04 **	0.03 *	0.04 **	-0.03 **	0.04 *	0.01	0.00	-0.05 ***
depressed within	-0.01	0.01	0.05 **	-0.01	0.11 ***	0.01	0.01	-0.03 *	0.03
outgoing within	-0.00	-0.01	0.02	0.00	0.00	0.02	0.01	0.01	-0.01
kind within	0.00	0.01	-0.02	-0.03	0.01	-0.03	0.03 *	0.01	0.05 ***
reliable within	-0.01	-0.01	0.01	-0.01	-0.00	-0.01	0.02 *	0.03 *	-0.01
worried within	0.00	0.03	-0.05 ***	-0.11 ***	0.05 ***	-0.05 **	-0.02 *	0.00	0.14 ***
rude between	-0.07		0.25 ***	0.15 *	0.02	-0.73 ***	0.27 **	0.04	0.14 *
quiet between	0.12 **	0.15 ***		0.17 ***	0.23 ***	0.04	-0.06	-0.39 ***	0.02
lazy between	-0.04	0.15 **	0.27 ***		-0.19 **	0.24 ***	0.13	0.14 *	-0.67 ***
relaxed between	0.19 ***	0.00	0.28 ***	-0.13 **		-0.05	0.20 **	0.03	0.38 ***
depressed between	0.14 **	-0.69 ***	0.04	0.24 ***	-0.07		0.46 ***	0.14 *	0.14 *
outgoing between	-0.11 **	0.13 ***	-0.06	0.06	0.11 **	0.21 ***		0.30 ***	0.01
kind between	-0.14 ***	0.02	-0.39 ***	0.09 *	0.03	0.09 *	0.43 ***		0.01
reliable between	0.04	0.10 *	0.02	-0.50 ***	0.38 ***	0.08	0.03	0.03	
worried between		-0.07	0.20 **	-0.04	0.27 ***	0.12 *	-0.26 **	-0.22 **	0.04
Random Parts									
σ^2	0.289	1.523	1.217	0.932	0.530	1.407	0.488	0.644	0.877
$\pi_{00,SID2}$	0.085	0.036	0.110	0.067	0.113	0.045	0.201	0.133	0.104
N_{SID2}	349	349	349	349	349	349	349	349	349
ICC_{SID2}	0.213	0.023	0.081	0.064	0.167	0.030	0.283	0.165	0.101
Observations	10775	10775	10775	10775	10775	10775	10775	10775	10775
R^2/Ω_0^2	.378 / .373	.134 / .128	.233 / .225	.277 / .270	.406 / .403	.177 / .172	.483 / .481	.372 / .369	.343 / .340
Notes	*p<.05 **p<.01 ***p<.001								

Table 3:

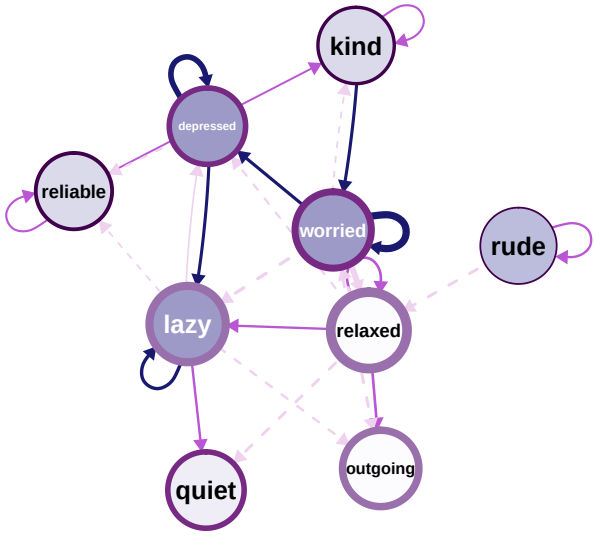
	rude	**quiet**	**lazy**	**relaxed**	**depressed**	**outgoing**	**kind**	**reliable**	**worried**
	b	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*
Fixed Parts									
Fixed Parts									
(Intercept)	0.69	3.27 ***	2.33 ***	2.82 ***	0.37	2.53 ***	0.57	1.94 ***	3.39 ***
rude within	0.02	0.00	0.01	0.01	0.01	-0.05	-0.00	0.03	0.05 *
quiet within	-0.02 *	0.02	0.00	-0.01	0.00	0.00	-0.00	-0.00	0.01
lazy within	0.02 *	-0.00	0.08 ***	0.04 **	0.01	0.01	0.01	-0.03 *	-0.00
relaxed within	-0.00	0.04	0.01	0.02	-0.02	0.00	0.00	0.00	0.01
depressed within	-0.00	-0.01	0.01	-0.08 ***	0.09 ***	-0.01	0.00	-0.02	0.02
outgoing within	-0.00	0.00	0.04 *	0.00	0.01	0.02	-0.00	-0.02	0.01
kind within	0.01	-0.01	0.03	0.01	0.01	0.01	0.06 ***	0.02	-0.00
reliable within	-0.01	-0.01	0.01	0.00	-0.00	0.03	0.00	0.06 ***	0.00
worried within	0.01	0.08 ***	-0.01	-0.08 ***	0.04 **	-0.03	-0.02	-0.02	0.16 ***
rude between	0.06		0.25 *	0.07	0.19 *	-0.73 ***	-0.00	0.37 ***	-0.12
quiet between	0.18 **	0.15 *		0.10	0.10	0.09	0.17	-0.48 ***	-0.00
lazy between	-0.01	0.07	0.10		-0.19 **	0.10	0.17	0.09	-0.44 ***
relaxed between	0.19 **	0.13 *	0.13	-0.22 *		-0.03	0.16	-0.05	0.43 ***
depressed between	0.29 ***	-0.66 ***	0.11	0.12	-0.02		0.26	0.43 ***	-0.09
outgoing between	-0.14 **	-0.00	0.08	0.09	0.06	0.10 *		0.16 **	0.03
kind between	-0.15 *	0.28 ***	-0.61 ***	0.12	-0.03	0.36 ***	0.39 **		0.00
reliable between	0.08	-0.02	-0.00	-0.31 ***	0.26 ***	-0.02	0.04	0.00	
worried between		0.03	0.26 **	0.02	0.24 **	0.29 ***	-0.42 **	-0.19 *	0.17
Random Parts									
σ^2	0.290	1.421	1.144	0.851	0.440	1.340	0.490	0.663	0.786
$\pi_{00,SID2}$	0.089	0.041	0.114	0.114	0.109	0.053	0.279	0.105	0.176
N_{SID2}	157	157	157	157	157	157	157	157	157
ICC_{SID2}	0.227	0.027	0.089	0.115	0.184	0.038	0.351	0.133	0.175
Observations	6434	6434	6434	6434	6434	6434	6434	6434	6434
R^2/Ω_0^2	.383 / .382	.138 / .132	.240 / .237	.22 / .242	.410 / .408	.176 / .173	.494 / .493	.365 / .363	.351 / .32
Notes	*p<.05 **p<.01 ***p<.001								

```
  ifelse(temporal_centrality_w1_raw$node.centrality$InStrength <= temporal_cutoffs_w1[2],
         border_colors[2],border_colors[3]))
temporal_centrality_colors_w2 <-
  ifelse(temporal_centrality_w2_raw$node.centrality$InStrength <= temporal_cutoffs_w2[1],
         border_colors[1],
  ifelse(temporal_centrality_w2_raw$node.centrality$InStrength <= temporal_cutoffs_w2[2],
         border_colors[2], border_colors[3]))


######## contemporaneous ########
# calculate quantiles of Strength
contemporaneous_cutoffs_w1 <-
  quantile(contemporaneous_centrality_w1_raw$node.centrality$Strength,
           probs = c(1/3,2/3))
contemporaneous_cutoffs_w2 <-
  quantile(contemporaneous_centrality_w2_raw$node.centrality$Strength,
           probs = c(1/3,2/3))

# assign colors to variables based on quantile
contemporaneous_centrality_colors_w1 <-
  ifelse(contemporaneous_centrality_w1_raw$node.centrality$Strength <=
           contemporaneous_cutoffs_w1[1], border_colors[1],
  ifelse(contemporaneous_centrality_w1_raw$node.centrality$Strength <=
           contemporaneous_cutoffs_w1[2], border_colors[2], border_colors[3]))
contemporaneous_centrality_colors_w2 <-
  ifelse(contemporaneous_centrality_w2_raw$node.centrality$Strength <=
           contemporaneous_cutoffs_w2[1], border_colors[1],
  ifelse(contemporaneous_centrality_w2_raw$node.centrality$Strength <=
           contemporaneous_cutoffs_w2[2], border_colors[2], border_colors[3]))


# run but don't generate temporal plot
  plot_w1 <- plot(fit1_w1,
     "temporal",
     layout = "spring",
     labels = varnames2, # can be a vector with node labels
     groups = sgc_w1_temporal,
     nonsig = 'hide',
     curve = -1,
     legend = FALSE,
     details = FALSE,
     mar = c(5,5,5,5), # controls margins
     border.color = temporal_centrality_colors_w1, # controls the border colors
     border.width = temporal_centrality_w1_raw$node.centrality$InStrength*40,
     title = "Temporal\nWave 1",
     loop = .7, # controls self-loop size, defaults to 1
     node.width = 1.6,
     edge.width = 1,
     asize = 5,
     label.font = 2,
     label.fill.vertical = 1,
     label.fill.horizontal = 1,
     edge.color = "black",
     color = t(brewer.pal(9, "Purples")[1:(length(sgc_w1_temporal))]),
      DoNotPlot = TRUE)

# run but don't generate contemporaneous plot
plot_w1_contemp <-   plot(fit1_w1,
     "contemporaneous",
     layout = "spring",
     labels = varnames2, # can be a vector with node labels
     groups = sgc_w1_contemp,
     nonsig = 'hide',
     curve = -1,
     legend = FALSE,
     details = FALSE,
     mar = c(5,5,5,5), # controls margins
```

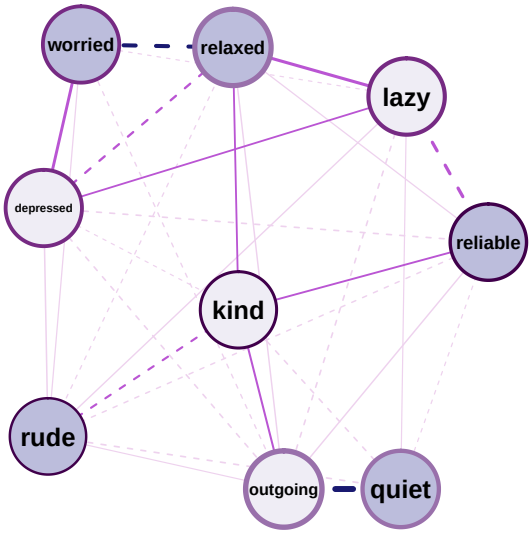
```
    border.color = contemporaneous_centrality_colors_w1, # controls the border colors
    border.width = contemporaneous_centrality_w1_raw$node.centrality$Strength*5,
    loop = .7, # controls self-loop size, defaults to 1
    node.width = 1.6,
    edge.width = 2,
    label.font = 2,
    label.fill.vertical = 1,
    label.fill.horizontal = 1,
    title = "Contemporaneous\nWave 1",
    esize = 3,
    edge.color = "black",
    color = brewer.pal(9, "Purples")[seq(2, length(sgc_w1_contemp) * 2 ,2)],
    DoNotPlot = TRUE)

 plot_w2 <- plot(fit1_w2,
    "temporal",
    layout = plot_w1$layout,
    labels = varnames2, # can be a vector with node labels
    groups = sgc_w2_temporal,
    nonsig = 'hide',
    curve = -1,
    legend = FALSE,
    details = FALSE,
    mar = c(5,5,5,5), # controls margins
    border.color = temporal_centrality_colors_w2, # controls the border colors
    border.width = temporal_centrality_w2_raw$node.centrality$InStrength*40,
    title = "Wave 2",
    loop = .7, # controls self-loop size, defaults to 1
    node.width = 1.6,
    edge.width = 1,
    asize = 5,
    label.font = 2,
    label.fill.vertical = 1,
    label.fill.horizontal = 1,
    edge.color = "black",
    color = brewer.pal(9, "Purples")[1:length(sgc_w2_temporal)],
    DoNotPlot = TRUE)

 # run but don't generate contemporaneous plot
plot_w2_contemp <-   plot(fit1_w2,
    "contemporaneous",
    layout = plot_w1_contemp$layout,
    labels = varnames2, # can be a vector with node labels
    groups = sgc_w2_contemp,
    nonsig = 'hide',
    curve = -1,
    legend = FALSE,
    details = FALSE,
    mar = c(5,5,5,5), # controls margins
    border.color = contemporaneous_centrality_colors_w2, # controls the border colors
    border.width = contemporaneous_centrality_w2_raw$node.centrality$Strength*5,
    loop = .7, # controls self-loop size, defaults to 1
    node.width = 1.6,
    edge.width = 2,
    label.font = 2,
    label.fill.vertical = 1,
    label.fill.horizontal = 1,
    title = "Wave 2",
    esize = 3,
    edge.color = "black",
    color = brewer.pal(9, "Purples")[seq(2, length(sgc_w2_contemp) * 2 ,2)],
    DoNotPlot = TRUE)

#change lines to dashed
plot_w1_contemp$graphAttributes$Edges$lty[plot_w1_contemp$Edgelist$weight < 0] <- 2
plot_w1$graphAttributes$Edges$lty[plot_w1$Edgelist$weight < 0] <- 2
plot_w2_contemp$graphAttributes$Edges$lty[plot_w2_contemp$Edgelist$weight < 0] <- 2
```

```
plot_w2$graphAttributes$Edges$lty[plot_w2$Edgelist$weight < 0] <- 2

#change line colors
plot_w1$graphAttributes$Edges$color <-
  ifelse(plot_w1$Edgelist$weight <.02, "thistle2",
  ifelse(plot_w1$Edgelist$weight <.04, "mediumorchid", "midnightblue"))
plot_w1_contemp$graphAttributes$Edges$color <-
  ifelse(abs(plot_w1_contemp$Edgelist$weight) < .1, "thistle2",
  ifelse(abs(plot_w1_contemp$Edgelist$weight) < .3, "mediumorchid", "midnightblue"))
plot_w2$graphAttributes$Edges$color <-
  ifelse(plot_w2$Edgelist$weight <.02, "thistle2",
  ifelse(plot_w2$Edgelist$weight <.04, "mediumorchid", "midnightblue"))
plot_w2_contemp$graphAttributes$Edges$color <-
  ifelse(abs(plot_w2_contemp$Edgelist$weight) < .1, "thistle2",
  ifelse(abs(plot_w2_contemp$Edgelist$weight) < .3, "mediumorchid", "midnightblue"))

#change variable names
plot_w1$graphAttributes$Nodes$labels <- varnames2
plot_w1_contemp$graphAttributes$Nodes$labels <- varnames2
plot_w2$graphAttributes$Nodes$labels <- varnames2
plot_w2_contemp$graphAttributes$Nodes$labels <- varnames2

#change font color for dark nodes
dark_colors <- c("#9E9AC8", "#807DBA", "#6A51A3", "#54278F", "#3F007D")
plot_w1$graphAttributes$Nodes$label.color[plot_w1$graphAttributes$Nodes$color %in% dark_colors] <- "white"
plot_w2$graphAttributes$Nodes$label.color[plot_w2$graphAttributes$Nodes$color %in% dark_colors] <- "white"
plot_w1_contemp$graphAttributes$Nodes$label.color[plot_w1_contemp$graphAttributes$Nodes$color %in% dark_colors] <- "white"
plot_w2_contemp$graphAttributes$Nodes$label.color[plot_w2_contemp$graphAttributes$Nodes$color %in% dark_colors] <- "white"


# generate modified plots
par(mfrow = c(2,2))
plot(plot_w1)
plot(plot_w1_contemp)
plot(plot_w2)
plot(plot_w2_contemp)
```

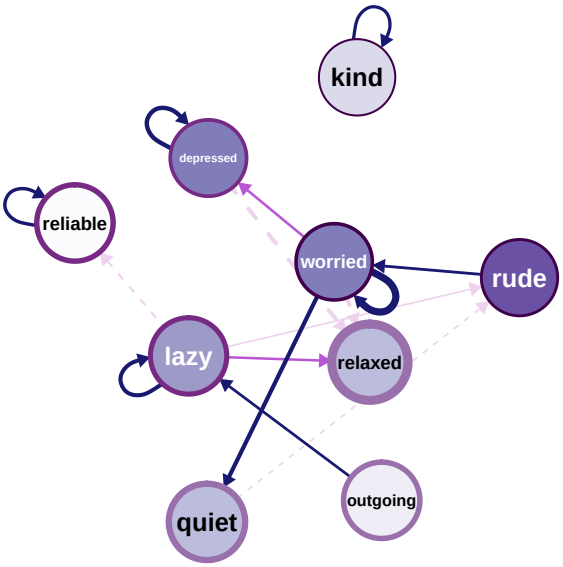
Temporal
Wave 1



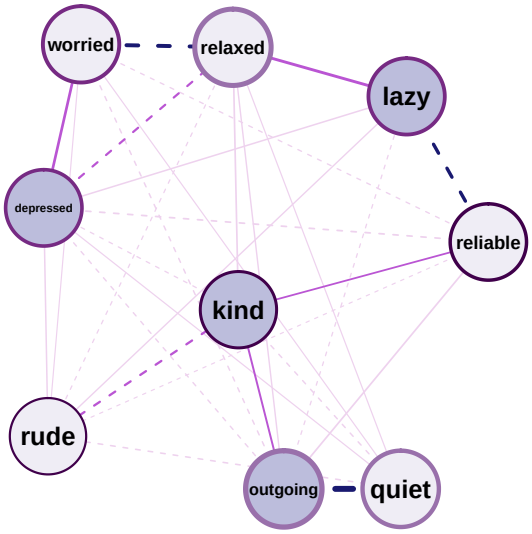
Contemporaneous
Wave 1



Wave 2



Wave 2



3.4 Edge Stability

The easiest and most straightforward way to assess the stability in responses across time is simply to correlate the regression coefficients at each time point with those at the other time points. We do this below for models generated using raw and centered data.
For these correlations, we are correlating two vectors, each of which contains 81 weights (9 x 9). We do so once for temporal and once for contemporaneous effects.

```
cors <- data.frame(
  comparison = c("W1 v. W2"),
  type = c("temporal", "contemporaneous"),
  raw_cor =
    c(cor(sum_fit1_w1$temporal$fixed,
          sum_fit1_w2$temporal$fixed),
```

```
      cor(sum_fit1_w1$contemporaneous$pcor,
          sum_fit1_w2$contemporaneous$pcor)),
  raw_p =
    c(cor.test(sum_fit1_w1$temporal$fixed,
               sum_fit1_w2$temporal$fixed)$p.value,
      cor.test(sum_fit1_w1$contemporaneous$pcor,
          sum_fit1_w2$contemporaneous$pcor)$p.value))

pandoc.table(cors, summary = F,
             caption = "Correlations of Temporal Fixed Effects Edges Across Waves")
```

Table 4: Correlations of Temporal Fixed Effects Edges Across Waves

comparison	type	raw_cor	raw_p
W1 v. W2	temporal	0.6759	4.428e-12
W1 v. W2	contemporaneous	0.9922	2.575e-32

3.5 Local Network Structure

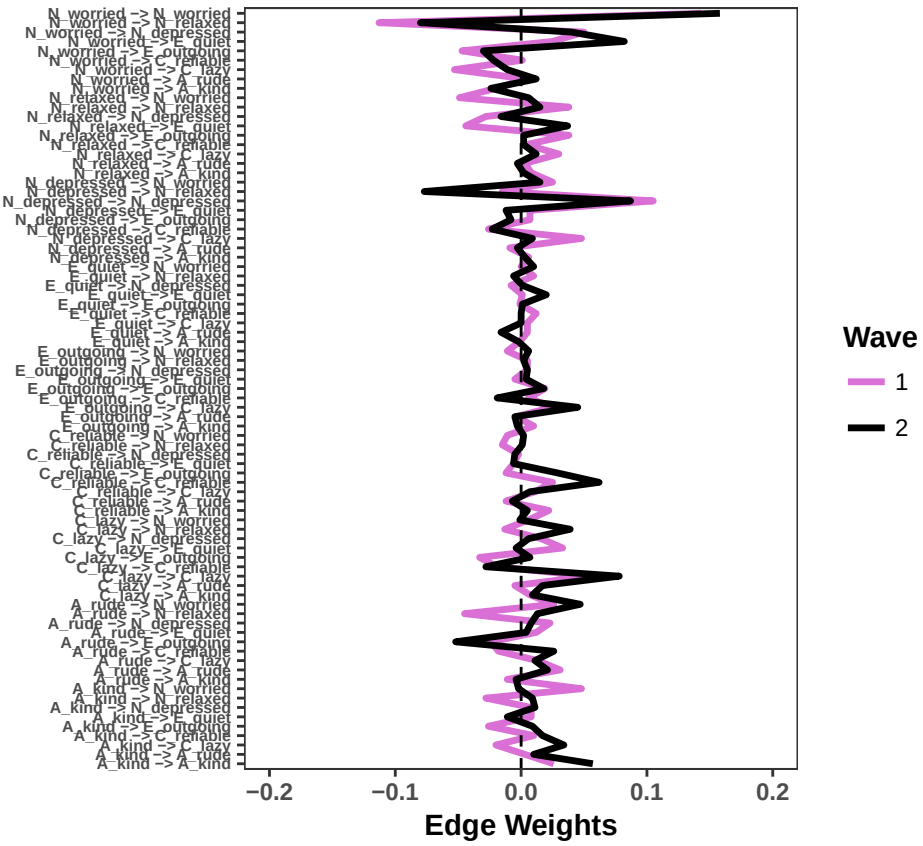
Another way of assessing network structure is locally – that is, by looking at the properties of specific nodes. Below, we test two families of measures of local network structure – centrality and clustering.

3.5.1 Centrality

Centrality refers to the relative importance of a focal node to the structure and dynamics of a network. In other words, it provides information about a node's role in the context of other nodes.

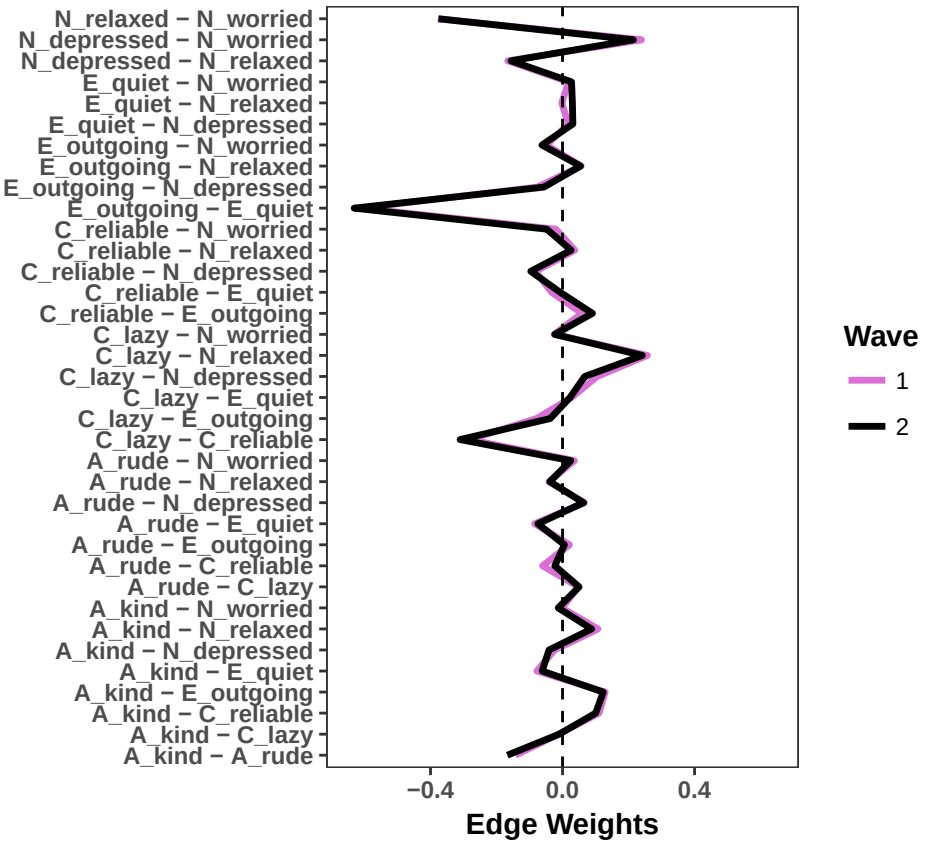
```
temporal_effects_w1 <- sum_fit1_w1$temporal
temporal_effects_w2 <- sum_fit1_w2$temporal

contemp_effects_w1 <- fit1_w1$results$Theta$pcor$mean
contemp_effects_w2 <- fit1_w2$results$Theta$pcor$mean

varnames2 <- c("A_rude", "E_quiet", "C_lazy",
               "N_relaxed", "N_depressed", "E_outgoing",
               "A_kind", "C_reliable", "N_worried")

contemp_long_fun <- function(fit){
  colnames(fit) <- varnames2; rownames(fit) <- varnames2
  fit <- fit[,order(colnames(fit))]
  fit <- fit[order(rownames(fit)),]
  fit[lower.tri(fit, diag = T)] <- NA
  fit.long <- tbl_df(fit) %>%
    mutate(Var1 = colnames(.),
           type = "Contemporaneous") %>%
    gather(key = Var2, value = weight, A_kind:N_worried) %>%
    filter(!is.na(weight)) %>%
    unite(var, Var1, Var2, sep = ".", remove = F)
}

contemp_eff_w1 <- contemp_long_fun(contemp_effects_w1)
contemp_eff_w2 <- contemp_long_fun(contemp_effects_w2)

temporal_effects_w1$V1V2 <- paste(temporal_effects_w1$from, temporal_effects_w1$to, sep = " -> ")
temporal_effects_w2$V1V2 <- paste(temporal_effects_w2$from, temporal_effects_w2$to, sep = " -> ")
contemp_eff_w1$V1V2 <- paste(contemp_eff_w1$Var1, contemp_eff_w1$Var2, sep = " - ")
contemp_eff_w2$V1V2 <- paste(contemp_eff_w2$Var1, contemp_eff_w2$Var2, sep = " - ")

temporal_effects_w1$wave <- "1"
temporal_effects_w2$wave <- "2"
contemp_eff_w1$wave <- "1"
contemp_eff_w2$wave <- "2"
```

```
temporal_effects_w1 %>%
  full_join(temporal_effects_w2) %>%
  mutate(wave2 = mapvalues(wave, c("1", "2"), c("1", "2"))) %>%
  ggplot(aes(x = V1V2, y = fixed)) +
    geom_line(aes(group = wave2, color = wave2), size = 1.2) +
    scale_color_manual(values = c("orchid", "black")) +
    #geom_point(size = .5) +
    ylim(-.2,.2) +
    geom_hline(aes(yintercept = 0), linetype = "dashed") +
    coord_flip() +
    labs(x = NULL, y = "Edge Weights", color = "Wave") +
    theme_bw() +
    theme(axis.text.y = element_text(size = rel(.7), face = "bold"),
          axis.text.x = element_text(face = "bold"),
          axis.title = element_text(face = "bold"),
          legend.title = element_text(face = "bold"),
          panel.grid.major = element_blank(),
          panel.grid.minor.x = element_blank())
```




```
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/pop_temp_EW_ipsative.png", width = 5,  height = 8)

contemp_eff_w1 %>%
  full_join(contemp_eff_w2) %>%
  mutate(wave2 = mapvalues(wave, c("1", "2"), c("1", "2"))) %>%
  ggplot(aes(x = V1V2, y = weight)) +
    geom_line(aes(group = wave2, color = wave2), size = 1.2) +
    scale_color_manual(values = c("orchid", "black")) +
    #geom_point(size = .5) +
    ylim(-.65,.65) +
    geom_hline(aes(yintercept = 0), linetype = "dashed") +
    coord_flip() +
    labs(x = NULL, y = "Edge Weights", color = "Wave") +
```

```
    theme_bw() +
    theme(axis.text.y = element_text(size = rel(1), face = "bold"),
          axis.text.x = element_text(face = "bold"),
          axis.title = element_text(face = "bold"),
          legend.title = element_text(face = "bold"),
          panel.grid.major = element_blank(),
          panel.grid.minor.x = element_blank())
```



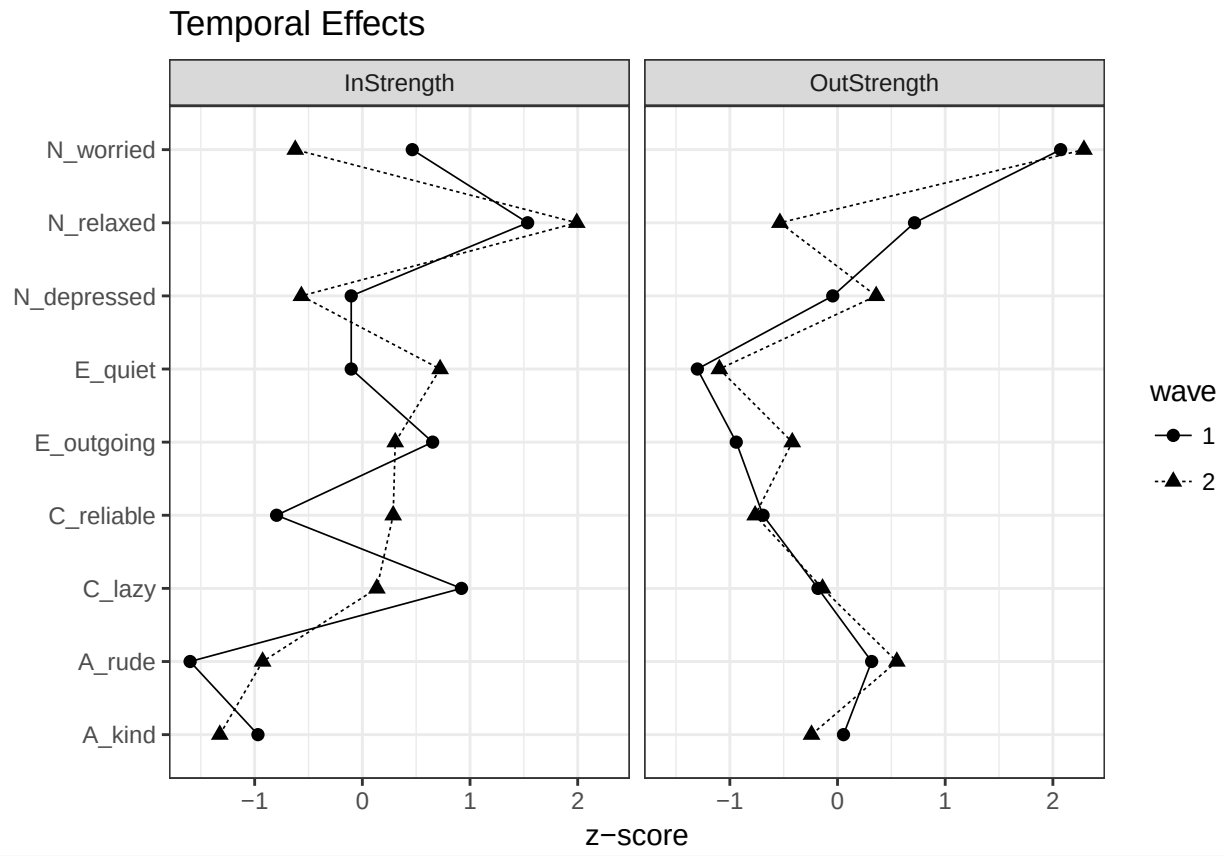

```
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/pop_contemp_EW_ipsative.png", width = 5,  height = 8)
```

```
#temporal
#raw
temporal_centrality_w1_raw      <- centrality_auto(temporal_effects_w1[,c(1,2,4)])
temporal_centrality_w2_raw      <- centrality_auto(temporal_effects_w2[,c(1,2,4)])

#contemporaneous
#raw
contemporaneous_centrality_w1_raw      <- centrality_auto(contemp_effects_w1)
contemporaneous_centrality_w2_raw      <- centrality_auto(contemp_effects_w2)
save(temporal_centrality_w1_raw, temporal_centrality_w2_raw,
     contemporaneous_centrality_w1_raw, contemporaneous_centrality_w2_raw,
     file = "~/Box Sync/network/PAIRS/manuscript/R/centrality_pop.RData")
```

3.5.1.1 Temporal

```
# save centrality results into data frame #
# temporal #
temp_cent <- temporal_centrality_w1_raw$node.centrality %>%
  mutate(wave = "1", type = "Temporal", var = varnames[6:14])
temp_cent <- temporal_centrality_w2_raw$node.centrality %>%
  mutate(wave = "2", type = "Temporal", var = varnames[6:14]) %>%
  full_join(temp_cent)
# contemporaneous #
```

```r
contemp_cent <- contemporaneous_centrality_w1_raw$node.centrality %>%
  mutate(wave = "1", type = "Contemporaneous", var = varnames[6:14])
contemp_cent <- contemporaneous_centrality_w2_raw$node.centrality %>%
  mutate(wave = "2", type = "Contemporaneous", var = varnames[6:14]) %>%
  full_join(contemp_cent)

# wrangle to wide format #
temp_cent_wide <- temp_cent %>%
  gather(key = Measure, value = Centrality, Betweenness: OutStrength) %>%
  spread(key = var, value = Centrality) %>%
  select(Measure, wave, everything(), -type) %>%
  arrange(Measure, wave)
# wrangle to wide format #
contemp_cent_wide <- contemp_cent %>%
  gather(key = Measure, value = Centrality, Betweenness: Strength) %>%
  spread(key = var, value = Centrality) %>%
  select(Measure, wave,  everything(), -type) %>%
  arrange(Measure, wave)


stargazer(temp_cent_wide, digits = 2, summary = F, font.size = "footnotesize",
          title = "Centrality Table of Temporal Fixed Effects", rownames = FALSE,
          column.sep.width = "0pt")

stargazer(contemp_cent_wide, digits = 2, summary = F, font.size = "footnotesize",
          title = "Centrality Table of Contemporaneous Fixed Effects", rownames = FALSE,
          column.sep.width = "0pt")
# wrangle to long form and calculate standardized measures #
temp_cent_long <- temp_cent %>%
  gather(key = Measure, value = Centrality, Betweenness:OutStrength) %>%
  group_by(wave, Measure) %>%
  mutate(z = scale(Centrality))

# plot Centrality Indices #
temp_cent_long %>%
  filter(!(Measure == "Closeness") & !(Measure == "Betweenness")) %>%
  ggplot(aes(x = var, y = z, group = wave))+
    geom_line(aes(linetype = wave), color = "black", size = .3) +
    geom_point(aes(shape = wave), size = 2) +
    labs(x = NULL, y = "z-score", title = "Temporal Effects") +
    coord_flip() +
    facet_grid(.~Measure) +
    theme_bw()
```

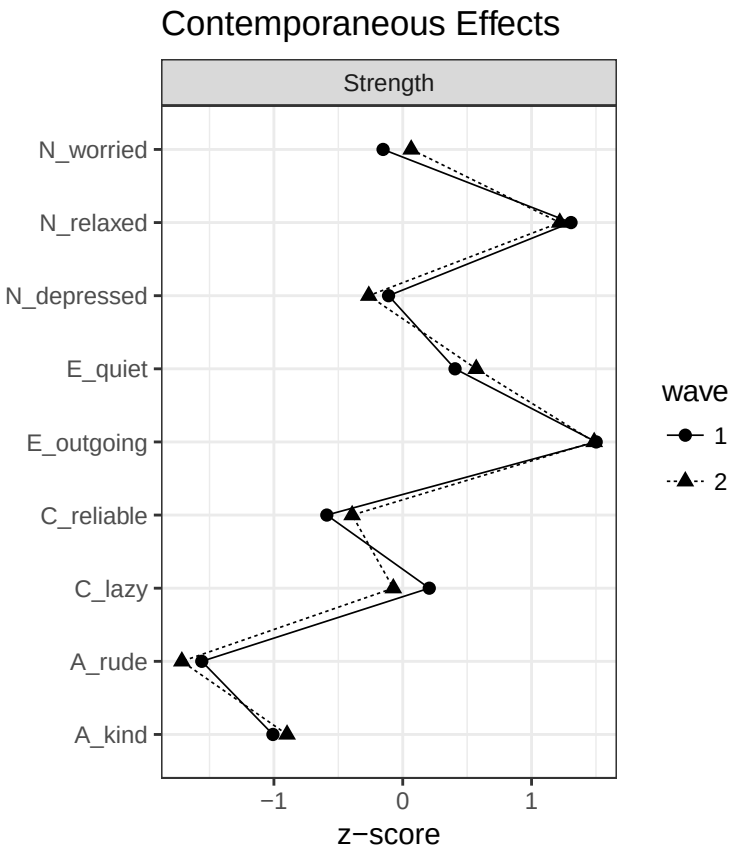



```
# plot wave 1 centrality indices #
w1_cent <- temp_cent_long %>%
  filter(wave == "1" & (Measure == "InStrength" | Measure == "OutStrength")) %>%
ggplot(aes(x = var, y = z, group = wave))+
  geom_line(aes(linetype = wave), color = "black", size = .3) +
  geom_point(aes(shape = wave), size = 2) +
  geom_hline(aes(yintercept = 0), linetype = "dashed") +
  labs(x = NULL, y = "z-score",
       linetype = "Wave", shape = "Wave") +
  coord_flip() +
  facet_grid(.~Measure) +
  theme_bw() +
  theme(axis.text = element_text(face = "bold"),
        axis.title = element_text(face = "bold"),
        legend.title = element_text(face = "bold"))
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/pop_temporal_centrality_w1.png",
       width = 6, height = 5)

# plot wave 2 centrality indices #
w2_cent <- temp_cent_long %>%
  filter(wave == "2" & (Measure == "InStrength" | Measure == "OutStrength")) %>%
ggplot(aes(x = var, y = z, group = wave)) +
  geom_line(aes(linetype = wave), linetype = "dashed", color = "black", size = 1) +
  geom_point( shape = 15, aes(shape = wave),size = 2) +
  geom_point(aes(shape = wave)) +
  geom_hline(aes(yintercept = 0), linetype = "dashed") +
  labs(x = NULL, y = "z-score",
       linetype = "Wave", shape = "Wave") +
  coord_flip() +
  facet_grid(.~Measure) +
  theme_bw()+
  theme(axis.text = element_text(face = "bold"),
        axis.title = element_text(face = "bold"),
        legend.title = element_text(face = "bold"))
```

```
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/pop_temporal_centrality_w2.png",
       width = 6, height = 5)
```

3.5.1.2 Contemporaneous

```
# Contemporaneous #
# wrangle to long format and calculate standardized indices #
contemp_cent_long <- contemp_cent %>%
  gather(key = Measure, value = Centrality, Betweenness:Strength) %>%
  group_by(wave, Measure) %>%
  mutate(z = scale(Centrality))

# plot centrality indices #
contemp_cent_long %>%
  filter(Measure == "Strength") %>%
  ggplot(aes(x = var, y = z, group = wave))+
  geom_line(aes(linetype = wave), color = "black", size = .3) +
  geom_point(aes(shape = wave), size = 2) +
  labs(x = NULL, y = "z-score", title = "Contemporaneous Effects") +
  coord_flip() +
  facet_grid(.~Measure) +
  theme_bw()
```




```
# plot wave 1 centrality indices #
w1_cent <- contemp_cent_long %>%
  filter(wave == "1" & Measure == "Strength") %>%
  ggplot(aes(x = var, y = z, group = wave))+
```

```
  geom_line(aes(linetype = wave),  color = "black", size = .8) +
  geom_point( aes(shape = wave),size = 2) +
  labs(x = NULL, y = "z-score",
       linetype = "Wave", shape = "Wave") +
  coord_flip() +
  facet_grid(.~Measure) +
  theme_bw() +
  theme(axis.text = element_text(face = "bold"),
        axis.title = element_text(face = "bold"),
        legend.title = element_text(face = "bold"))
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/pop_contemp_centrality_w1.png",
       width = 4, height = 5)

# plot wave 1 centrality indices #
w2_cent <- contemp_cent_long %>%
  filter(wave == "2" & Measure == "Strength") %>%
  ggplot(aes(x = var, y = z, group = wave))+
  geom_line(aes(linetype = wave), linetype = "dashed", color = "black", size = 1) +
  geom_point( shape = 15, aes(shape = wave),size = 2) +
  geom_point(aes(shape = wave)) +
  labs(x = NULL, y = "z-score", title = "Contemporaneous Effects") +
  coord_flip() +
  facet_grid(.~Measure) +
  theme_bw()
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/pop_contemp_centrality_w2.png",
       width = 4, height = 5)

# create combined data frame of temporal and contemporaneous effects #
combined_centrality <- temp_cent_long %>% full_join(contemp_cent_long) %>% ungroup()
```

3.5.2 Centrality Edge Correlations

One simple way to assess the centrality or importance of a node in a network over time is to correlate centrality indices across the waves. Below, we calculate profile correlations of node centrality across the waves for 4 (temporal) or 3 (contemporaneous) measures of centrality.

3.5.2.1 Temporal & Contemporaneous

```
# create simple function to correlate edges across waves #
cor_fun <- function(x){cor(x$`1`, x$`2`, use = "pairwise")}

# calculate profile correlations across the waves #
pop_cors <- tbl_df(combined_centrality) %>%
  select(-z) %>%
  spread(key = wave, value = Centrality) %>%
  group_by(type, Measure) %>%
  nest() %>%
  mutate(r = map(data, cor_fun)) %>%
  unnest(r, .drop = T) %>%
  spread(key = Measure, value = r)
```

4 Idiographic Networks

Although mlVAR includes both population and subject level effects, it represents subject level effects as deviations from population effects rather than exmaning unique subject-level patterns. To assess such unique effects, below we construct individual networks for all subjects at each wave.

4.1 Clean Data and Prepare to Run Models

```
# get subjects
subs_w1  <- as.character(unique(w1_com$SID))
subs_w2 <- as.character(unique(w2_com$SID))

# find subjects' SDs of responses
# need to flat subjects with SD's = 0
w1_test <- w1_com[,c(1,6:14)] %>%
  group_by(SID) %>%
  mutate_each(funs(sd = sd(., na.rm = TRUE)))
w2_test <- w2_com[,c(1,6:14)] %>%
  group_by(SID) %>%
  mutate_each(funs(sd = sd(., na.rm = TRUE)))

# loop through data set and jitter where individual SD's = 0 #
for(i in 1:9){
  for(k in 1:dim(w1_test)[1]){
    if(w1_test[k, i + 10] == 0){
      w1_test[k, i + 1] <-
        jitter(as.numeric(w1_test[k, i + 1]), amount = runif(1, 0, .05))
    }
  }
}
# loop through data set and jitter where individual SD's = 0 #
for(i in 1:9){
  for(k in 1:dim(w2_test)[1]){
    if(w2_test[k, i + 10] == 0){
      w2_test[k, i + 1] <-
        jitter(as.numeric(w2_test[k, i + 1]), amount = runif(1, 0, .05))
    }
  }
}
```

4.1.1 Run Models

For idiographic networks, we estimate a Gaussian graphical model (GGM) variation of the vector autoregressive model (VAR), which estimates a partial correlation network in which correlations represent the correlation between variables after conditioning on all other variables. These models are regularized using a variant of the least absolute shrinkage and selection operator (LASSO), graphical LASSO (glasso). In addition, glasso includes a tuning parameter that can be set to control the sparsity of the network. Different values of the parameter can be chosen to optimize prediction accuracy to minimize an information criterion, such as the Bayesian information criterion (BIC) or the extended BIC (EBIC; Chen & Chen, 2008).

```
# load final data sets with jitter
w1_test <- read.csv("~/Box Sync/network/PAIRS/Wave 1/esm_w1_jittered.csv")
w2_test <- read.csv("~/Box Sync/network/PAIRS/Wave 4 + 7/esm_w47_jittered.csv")

# save subjects to a vector
subs2_w1 <- as.character(unique(w1_test$SID))
subs2_w2 <- as.character(unique(w2_test$SID))

# create function to run graphicalVAR for each subject with scaled tuning parameters
gVAR_fun <- function(x){
  n <- dim(x)[1]
```

```
  gamma <- .005 * n
  lambda <- .003 * n
  fit <-
    graphicalVAR_test(x, gamma = gamma, maxit.in = 1000, maxit.out = 1000,
                      lambda_beta = lambda, lambda_kappa = lambda,
                      verbose = F, scale = F)
  return(fit)
}

# calculate networks for each subject for wave 1 #
gVAR_fit_w1 <- w1_test %>%
  select(-contains("sd")) %>%
  group_by(SID) %>%
  nest() %>%
  mutate(gVAR_fit = map(data, possibly(gVAR_fun, NA_real_)))
save(gVAR_fit_w1, file = "~/Box Sync/network/PAIRS/Wave 1/graphicalVAR_allSubs.RData")

# calculate networks for each subject for wave 2 #
gVAR_fit_w2 <- w2_test %>%
  select(-contains("sd")) %>%
  group_by(SID) %>%
  nest() %>%
  mutate(gVAR_fit = map(data, possibly(gVAR_fun, NA_real_)))
save(gVAR_fit_w2, file = "~/Box Sync/network/PAIRS/Wave 4 + 7/graphicalVAR_allSubs.RData")
```

4.2 Extract Results and Save Into Data Frames

4.2.1 Temporal: Partial Directed Correlations

```
# load idiographic networks #
load("~/Box Sync/network/PAIRS/Wave 1/graphicalVAR_allSubs.RData")
load("~/Box Sync/network/PAIRS/Wave 4 + 7/graphicalVAR_allSubs.RData")

# create simple function to extract temporal networks #
# and save them to a formatted long format data frame #
beta_fun <- function(fit, SID){
  PDC <- fit$PDC
  from <- row.names(PDC)
  PDC.long <- tbl_df(PDC) %>%
    mutate(from = from,
           type = "Temporal") %>%
    gather(key = to, value = weight, A_rude:N_worried)
}

# create a simple function to extract matrix of contemporaneous effects #
kappa_mat_fun <- function(fit){fit$PCC}

# create simple function to extract contemporaneous networks #
# and save them to a formatted long format data frame #
kappa_long_fun <- function(fit){
  PCC <- fit$PCC
  PCC <- PCC[,order(colnames(PCC))]
  PCC <- PCC[order(rownames(PCC)),]
  PCC[lower.tri(PCC, diag = T)] <- NA
  vars <- rownames(PCC)
  PCC.long <- tbl_df(PCC) %>%
    mutate(Var1 = vars,
           type = "Contemporaneous") %>%
    gather(key = Var2, value = weight, A_rude:N_worried) %>%
    filter(!is.na(weight)) %>%
    unite(var, Var1, Var2, sep = ".", remove = F)
}

# run extraction functions on each idiographic network #
```

```
gVAR_fit_w1 <- gVAR_fit_w1 %>%
  filter(!is.na(gVAR_fit)) %>%
  mutate(wave = "1",
         beta = map2(gVAR_fit, SID, beta_fun),
         kappa_mat = map(gVAR_fit, kappa_mat_fun),
         kappa = map(gVAR_fit, kappa_long_fun))

# run extraction functions on each idiographic network #
gVAR_fit_w2 <- gVAR_fit_w2 %>%
  filter(!is.na(gVAR_fit)) %>%
  mutate(wave = "2",
         beta = map2(gVAR_fit, SID, beta_fun),
         kappa_mat = map(gVAR_fit, kappa_mat_fun),
         kappa = map(gVAR_fit, kappa_long_fun))

# unnest temporal effects and merge waves #
beta_long <- gVAR_fit_w1 %>%
  unnest(beta)
beta_long <- gVAR_fit_w2 %>%
  unnest(beta) %>%
  full_join(beta_long)

# get SIDs from models
SID_w1  <- as.character(unique(gVAR_fit_w1$SID))
SID_w2 <- as.character(unique(gVAR_fit_w2$SID))

#get variable names from models
varnames <- unique(beta_long$from)

# find subjects in both waves
w1w2_subs <- unique(SID_w1)[unique(SID_w1) %in% unique(SID_w2)]
```

4.2.2 Contemporaneous: Partial Contemporaneous Correlations

```
kappa_w1  <- gVAR_fit_w1$kappa
kappa_w2 <- gVAR_fit_w2$kappa

# unnest contemporaneous effects and merge waves #
kappa_long <- gVAR_fit_w1 %>%
  unnest(kappa)
kappa_long <- gVAR_fit_w2 %>%
  unnest(kappa) %>%
  full_join(kappa_long)
```

4.3 Stability

4.3.1 Rank-Order

One way to conceptualize edge stability is to examine the stability of specific edges across all subjects over time. That is, are some edges more stable than others?

4.3.1.1 Temporal

```
# create simple function to calculate rank-order correlations across waves #
cor_fun <- function(x){
    results <- cor(x$`1`, x$`2`, use = "pairwise", method = "spearman")
}

# create rank variable for each edge #
beta_long <- beta_long %>%
  dplyr::filter(SID %in% w1w2_subs) %>%
```

```
  group_by(wave, from, to) %>%
  mutate(rank = dense_rank(desc(weight))) %>%
  ungroup() %>%
  mutate(x = ifelse(wave == "1", -1, 1))

# calculate rank-order correlations across waves for each edge #
beta_cors_long <- beta_long %>%
  select(-x, -weight) %>%
  spread(key = wave, value = rank) %>%
  group_by(from, to) %>%
  nest() %>%
  mutate(r = map(data, cor_fun))

# unnest rank-order correlations #
beta_cors_long.df <- beta_cors_long %>% unnest(r, .drop = T)

# average rank-order correlation across all variables #
kable(beta_cors_long.df %>%
  summarize(mean = meanSD_r2z2r(r)[1],
            sd = meanSD_r2z2r(r)[2]),
  caption = "Average Idiographic Partial Contemporaneous Correlations")
```

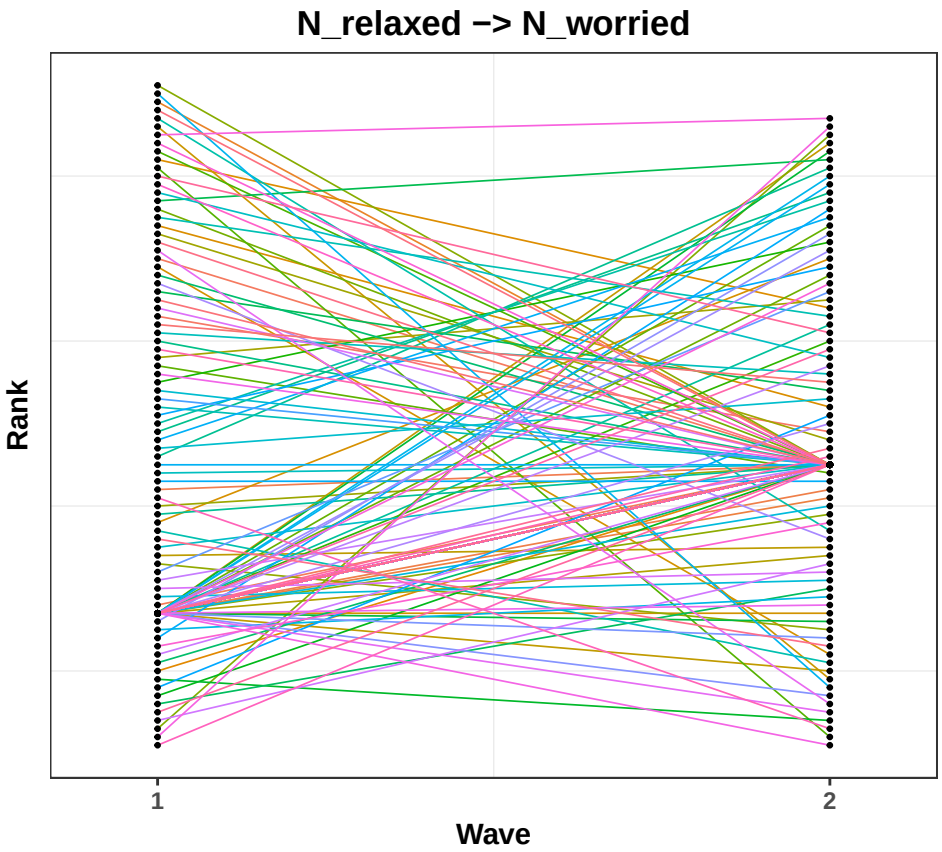
Table 5: Average Idiographic Partial Contemporaneous Correlations

mean	sd
-0.0011248	0.0917313

```
# create heatmap correlation matrix of rank-order correlations #
PDC_cors_heatmap <- beta_cors_long.df %>%
  mutate(measure = "Rank-Order") %>%
  filter(measure == "Rank-Order") %>%
  mutate(from = factor(from, levels = sort(unique(from))),
         to = factor(to, levels = rev(sort(unique(to))))) %>%
  ggplot(aes(x = from, y = to, fill = r)) +
    geom_tile() +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
   midpoint = 0, limit = c(-.5,.5), space = "Lab",
   name="Rank-Order\nCorrelations") +
  geom_text( aes(label = round(r,2))) +
  facet_grid(.~measure) +
  labs(x = "From", y = "To") +#, title = "Temporal Edge Stability Correlations") +
  theme_classic() +
  theme(axis.text = element_text(face = "bold"),
        axis.title = element_text(face = "bold"),
        strip.text = element_text(face = "bold"),
        legend.title = element_text(face = "bold"))
ggsave(file = "~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/PDC_Edge_Cors_heatmap.png", width = 9, height = 6)
```

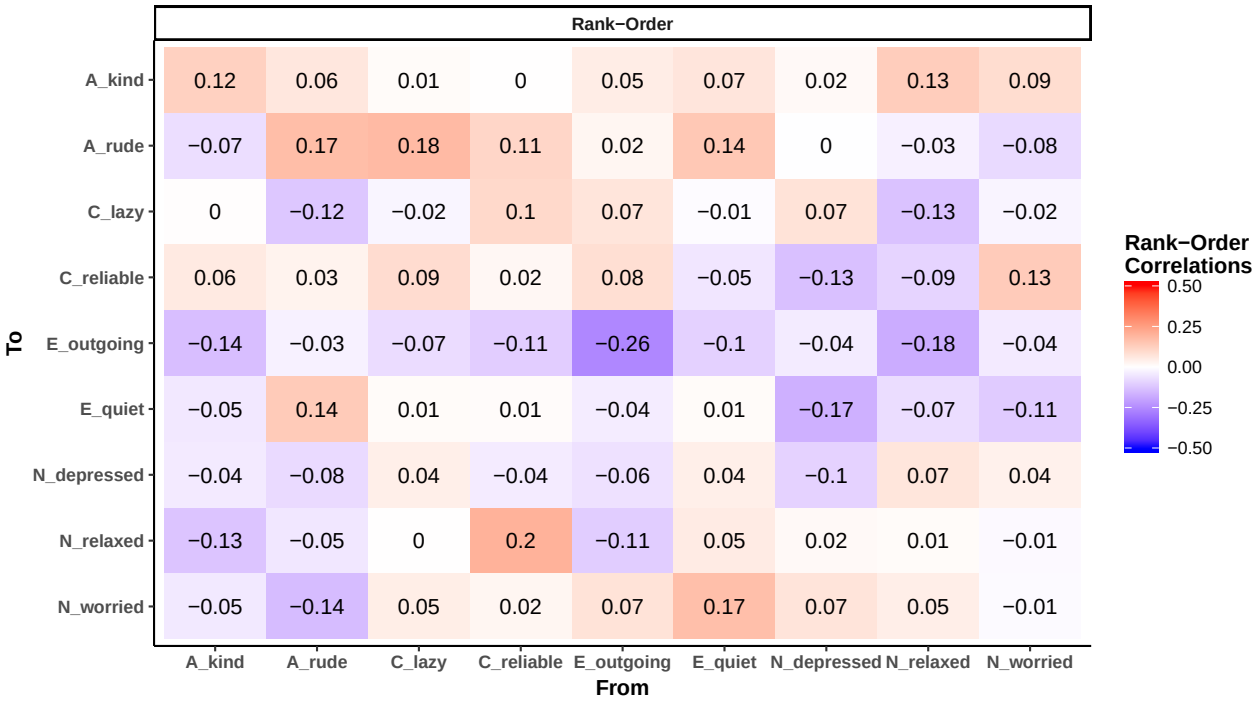
```
# create vector with all edges #
vars <- rep(NA, 81)
k <- 1
for(i in varnames[1:9]){for(j in varnames[1:9]){vars[k] <- paste(i, j, sep = "."); k <- k+1}}

# create a function for plotting ranks #
draw.a.plot <-  function(x){
  var <- separate(as.data.frame(x), x, into = c("from", "to"), sep = "[.]")
  data <- beta_long %>%
    filter(from == var$from & to == var$to &
             SID %in% w1w2_subs) %>%
    mutate(V1V2 = paste(from, to, sep = " -> "),
           wave = recode(wave, `1` = "1", `2` = "2"))
  ggplot(data, aes(x = as.numeric(wave), y = rank, group = factor(SID))) +
    geom_line(aes(color = factor(SID)), size = .3) +
    scale_x_continuous(limits = c(.9,2.1), breaks = 1:2) +
    #scale_color_manual(values = c("indianred1", "black")) +
    geom_point(size = .5) +
```

```
    labs(x = "Wave", y = "Rank", color = "Subject",
         title = data$V1V2) +
    theme_bw() +
    theme(axis.text.y = element_blank(),
          axis.ticks.y = element_blank(),
          panel.grid.major = element_blank(),
          plot.title = element_text(hjust = .5, face = "bold"),
          axis.title = element_text(face = "bold"),
          axis.text = element_text(face = "bold"),
          legend.title = element_text(face = "bold"),
          legend.position = "none")
}

# plot sample edge ranks #
draw.a.plot("N_relaxed.N_worried")
```




```
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/PDC_vars_denserank.png", width = 5, height = 8)

#set up function to loop through the draw.a.plot() function
loop.animate <- function() {
    lapply(vars, function(i) {
        print(draw.a.plot(i))
    })
}

# create GIF of all edges
# saveGIF(loop.animate(), interval = .5, movie.name="PDC_vars_denserank.gif", ani.width = 250, ani.height = 400,
#         imgdir = "~/Box Sync/network/PAIRS/Brown Bag 3.31/")
```

```
PDC_cors_heatmap
```



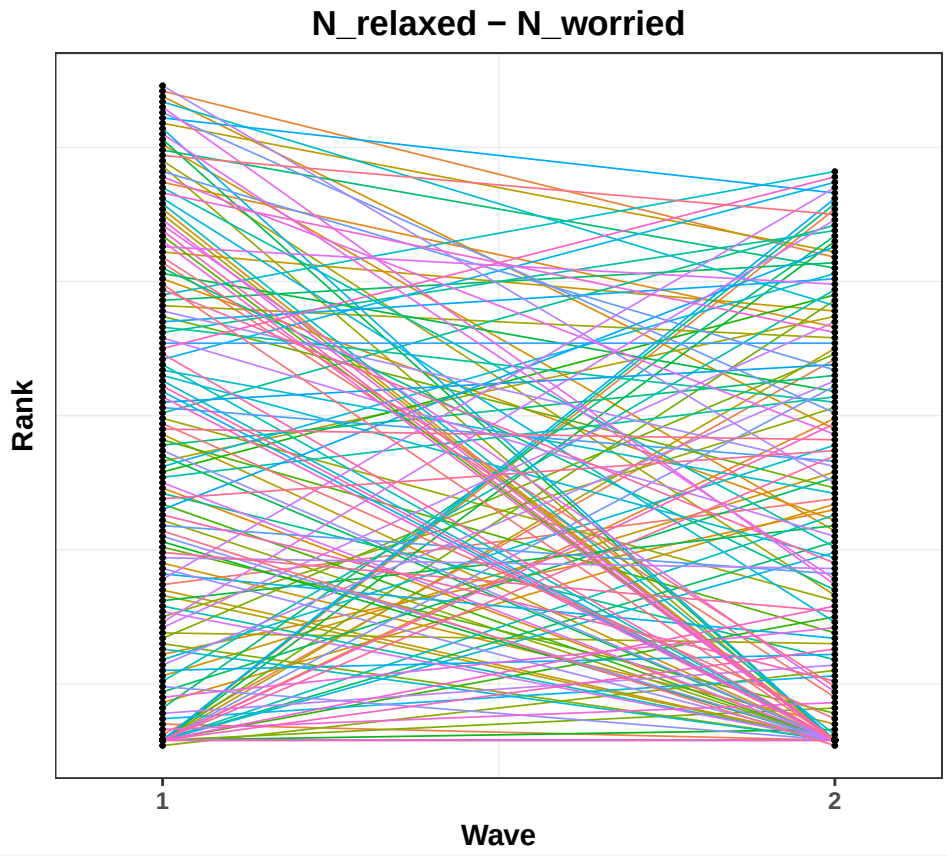
4.3.1.2 Contemporaneous

```
# create a simple function to calculate rank order correlations #
cor_fun <- function(x){
    results <- cor(x$`1`, x$`2`, use = "pairwise", method = "spearman")
}

# create rank variable #
kappa_long <- kappa_long %>%
  dplyr::filter(SID %in% w1w2_subs) %>%
  group_by(wave, Var1, Var2) %>%
  mutate(rank = min_rank(desc(weight))) %>%
  ungroup() %>%
  mutate(x = ifelse(wave == "1", -1, 1))

# calculate rank order correlations #
kappa_cors_long <- kappa_long %>%
  select(-x, -weight) %>%
  spread(key = wave, value = rank) %>%
  group_by(Var1, Var2) %>%
  nest() %>%
  mutate(r = map(data, cor_fun))

# unnest rank order correlations #
kappa_cors_long.df <- kappa_cors_long %>% unnest(r, .drop = T) %>%
  mutate(measure = "Rank-Order", "Absolute Change")

# table of average Contemporaneous Rank-Order Correlations #
kable(kappa_cors_long.df %>%
  group_by(measure) %>%
  summarize(mean = meanSD_r2z2r(r)[1],
            sd = meanSD_r2z2r(r)[2]),
  caption = "Average Idiographic Partial Contemporaneous Rank-Order Correlations")
```

Table 6: Average Idiographic Partial Contemporaneous Rank-Order Correlations

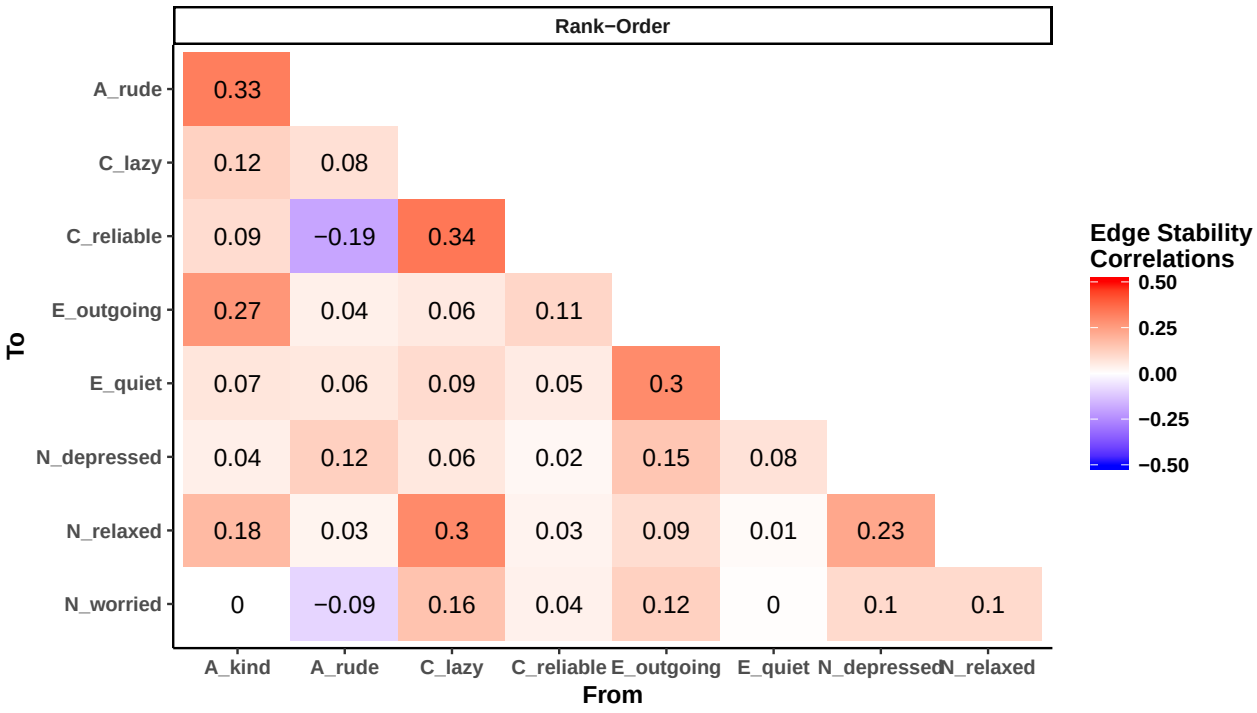
measure	mean	sd
Rank-Order	0.1006456	0.1137547

```r
# plot heatmap of contemporaneous rank-order correlations #
PCC_cors_heatmap <- kappa_cors_long.df %>%
  filter(measure == "Rank-Order") %>%
  mutate(Var2 = factor(Var2, levels = rev(sort(unique(Var2))))) %>%
  ggplot(aes(x = Var1, y = Var2, fill = r)) +
    geom_raster() +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
   midpoint = 0, limit = c(-.5,.5), space = "Lab",
   name="Edge Stability\nCorrelations") +
  geom_text(aes(label = round(r,2))) +
  #geom_text(data = filter(PCC_cors_long.df, measure == "Absolute Change"), aes(label = round(sd,2))) +
  facet_grid(.~measure) +
  labs(x = "From", y = "To") +
  theme_classic() +
  theme(strip.text = element_text(face = "bold"),
        axis.text = element_text(face = "bold"),
        axis.title = element_text(face = "bold"),
        legend.title = element_text(face = "bold"),
        legend.text = element_text(face = "bold"))
ggsave(file = "~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/PCC_Edge_Cors_heatmap.png", width = 9, height = 6)

# create a function for plotting contemporaneous ranks #
draw.a.plot <-  function(x){
  var <- separate(as.data.frame(x), x, into = c("Var1", "Var2"), sep = "[ ]")
  data <- kappa_long %>%
    filter(Var1 == var$Var1 & Var2 == var$Var2 &
             SID %in% w1w2_subs) %>%
    mutate(V1V2 = paste(Var1, Var2, sep = " - ")) %>%
    group_by(wave, V1V2) %>%
    mutate(rank = dense_rank(desc(weight)))
  ggplot(data, aes(x = as.numeric(wave), y = rank, group = factor(SID))) +
    geom_line(aes(color = factor(SID)), size = .3) +
    scale_x_continuous(limits = c(.9,2.1), breaks = 1:2) +
    #scale_color_manual(values = c("indianred1", "black")) +
    geom_point(size = .5) +
    labs(x = "Wave", y = "Rank", color = "Subject",
         title = data$V1V2) +
    theme_bw() +
    theme(axis.text.y = element_blank(),
          axis.ticks.y = element_blank(),
          panel.grid.major = element_blank(),
          plot.title = element_text(hjust = .5, face = "bold"),
          axis.title = element_text(face = "bold"),
          axis.text = element_text(face = "bold"),
          legend.title = element_text(face = "bold"),
          legend.position = "none")
}

# plot sample edge ranks #
draw.a.plot("N_relaxed N_worried")
```




```
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/PCC_vars_denserank.png", width = 5, height = 8)

#set up function to loop through the draw.a.plot() function
loop.animate <- function() {
    lapply(unique(kappa_long$var), function(i) {
        print(draw.a.plot(i))
    })
}

# saveGIF(loop.animate(), interval = .5, movie.name="PCC_vars_denserank.gif", ani.width = 250, ani.height = 400,
#         imgdir = "~/Box Sync/network/PAIRS/Brown Bag 3.31/")
```

```
PCC_cors_heatmap
```



4.3.2 Ipsative Edge Stability

Another way to conceptualize stability is to assess a single person's stability over time. Below, we take profile correlations of all edges for each subject individually for both temporal and contemporaneous effects. The results are then displayed in a histogram.

```
# calculate profile correlations of temporal effects #
ip_beta_cors <- beta_long %>%
  dplyr::filter(SID %in% w1w2_subs) %>%
  arrange(wave, SID, from, to) %>%
  select(-x, -rank) %>%
  spread(wave, weight) %>%
  group_by(SID) %>%
  summarize(cors = cor(`1`, `2`, use = "pairwise.complete.obs")) %>%
  mutate(type = "PDC")

# calculate profile correlations of contemporaneous effects #
# merge with temporal effects #
ip_cors <- kappa_long %>%
  dplyr::filter(SID %in% w1w2_subs) %>%
  arrange(wave, SID, var) %>%
  select(-x, -rank) %>%
  spread(wave, weight) %>%
  group_by(SID) %>%
  summarize(cors = cor(`1`, `2`, use = "pairwise.complete.obs")) %>%
  mutate(type = "PCC") %>%
  full_join(ip_beta_cors)

# save network type #
ip_cors$type <- factor(ip_cors$type, levels = c("PDC", "PCC"))
ip_cors$type2 <- factor(ifelse(ip_cors$type == "PDC", "Temporal", "Contemporaneous"),
                        levels = c("Temporal", "Contemporaneous"))

# create a table of average profile correlations #
kable(ip_cors %>%
  group_by(type2) %>%
  summarize(mean_cor = meanSD_r2z2r(cors)[1],
            sd_cor = meanSD_r2z2r(cors)[2]), caption = "Average Ipsative Correlations")
```

Table 7: Average Ipsative Correlations

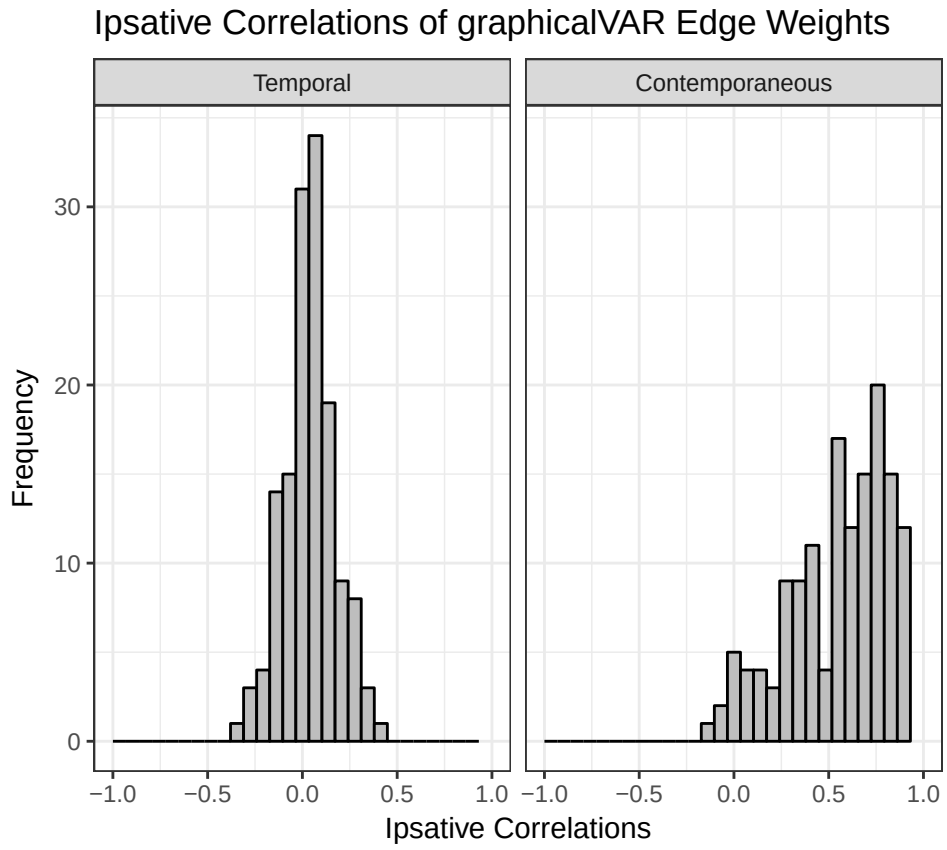
type2	mean_cor	sd_cor
Temporal	0.0377936	0.1380830
Contemporaneous	0.6211319	0.4112329

```
# create table of profile correlations for subject 90 #
kable(ip_cors %>%
  filter(SID == "90"), caption = "Ipsative Correlation for Subject 90")
```

Table 8: Ipsative Correlation for Subject 90

SID	cors	type	type2
90	0.3418463	PCC	Contemporaneous
90	0.1351309	PDC	Temporal

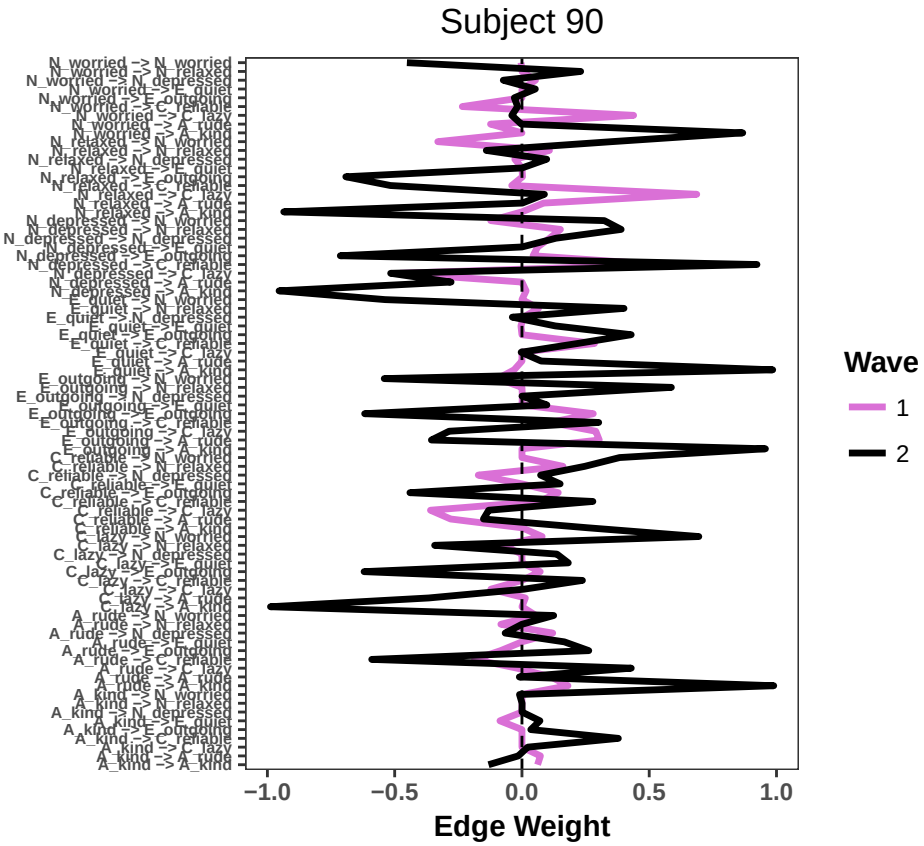
```
# create histogram of profile correlations #
ggplot(ip_cors, aes(x = cors)) +
  geom_histogram(color = "black", fill = "gray") +
  labs(x = "Ipsative Correlations", y = "Frequency",
       title = "Ipsative Correlations of graphicalVAR Edge Weights") +
  xlim(c(-1, 1)) +
  facet_grid(.~type2) +
  theme_bw()
```




```
ggsave("~/Box Sync/network/PAIRS/manuscript/ipsative_cor.png", width  = 6, height = 3)
```

4.3.2.1 Temporal

```r
# create histogram of temporal profile correlations #
PDC_ip_cors <- ip_cors %>%
  filter(type2 == "Temporal") %>%
  ggplot(aes(x = cors)) +
  geom_histogram(color = "black", fill = "gray") +
  labs(x = "Ipsative Correlations", y = "Frequency",
       title = "Ipsative Correlations of graphicalVAR Edge Weights") +
  xlim(c(-1, 1)) +
  facet_grid(.~type2) +
  theme_bw() +
  theme(plot.title = element_text(hjust = .5))

# create a function for plotting subjects' profiles of edges #
draw.a.plot <-  function(sub){
  data <- beta_long %>%
    filter(SID == sub) %>%
    mutate(V1V2 = paste(from, to, sep = " -> "))
  ggplot(data, aes(x = V1V2, y = weight, group = wave)) +
    geom_line(aes(color = wave), size = 1.2) +
    #geom_point(size = .5) +
    scale_color_manual(values = c("orchid", "black")) +
    labs(x = NULL, y = "Edge Weight", color = "Wave", title = sprintf("Subject %s", sub)) +
    geom_hline(aes(yintercept = 0), linetype = "dashed") +
    coord_flip() +
    theme_bw() +
    theme(axis.text.y = element_text(size = rel(.7), face = "bold"),
            axis.text.x = element_text(face = "bold"),
            axis.title = element_text(face = "bold"),
            legend.title = element_text(face = "bold"),
            panel.grid.major = element_blank(),
            panel.grid.minor.x = element_blank(),
            plot.title = element_text(hjust = .5))
}

# sample profile plot for subject 90 #
draw.a.plot("90")
```



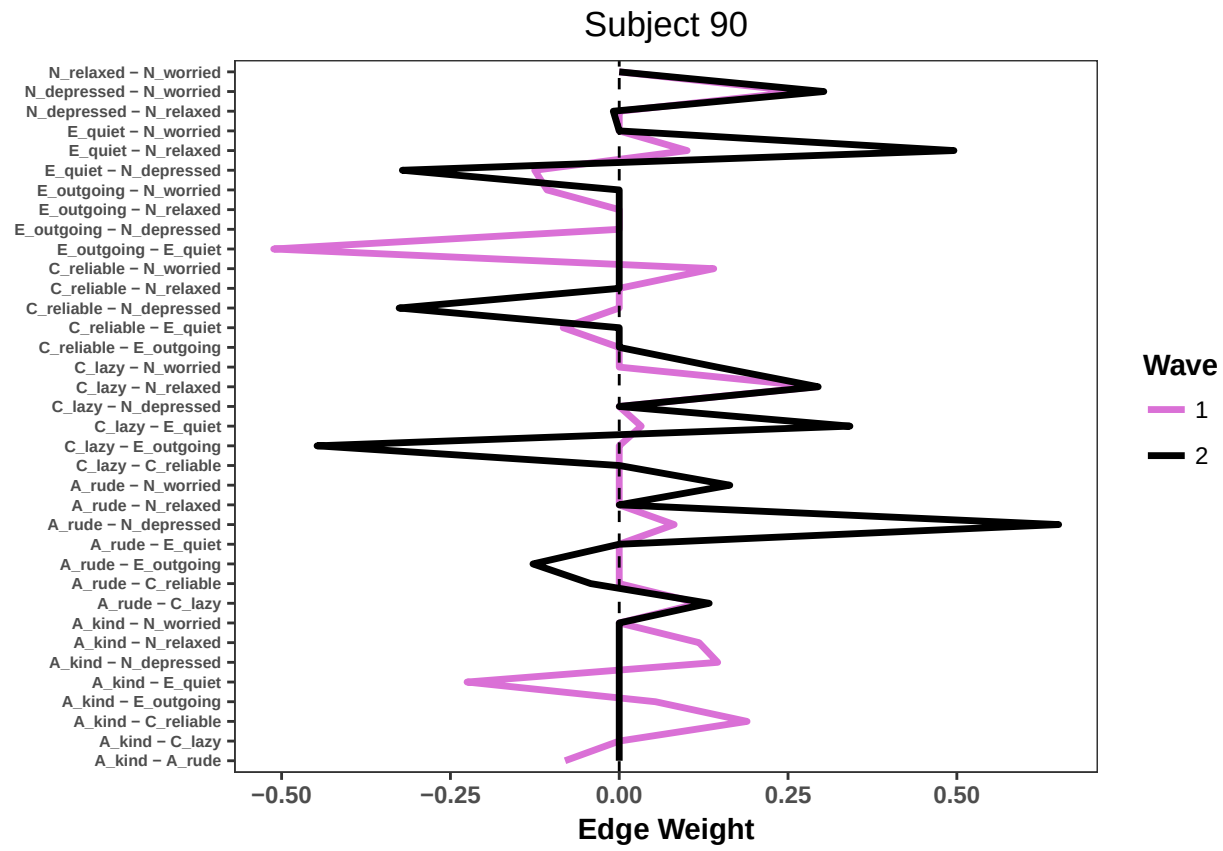

```
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/PDC_EW_ipsative.png", width = 5,  height = 8)


#set up function to loop through the draw.a.plot() function
loop.animate <- function() {
    lapply(as.character(w1w2_subs[10:30]), function(i) {
        print(draw.a.plot(i))
    })
}

saveGIF(loop.animate(), interval = .5, movie.name="PDC_EW_ipsative.gif", ani.width = 300, ani.height = 450,
        imgdir = "~/Box Sync/network/PAIRS/Brown Bag 3.31/")
```

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

4.3.2.2 Contemporaneous

```
# create histogram of contemporaneous profile correlations #
PCC_ip_cors <- ip_cors %>%
  filter(type2 == "Contemporaneous") %>%
  ggplot(aes(x = cors)) +
  geom_histogram(color = "black", fill = "gray") +
  labs(x = "Ipsative Correlations", y = "Frequency",
       title = "Ipsative Correlations of graphicalVAR Edge Weights") +
  xlim(c(-1, 1)) +
  facet_grid(.~type2) +
  theme_bw() +
  theme(plot.title = element_text(hjust = .5))

# create a function to plot subjects' contemporaneous profiles #
draw.a.plot <-  function(z){
  data <- kappa_long %>%
    filter(SID == z) %>%
    mutate(V1V2 = paste(Var1, Var2, sep = " - "))
```

```
  ggplot(data, aes(x = V1V2, y = weight, group = wave)) +
    geom_line(aes(color = wave), size = 1.2) +
    scale_color_manual(values = c("orchid", "black")) +
    geom_hline(aes(yintercept = 0), linetype = "dashed") +
    #geom_point(size = .5) +
    labs(x = NULL, y = "Edge Weight", color = "Wave", title = sprintf("Subject %s", z)) +
    coord_flip() +
    theme_bw() +
    theme(axis.text.y = element_text(size = rel(.7), face = "bold"),
            axis.text.x = element_text(face = "bold"),
            axis.title = element_text(face = "bold"),
            legend.title = element_text(face = "bold"),
            panel.grid.major = element_blank(),
            panel.grid.minor.x = element_blank(),
            plot.title = element_text(hjust = .5))
    }

# sample plot for subject 90 #
draw.a.plot("90")
```




```
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/PCC_EW_ipsative.png", width = 5,  height = 8)

#set up function to loop through the draw.a.plot() function
loop.animate <- function() {
    lapply(as.character(w1w2_subs[10:30]), function(i) {
        print(draw.a.plot(i))
    })
}

saveGIF(loop.animate(), interval = .5, movie.name="PCC_ipsative.gif", ani.width = 300, ani.height = 450,
        imgdir = "~/Box Sync/network/PAIRS/Brown Bag 3.31/")
```

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

4.4 Centrality

As with between-person effects, we can calculate centrality for individuals.

```
# save variable names #
varnames <- varnames[-c(10:13)]

# create function to save both centrality measure and variable
# names to a data frame for temporal networks #
centrality_fun <- function(x) {
  data <- x %>%
    select(from, to, weight) %>%
    mutate(weight = as.numeric(weight))
  centrality <- centrality_auto(data.frame(data))
  df <- data.frame(var = varnames,
                   centrality$node.centrality)
}

# create function to save both centrality measure and variable
# names to a data frame for contemporaneous networks #
kappa_cen_fun <- function(x){
  centrality <- centrality_auto(x)$node.centrality
  centrality$var <- varnames
  return(centrality)
}

# calculate centrality for each subject for each wave and save them to a list #
gVAR_fit_w1 <- gVAR_fit_w1 %>%
  mutate(beta_centrality = map(beta, centrality_fun),
         kappa_centrality = map(kappa_mat, kappa_cen_fun))
gVAR_fit_w2 <- gVAR_fit_w2 %>%
  mutate(beta_centrality = map(beta, centrality_fun),
         kappa_centrality = map(kappa_mat, kappa_cen_fun))

# unnest and merge temporal centrality #
beta_centrality <- gVAR_fit_w1 %>%
  unnest(beta_centrality)
beta_centrality <- gVAR_fit_w2 %>%
  unnest(beta_centrality) %>%
  full_join(beta_centrality) %>%
  select(-Degree) %>%
  mutate(type = "Temporal")

# unnest and merge contemporaneous centrality #
kappa_centrality <- gVAR_fit_w1 %>%
  unnest(kappa_centrality)
kappa_centrality <- gVAR_fit_w2 %>%
  unnest(kappa_centrality) %>%
  full_join(kappa_centrality) %>%
  select(-Degree) %>%
  mutate(type = "Contemporaneous")

save(kappa_centrality, beta_centrality,
     file = "~/Box Sync/network/PAIRS/centrality/graphicalVAR_centrality.RData")
```

4.4.1 Sample Centrality Plot

```
load("~/Box Sync/network/PAIRS/centrality/graphicalVAR_centrality.RData")

# create a function to plot centrality for individual subjects #
centrality_Plot_fun <- function(x, type2){
  dat <- centrality_all  %>%
    filter(SID == x, type == type2) %>%
    arrange(measure, wave)
  dat %>%
    ggplot(aes(x = var, y = z, group = wave))+
```

```
      geom_line(aes(linetype = wave), color = "black", size = .3) +
      geom_point(aes(shape = wave), size = 2) +
      labs(x = NULL, y = "z-score") +
      ylim(-2,2) +
      coord_flip() +
      labs(title = x) +
      facet_wrap(~type + measure, nrow = 1) +
      theme_bw() +
      theme(plot.title = element_text(hjust = .5),
            legend.position = "bottom")
}

# wrangle temporal centrality to long form and standardize #
centrality <- beta_centrality %>%
  gather(key = measure, value = centrality,
         Betweenness, Closeness, InStrength, OutStrength) %>%
  group_by(SID, wave, measure) %>%
  mutate(z = scale(centrality)) %>%
  ungroup()

# wrangle contemporaneous centrality to long form and standardize #
# merge with temporal #
centrality_all <- kappa_centrality %>%
  gather(key = measure, value = centrality,
         Betweenness, Closeness, Strength) %>%
  group_by(SID, wave, measure) %>%
  mutate(z = scale(centrality)) %>%
  ungroup() %>%
  full_join(centrality)

# create a function to plot centrality profiles #
draw.a.plot <-  function(x,y){
  centrality <- centrality_all %>%
    filter(SID == x) %>%
    arrange(measure, wave)
  centrality %>%
    filter(type == y & (measure != "Betweenness" & measure != "Closeness")) %>%
    ggplot(aes(x = var, y = z, group = wave))+
      geom_line(aes(linetype = wave), color = "black", size = .3) +
      geom_point(aes(shape = wave), size = 2) +
      labs(x = NULL, y = "z-score", title = sprintf("Subject %s", x),
           shape = "Wave", linetype = "Wave") +
      ylim(-2.5,2.5) +
      geom_hline(aes(yintercept = 0), linetype = "dashed") +
      coord_flip() +
      facet_wrap(~type + measure, nrow = 1) +
      theme_bw() +
      theme(plot.title = element_text(hjust = .5, face = "bold"),
            axis.text = element_text(face = "bold"),
            axis.title = element_text(face = "bold"),
            legend.title = element_text(face = "bold"))
  }

#set up function to loop through the draw.a.plot() function #
# temporal network centrality #
loop.animate <- function() {
    lapply(w1w2_subs[10:30], function(i) {
        print(draw.a.plot(i, "Temporal"))
    })
}

# saveGIF(loop.animate(), interval = .5, movie.name="PDC_centrality.gif", ani.width = 400, ani.height = 350,
#         imgdir = "~/Box Sync/network/PAIRS/Brown Bag 3.31/")

#set up function to loop through the draw.a.plot() function #
# contemporaneous network centrality #
loop.animate <- function() {
```

```
    lapply(w1w2_subs[10:30], function(i) {
        print(draw.a.plot(i, "Contemporaneous"))
    })
}

# saveGIF(loop.animate(), interval = .5, movie.name="PCC_centrality.gif", ani.width = 275, ani.height = 350,
#         imgdir = "~/Box Sync/network/PAIRS/Brown Bag 3.31/")

# relevel network type variable
centrality$type <- factor(centrality$type, levels = c("Temporal", "Contemporaneous"))

# sample centrality plot for subject 90 #
PDC_centralityPlot_90 <- centrality_all %>%
  filter(type == "Temporal" & (measure != "Betweenness" & measure != "Closeness") & SID == "90") %>%
ggplot(aes(x = var, y = z, group = wave))+
  geom_line(aes(linetype = wave), color = "black", size = .3) +
  geom_point(aes(shape = wave), size = 2) +
  labs(x = NULL, y = "z-score", linetype = "Wave", shape = "Wave") +
  scale_y_continuous(limits = c(-2.5,2.5), breaks = seq(-2.5,2.5,.5)) +
  geom_hline(aes(yintercept = 0), linetype = "dashed") +
  coord_flip() +
  facet_wrap(~type + measure, nrow = 1) +
  theme_bw()+
  theme(axis.text = element_text(face = "bold"),
        axis.title = element_text(face = "bold"),
        legend.title = element_text(face = "bold"))

ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/PDC_centrality_plot_90.png", width = 6, height = 5)

# create contemporaneous centrality plot for subject 90 #
PCC_centralityPlot_90 <- centrality_all %>%
  filter(type == "Contemporaneous" & measure == "Strength" & SID == "90") %>%
ggplot(aes(x = var, y = z, group = wave))+
  geom_line(aes(linetype = wave), color = "black", size = .3) +
  geom_point(aes(shape = wave), size = 2) +
  labs(x = NULL, y = "z-score", linetype = "Wave", shape = "Wave") +
  scale_y_continuous(limits = c(-2.5,2.5), breaks = seq(-2.5,2.5,.5)) +
  geom_hline(aes(yintercept = 0), linetype = "dashed") +
  coord_flip() +
  facet_wrap(~type + measure, nrow = 1) +
  theme_bw() +
  theme(axis.text = element_text(face = "bold"),
        axis.title = element_text(face = "bold"),
        legend.title = element_text(face = "bold"))
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/PCC_centrality_plot_90.png", width = 4, height = 5)
```

4.4.2 Correlations Across Waves

4.4.2.1 Rank Order Correlations

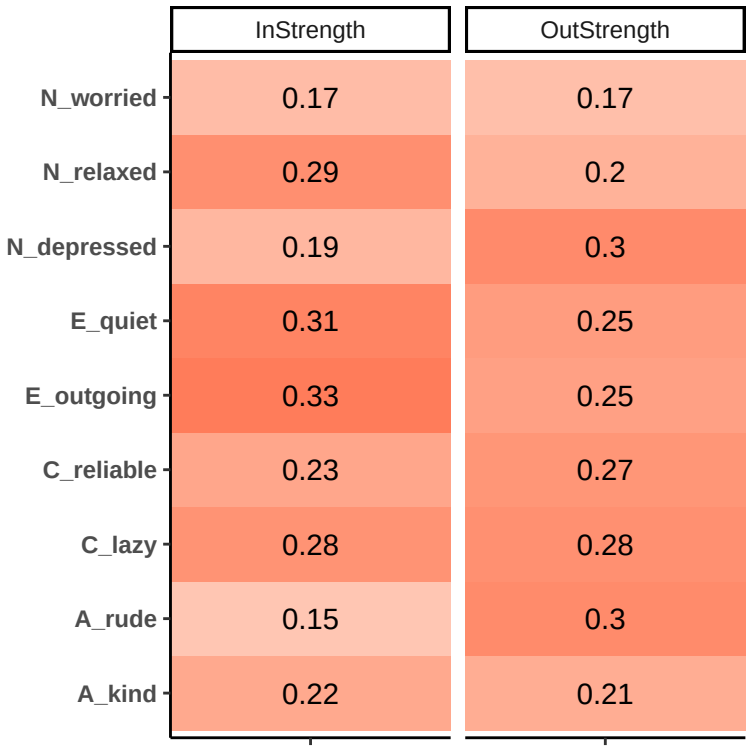
We can also look at the stability of the centrality of specific variables across the waves. That is, do variables maintain relatively the same importance in a subject's network across the waves.

4.4.2.1.1 Temporal

```
# save subjects to a vector #
subs_w1 <- as.character(unique(kappa_centrality$SID[kappa_centrality$wave == "1"]))
subs_w2 <- as.character(unique(kappa_centrality$SID[kappa_centrality$wave == "2"]))
w1w2_subs <- as.character(subs_w1[subs_w1 %in% subs_w2])

# create a simpel function to calculate rank order correlations
cor_fun <- function(x){
  x <- x %>% separate(measure3, into = c("Measure", "cor_type"), remove = F, "[.]")
  cor(x$`1`, x$`2`, use = "pairwise", method = "spearman")
```

```
}

# wrangle centrality to long form and calculate ranks #
beta_centrality_rank <- tbl_df(beta_centrality) %>%
  filter(SID %in% w1w2_subs) %>%
  gather(key = measure, value = Centrality, Betweenness:OutStrength) %>%
  group_by(measure, var, type, wave) %>%
  mutate(rank = min_rank(desc(Centrality))) %>%
  ungroup() %>%
  gather(key = measure2, value = value, Centrality, rank) %>%
  unite(measure3, measure, measure2, remove = F, sep = ".")  %>%
  spread(key = wave, value = value)

# caclulate temporal rank order correlations #
beta_centrality_cor <- beta_centrality_rank %>%
  filter(measure2 == "rank") %>%
  group_by(var, type, measure, measure2) %>%
  nest() %>%
  mutate(r = map(data, cor_fun))

# unnest temporal rank order correlations #
beta_centrality_cor.df <- beta_centrality_cor %>%
  unnest(r, .drop = T)

# create table of average rank order correlations #
kable(beta_centrality_cor.df %>%
  group_by(measure2, measure) %>%
  summarize(mean = meanSD_r2z2r(r)[1],
            sd = meanSD_r2z2r(r)[2]),
  caption = "Average Rank Order Centrality Partial Directed Correlation")
```

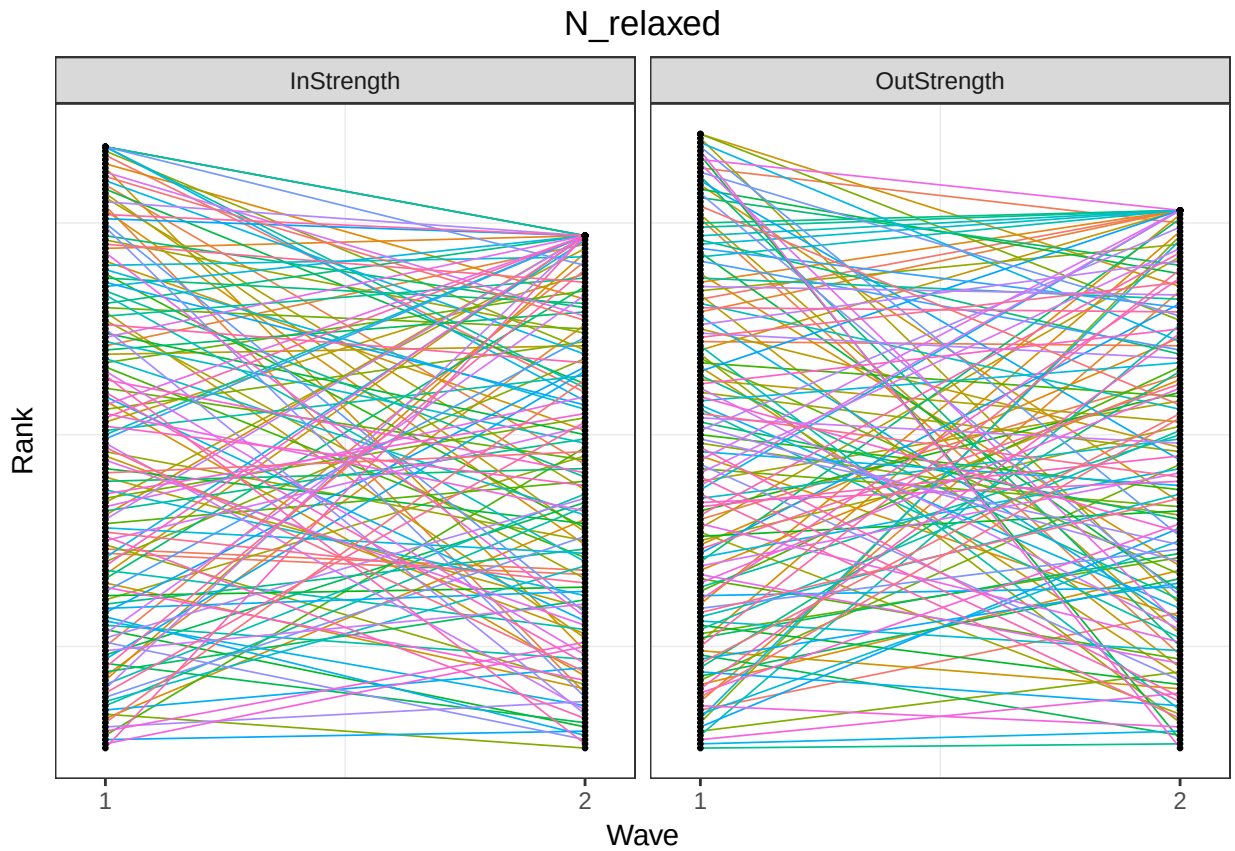
```
# create heatmap of centrality rank order correlations #
beta_centrality_cor.df %>%
  filter(measure == "InStrength" | measure == "OutStrength") %>%
  select(var, measure, measure2, r) %>%
  mutate(measure2 = "Rank-Order") %>%
  ggplot(aes(x = measure, y = var, fill = r)) +
    geom_raster() +
    scale_fill_gradient2(low = "blue", high = "red", mid = "white",
     midpoint = 0, limit = c(-.5,.5), space = "Lab",
     name="Centrality\nCorrelations") +
    geom_text(aes(label = round(r,2))) +
    labs(x = NULL, y = NULL) +
    facet_grid(.~measure, scale = "free") +
    theme_classic() +
    theme(axis.text = element_text(face = "bold"), legend.position = "none",
          axis.text.x = element_blank())
```




```
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/PDC_centrality_var_cor.png",
       width = 4, height = 4)

# create function to plot profiles of temporal centrality #
draw.a.plot <-  function(x){
  data <- beta_centrality %>%
    filter(var == x & SID %in% w1w2_subs) %>%
    select(-Betweenness, -Closeness) %>%
    gather(key = type2, value = Centrality, InStrength, OutStrength) %>%
    spread(key = wave, value = Centrality) %>%
    mutate(diff = `1` - `2`) %>%
    gather(key = wave, value = Centrality, `1`, `2`)
  max_cen <- ceiling(max(data$Centrality, na.rm = T))
  break_size <- max_cen/2
  plot <- ggplot(data, aes(x = SID, y = Centrality, group = wave)) +
    geom_hline(aes(yintercept = 0), linetype = "dashed") +
    geom_line(aes(color = wave), size = 1.2) +
    scale_color_manual(values = c("orchid", "black")) +
    scale_y_continuous(limits = c(-1*max_cen, max_cen),
                       breaks = seq(-max_cen, max_cen, break_size)) +
    #geom_point(size = .5) +
    labs(x = "Subject", y = "Centrality", color = "Wave",
         title = x) +
    coord_flip() +
    theme_bw() +
    facet_grid(.~type2) +
    theme(axis.text.y = element_blank(),
          axis.ticks.y = element_blank(),
          panel.grid.major = element_blank(),
          plot.title = element_text(hjust = .5))
  plot2 <- plot + geom_line(aes(y = diff), color = "blue", size = 1)
  return(list(plot, plot2))
  }

#set up function to loop through the draw.a.plot() function
loop.animate <- function() {
    lapply(varnames[1:9], function(i) {
        print(draw.a.plot(i))
```

```
    })
}

saveGIF(loop.animate(), interval = .5, movie.name="PDC_centrality_var.gif", ani.width = 400, ani.height = 400)
```

```
# create a function to plot ranks of temporal centrality #
draw.a.plot <-  function(x){
  data <- beta_centrality %>%
    filter(var == x & SID %in% w1w2_subs) %>%
    select(-Betweenness, -Closeness) %>%
    gather(key = type2, value = Centrality, InStrength, OutStrength) %>%
    group_by(wave, type2, var) %>%
    mutate(rank = dense_rank(desc(Centrality)))
  ggplot(data, aes(x = as.numeric(wave), y = rank, group = factor(SID))) +
    geom_line(aes(color = factor(SID)), size = .3) +
    geom_point(size = .5) +
    scale_x_continuous(limits = c(.95, 2.05), breaks = c(1,2)) +
    labs(x = "Wave", y = "Rank", title = x) +
    theme_bw() +
    facet_grid(.~type2, scale = "free") +
    theme(panel.grid.major = element_blank(),
          axis.text.y = element_blank(),
          axis.ticks.y = element_blank(),
          plot.title = element_text(hjust = .5),
          legend.position = "none")
}

# plot sample ranks plot for N_relaxed #
draw.a.plot("N_relaxed")
```



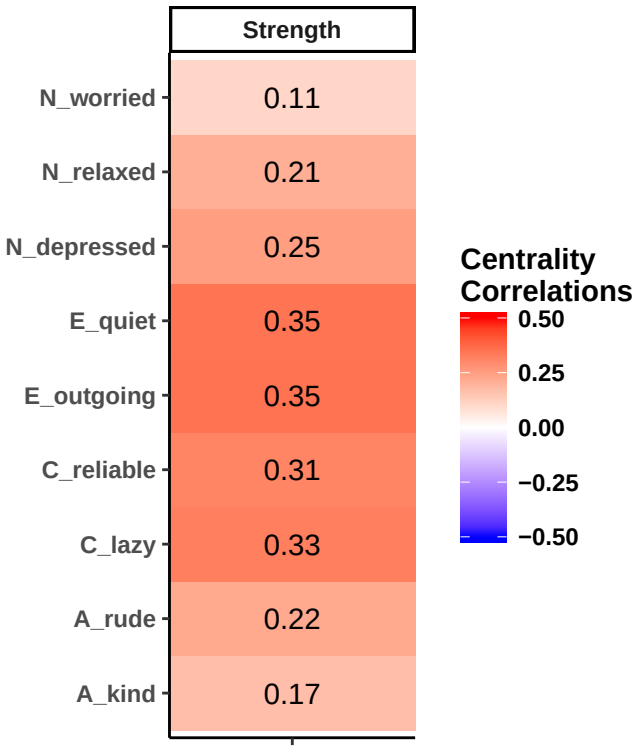

```
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/PDC_centrality_rank.png", width = 5, height = 5)

#set up function to loop through the draw.a.plot() function
loop.animate <- function() {
```

```
    lapply(varnames[1:9], function(i) {
        print(draw.a.plot(i))
    })
}
# saveGIF(loop.animate(), interval = .5, movie.name="PDC_centrality_rank.gif", ani.width = 400, ani.height = 400)
```

4.4.2.1.2 Contemporaneous

```
# create ranks and calculate rank order correlations #
kappa_centrality_cor <- tbl_df(kappa_centrality) %>%
  filter(SID %in% w1w2_subs) %>%
  gather(key = measure, value = Centrality, Betweenness:Strength) %>%
  group_by(measure, var, type, wave) %>%
  mutate(rank = min_rank(desc(Centrality))) %>%
  ungroup() %>%
  gather(key = measure2, value = value, Centrality, rank) %>%
  unite(measure3, measure, measure2, remove = F, sep = ".") %>%
  spread(key = wave, value = value) %>%
  group_by(var, type, measure, measure2) %>%
  nest() %>%
  mutate(r = map(data, cor_fun))

# unnest rank order correlations #
kappa_centrality_cor.df <- kappa_centrality_cor %>%
  filter(measure2 == "rank") %>%
  unnest(r, .drop = T)

# create a table of average contemporaneous rank order correlations #
kable(kappa_centrality_cor.df %>%
  group_by(measure2, measure) %>%
  summarize(mean = meanSD_r2z2r(r)[1],
            sd = meanSD_r2z2r(r)[2]),
  caption = "Average Centrality Partial Contemporaneous Correlation")
```

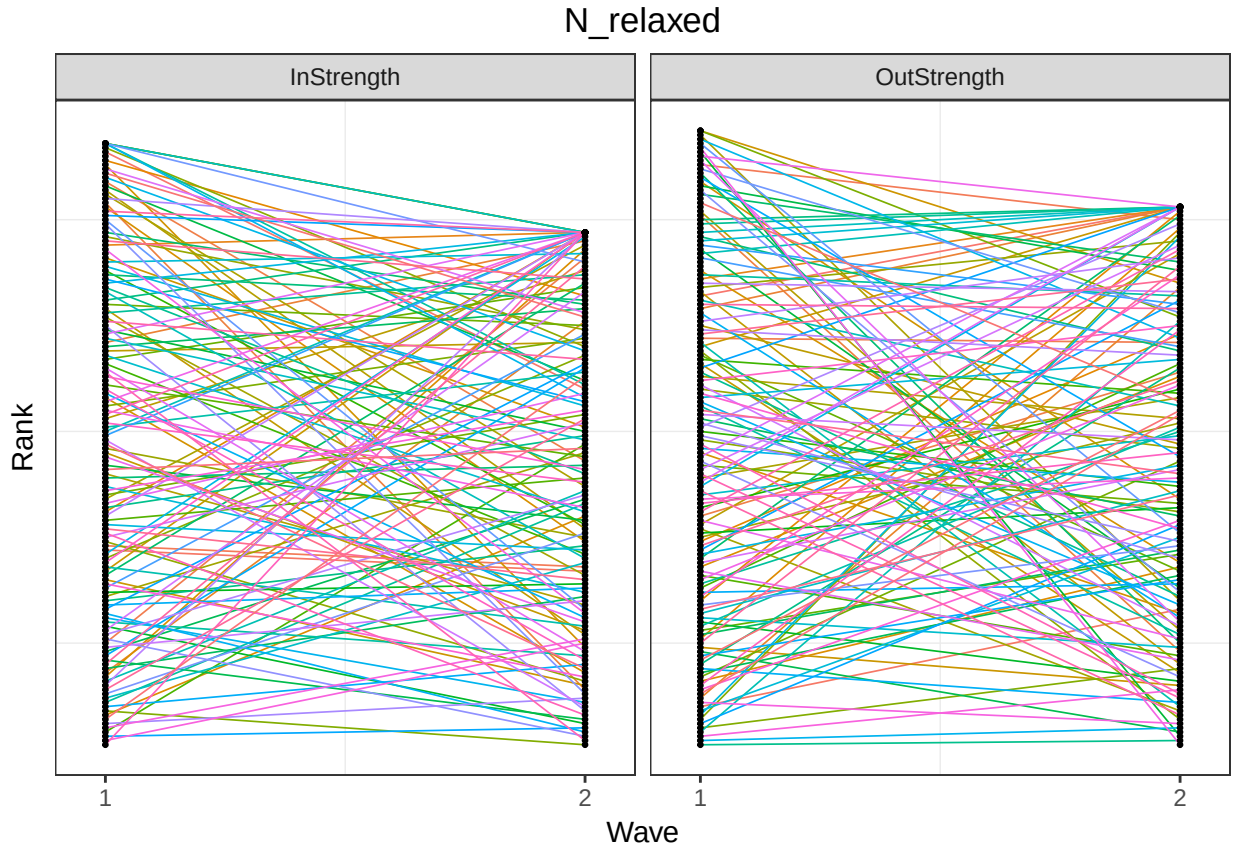
```
# create heatmap of contemporaneous rank order centrality correlations #
kappa_centrality_cor.df %>%
  filter(measure == "Strength") %>%
  mutate(measure2 = "Rank-Order") %>%
  filter(measure2 == "Rank-Order") %>%
  ggplot(aes(x = measure, y = var, fill = r)) +
    geom_raster() +
    scale_fill_gradient2(low = "blue", high = "red", mid = "white",
     midpoint = 0, limit = c(-.5,.5), space = "Lab",
     name="Centrality\nCorrelations") +
    geom_text(aes(label = round(r,2))) +
    labs(x = NULL, y = NULL) +
    facet_grid(.~measure) +
    theme_classic() +
    theme(axis.text = element_text(face = "bold"),
          axis.text.x = element_blank(),
          legend.text = element_text(face = "bold"),
          legend.title = element_text(face = "bold"),
          strip.text = element_text(face = "bold"))
```




```
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/PCC_centrality_var_cor.png",
       width = 3.5, height = 4)

# create a function to plot profiles of contemporaneosu centrality #
draw.a.plot <-  function(x){
  data <- kappa_centrality %>%
    select(-Betweenness, -Closeness) %>%
    filter(var == x & SID %in% w1w2_subs) %>%
    spread(key = wave, value = Strength) %>%
    mutate(diff = `1` - `2`) %>%
    gather(key = wave, value = Strength, `1`, `2`)
  max_cen <- ceiling(max(data$Strength, na.rm = T))
  break_cen <- max_cen/2
  plot <- ggplot(data, aes(x = SID, y = Strength, group = wave)) +
    geom_line(aes(color = wave), size = 1.2) +
    scale_y_continuous(limits = c(-max_cen, max_cen), breaks = seq(-max_cen, max_cen, break_cen)) +
    scale_color_manual(values = c("orchid", "black")) +
    #geom_point(size = .5) +
    labs(x = "Subject", y = "Centrality", color = "Wave",
         title = x) +
    coord_flip() +
    theme_bw() +
    theme(axis.text.y = element_blank(),
          axis.ticks.y = element_blank(),
          panel.grid.major = element_blank(),
          plot.title = element_text(hjust = .5))
  plot2 <- plot + geom_line(aes(y = diff), color = "blue", size = 1)
  return(list(plot, plot2))
  }
```

```
# create a function to plot ranks of contemporaneous centrality #
draw.a.plot <-  function(x){
  data <- beta_centrality %>%
    filter(var == x & SID %in% w1w2_subs) %>%
    select(-Betweenness, -Closeness) %>%
    gather(key = type2, value = Centrality, InStrength, OutStrength) %>%
    group_by(wave, type2, var) %>%
    mutate(rank = dense_rank(desc(Centrality)))
```

```
  ggplot(data, aes(x = as.numeric(wave), y = rank, group = factor(SID))) +
    geom_line(aes(color = factor(SID)), size = .3) +
    geom_point(size = .5) +
    scale_x_continuous(limits = c(.95, 2.05), breaks = c(1,2)) +
    labs(x = "Wave", y = "Rank", title = x) +
    theme_bw() +
    facet_grid(.~type2, scale = "free") +
    theme(panel.grid.major = element_blank(),
          axis.text.y = element_blank(),
          axis.ticks.y = element_blank(),
          plot.title = element_text(hjust = .5),
          legend.position = "none")
}

# plot sample ranks plot for N_relaxed #
draw.a.plot("N_relaxed")
```




```
ggsave("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/PCC_centrality_rank.png", width = 5, height = 5)

#set up function to loop through the draw.a.plot() function
loop.animate <- function() {
    lapply(varnames[1:9], function(i) {
        print(draw.a.plot(i))
    })
}
# saveGIF(loop.animate(), interval = .5, movie.name="PCC_centrality_rank.gif", ani.width = 400, ani.height = 400)
```

4.4.2.2 Profile Correlations by Centrality Measure

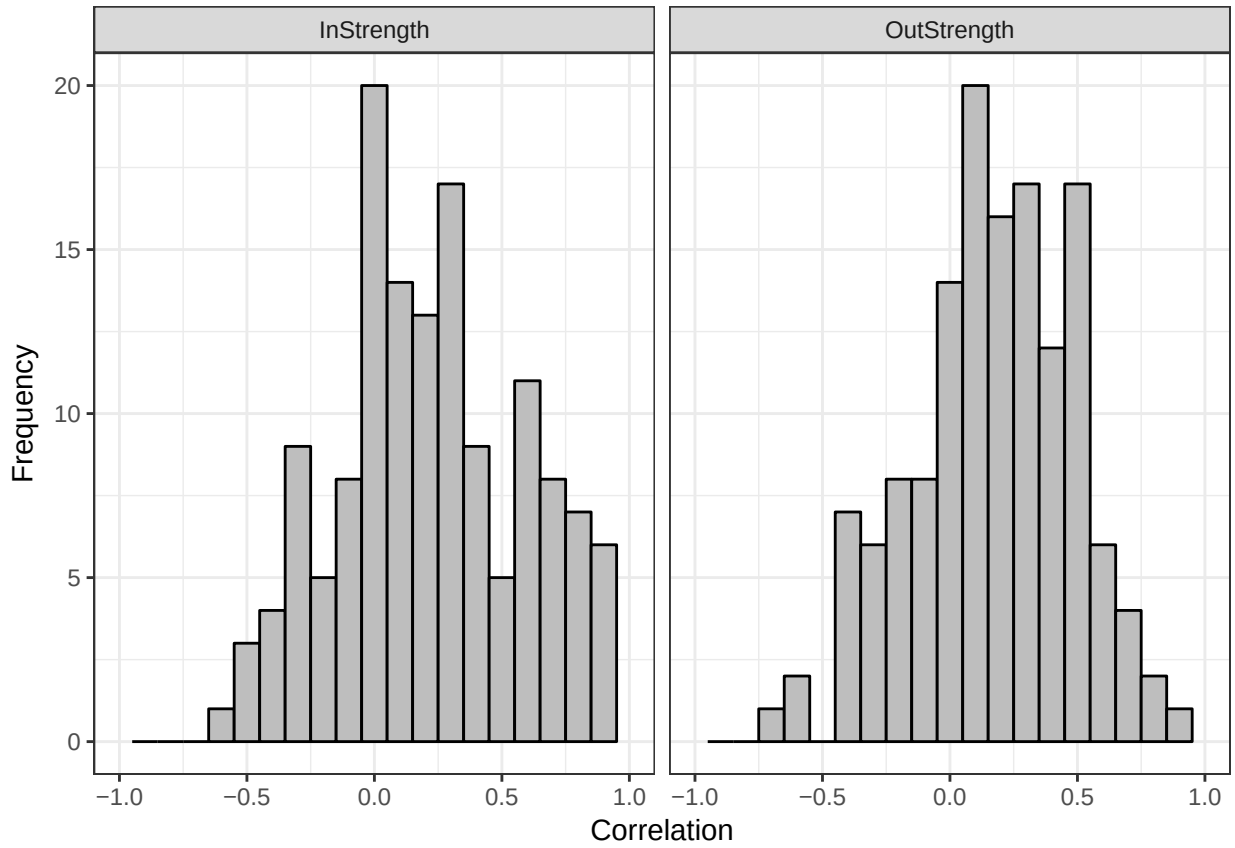
We can also examine profile correlations of the subjects across waves. That is, for a single subject, does the relative importance of a node in a network remain stable over time. We calculate a correlation for each subject and plot the results in a histogram.
#####Temporal

```r
# calculate each subjects profile correlations of temporal centrality #
beta_profile_cors <- tbl_df(beta_centrality) %>%
  dplyr::filter(SID %in% w1w2_subs) %>%
  gather(key = measure, value = Centrality, Betweenness:OutStrength) %>%
  spread(key = wave, value = Centrality) %>%
  arrange(SID, type, measure, var) %>%
  group_by(SID, type, measure) %>%
  nest() %>%
  mutate(r = map(data, ~cor(.$`1`, .$`2`, use = "pairwise")))

# unnest temporal centrality profile correlations #
beta_profile_cors.df <- beta_profile_cors %>% unnest(r, .drop = T)

# create table of average temporal centrality profile correlations #
kable(beta_profile_cors.df %>%
  group_by(type, measure) %>%
  summarize(mean = meanSD_r2z2r(r)[1],
            sd = meanSD_r2z2r(r)[2]),
  caption = "Average Profile Correlation for Partial Directed Correlations")
```

Table 9: Average Profile Correlation for Partial Directed Correlations

type	measure	mean	sd
Temporal	Betweenness	0.0454000	0.3975868
Temporal	Closeness	0.1146243	0.6533134
Temporal	InStrength	0.2773911	0.4430835
Temporal	OutStrength	0.1910302	0.3559153

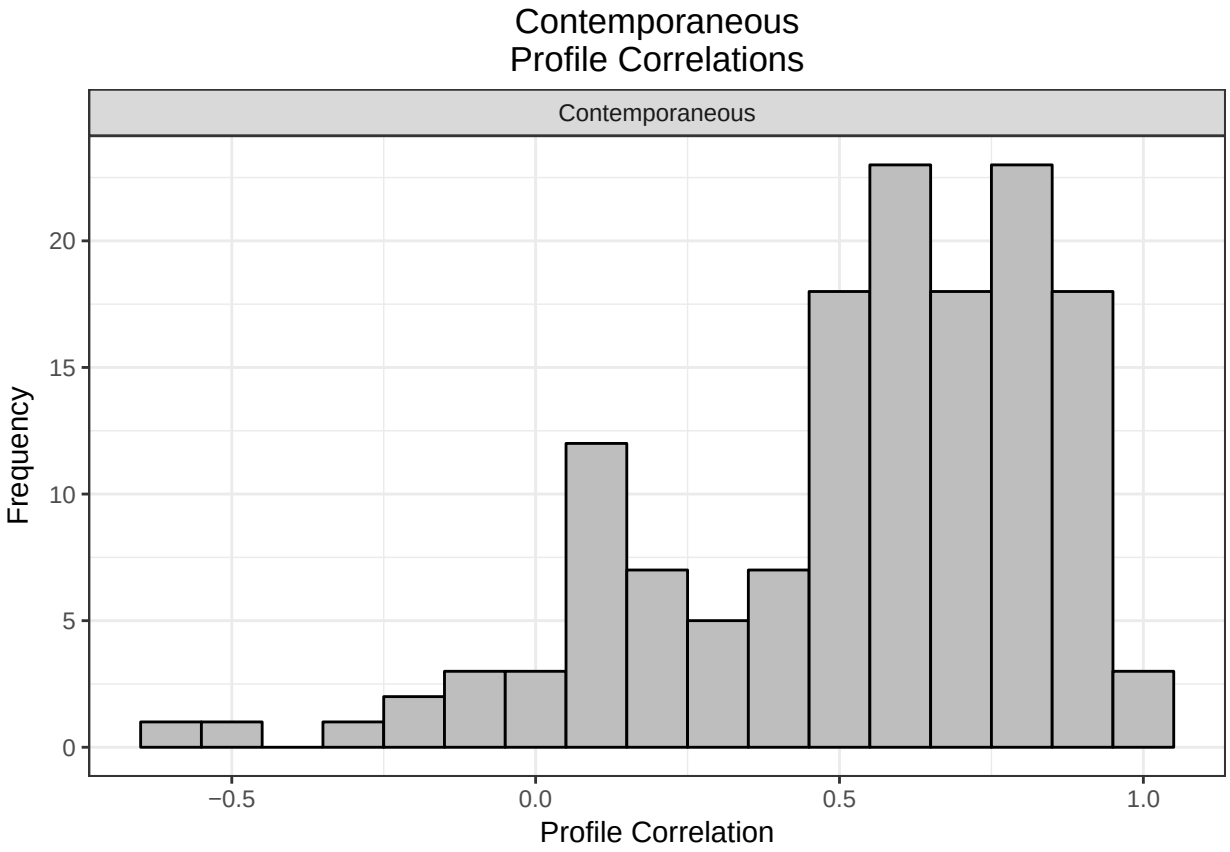
```r
# plot histogram of temporal centrality profile correlations #
beta_profile_cors.df %>%
  filter(measure %in% c("InStrength", "OutStrength")) %>%
  ggplot(aes(x = r)) +
  geom_histogram(binwidth = .1, color = "black", fill = "gray") +
  xlim(-1,1) +
  labs(x = "Correlation", y = "Frequency") +
  facet_grid(.~measure) +
  theme_bw()
```




```
ggsave("~/Box Sync/network/PAIRS/manuscript/R/ipsative_centralty.png", width = 6, height = 4)
```

4.4.2.2.1 Contemporaneous

```
# calculate each subjects profile correlations of contemporaneous centrality #
kappa_profile_cors <- kappa_centrality %>%
  filter(SID %in% w1w2_subs) %>%
  gather(key = measure, value = Centrality, Betweenness:Strength) %>%
  spread(key = wave, value = Centrality) %>%
  arrange(SID, type, measure, var) %>%
  group_by(SID, type, measure) %>%
  nest() %>%
  mutate(r = map(data, ~cor(.$`1`, .$`2`, use = "pairwise")))

# unnest contemporaneous centrality profile correlations #
kappa_profile_cors.df <- kappa_profile_cors %>% unnest(r, .drop = T)

# histogram of contemporaneous centrality profile correlations #
kappa_profile_cors.df %>%
  filter(measure == "Strength") %>%
  ggplot(aes(x = r))+
  geom_histogram(binwidth = .1, color = "black", fill = "gray")+
  labs(y = "Frequency", x = "Profile Correlation",
       title = "Contemporaneous\nProfile Correlations") +
  facet_wrap(~type) +
  theme_bw() +
  theme(plot.title = element_text(hjust = .5))
```



5 Supplemental Analyses

5.1 Gender

5.1.1 Population

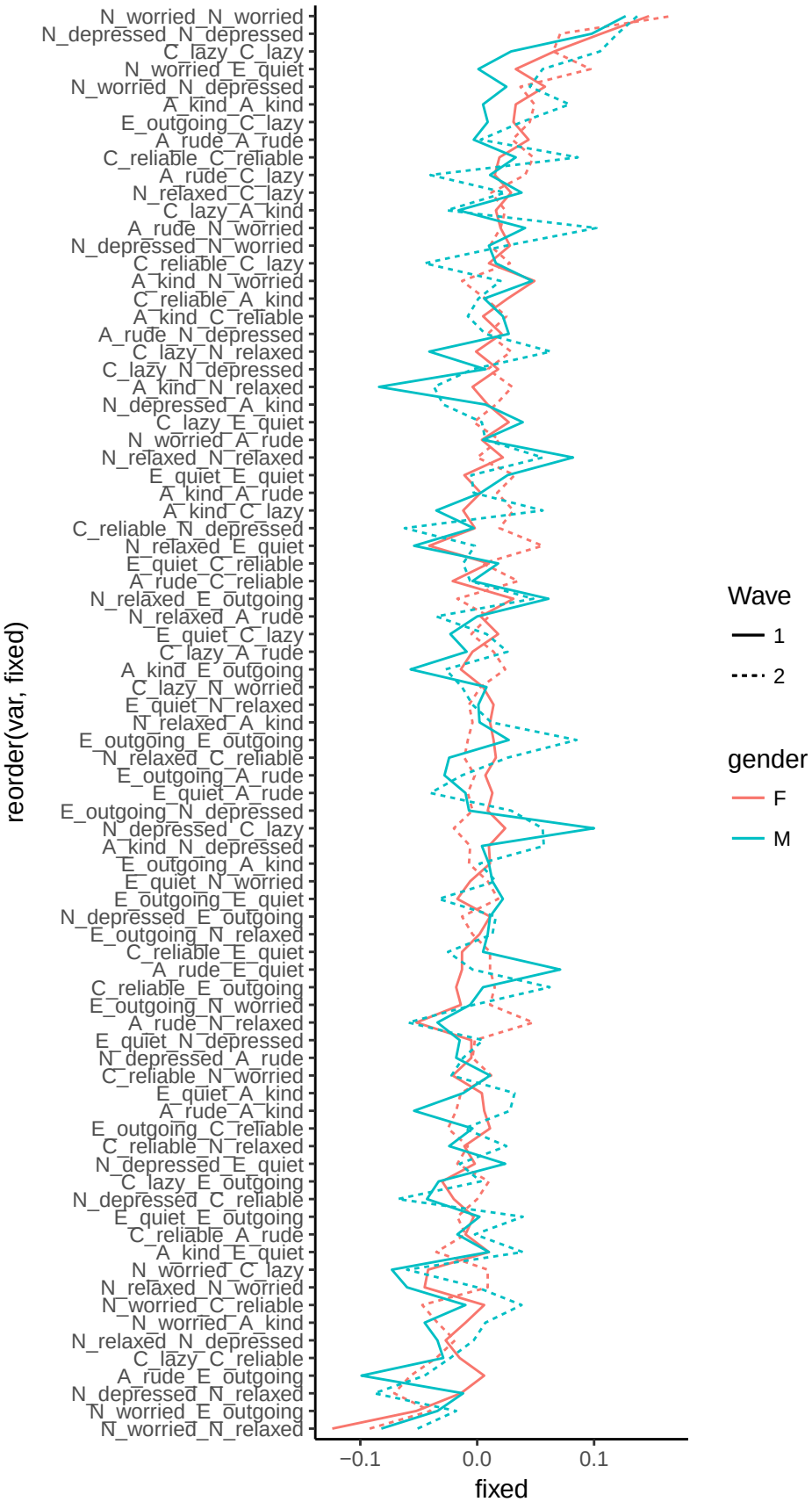
```
# create function #
mlVAR_fun <- function(x){
  mlVAR_test(x,
        vars = varnames2, #4:16
        idvar = "SID2",
        lags = 1,
        #dayvar = "day",
        beepvar = "beepvar3",
        temporal = "orthogonal",
        contemporaneous = "orthogonal",
        verbose = TRUE,
        scale = FALSE)
}

# create simple functions to extract temporal and contemoraneous effects
temp_eff_fun <- function(fit){summary(fit)$temporal}
contemp_eff_fun <- function(fit){summary(fit)$contemporaneous}

edge_colors <- RColorBrewer::brewer.pal(8, "Purples")[c(3,4,6)]

plot_fun <- function(mod, wave, type, dem){
  type <- ifelse(type == "Temporal", "temporal", "contemporaneous")
  plot <- plot(mod, type, layout = "spring", title = sprintf("%s Wave %s for %s" , type, wave, dem),
```

```
                  loop = .7, node.width = 1.4, edge.width = 1, label.font = 2,
                  label.fill.vertical = 1, label.fill.horizontal = 1, edge.color = "black",
                  legend = F, DoNotPlot = TRUE, mar = c(4,4,4,4))
  #change lines to dashed
  plot$graphAttributes$Edges$lty[plot$Edgelist$weight < 0] <- 2
  #change line colors
  plot$graphAttributes$Edges$color <-
    ifelse(abs(plot$Edgelist$weight) <.1, edge_colors[1],
    ifelse(abs(plot$Edgelist$weight) <.3, edge_colors[2], edge_colors[3]))
  #change variable names
  plot$graphAttributes$Nodes$labels <- varnames2
  return(plot)
}

# read in, rename, and recode gender, class year, and race
target.ratings.initial.w1 <- tbl_df(read.csv("~/Box Sync/network/PAIRS/Wave 1/target_w1_RENAMED.csv")) %>%
  select(ts.IDnum.w1, ts.DEM01.w1, ts.DEM05.w1, ts.DEM07.w1) %>%
  rename(SID = ts.IDnum.w1, gender = ts.DEM01.w1, class = ts.DEM05.w1, race = ts.DEM07.w1) %>%
  mutate(SID = factor(SID), gender = mapvalues(gender, c(1,2), c("M", "F")),
         class = mapvalues(class, c(seq(1,7,1),9,10,14), c(NA_real_, "Freshman", rep("Upperclassman",3), rep(NA_real_,5))),
         race = ifelse(race == 5, "white", "other"))

# add wave info
w1_com$Wave <- "1"
w2_com$Wave <- "2"

# combine w1, w2 ESM with demographic info
dat <- w1_com %>%
  full_join(w2_com) %>%
  full_join(target.ratings.initial.w1, by = "SID")
```

```
# keep only necessary variables and run mlVAR for each group@wave
gender_dat <- dat %>%
  select(SID, A_rude:SID2, Wave, gender) %>%
  filter(!(is.na(Wave)) & !(is.na(gender))) %>%
  group_by(Wave, gender) %>%
  nest() %>%
  mutate(mlvar = map(data, possibly(mlVAR_fun, NA_real_)))
```

```
load("~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/moderator_mlVAR.RData")

# extract temporal effects and unnest into data frame
gender_temp.df <- gender_dat %>%
  mutate(temp.df = map(mlvar, possibly(temp_eff_fun, NA_real_))) %>%
  unnest(temp.df, .drop = T) %>%
  unite(var, from, to, remove = F)
```

```
# plot profiles of each gender@wave
gender_temp.df %>%
  ggplot(aes(x = reorder(var, fixed), y = fixed)) +
    geom_line(data = filter(gender_temp.df, gender == "F"),
              aes(linetype = Wave, group = Wave, color = gender)) +
    geom_line(data = filter(gender_temp.df, gender == "M"),
              aes(linetype = Wave, group = Wave, color = gender)) +
    coord_flip() +
    theme_classic()
```


N_worried_N_worried
N_depressed_N_depressed
C_lazy_C_lazy
N_worried_E_quiet
N_worried_N_depressed
A_kind_A_kind
E_outgoing_C_lazy
A_rude_A_rude
C_reliable_C_reliable
A_rude_C_lazy
N_relaxed_C_lazy
C_lazy_A_kind
A_rude_N_worried
N_depressed_N_worried
C_reliable_C_lazy
A_kind_N_worried
C_reliable_A_kind
A_kind_C_reliable
A_rude_N_depressed
C_lazy_N_relaxed
C_lazy_N_depressed
A_kind_N_relaxed
N_depressed_A_kind
C_lazy_E_quiet
N_worried_A_rude
N_relaxed_N_relaxed
E_quiet_E_quiet
A_kind_A_rude
A_kind_C_lazy
C_reliable_N_depressed
N_relaxed_E_quiet
E_quiet_C_reliable
A_rude_C_reliable
N_relaxed_E_outgoing
N_relaxed_A_rude
E_quiet_C_lazy
C_lazy_A_rude
A_kind_E_outgoing
C_lazy_N_worried
E_quiet_N_relaxed
N_relaxed_A_kind
E_outgoing_E_outgoing
N_relaxed_C_reliable
E_outgoing_A_rude
E_quiet_A_rude
E_outgoing_N_depressed
N_depressed_C_lazy
A_kind_N_depressed
E_outgoing_A_kind
E_quiet_N_worried
E_outgoing_E_quiet
N_depressed_E_outgoing
E_outgoing_N_relaxed
C_reliable_E_quiet
A_rude_E_quiet
C_reliable_E_outgoing
E_outgoing_N_worried
A_rude_N_relaxed
E_quiet_N_depressed
N_depressed_A_rude
C_reliable_N_worried
E_quiet_A_kind
A_rude_A_kind
E_outgoing_C_reliable
C_reliable_N_relaxed
N_depressed_E_quiet
C_lazy_E_outgoing
N_depressed_C_reliable
E_quiet_E_outgoing
C_reliable_A_rude
A_kind_E_quiet
N_worried_C_lazy
N_relaxed_N_worried
N_worried_C_reliable
N_worried_A_kind
N_relaxed_N_depressed
C_lazy_C_reliable
A_rude_E_outgoing
N_depressed_N_relaxed
N_worried_E_outgoing
N_worried_N_relaxed
reorder(var, fixed)
−0.1
0.0
0.1
fixed
Wave
1
2
gender
F
M

```r
# profile correlation across waves for each gender
gender_temp.df %>%
  select(Wave:var, fixed) %>%
  spread(key = Wave, value = fixed) %>%
  group_by(gender) %>%
  summarize(r = cor(`1`, `2`, use = "pairwise"))
```

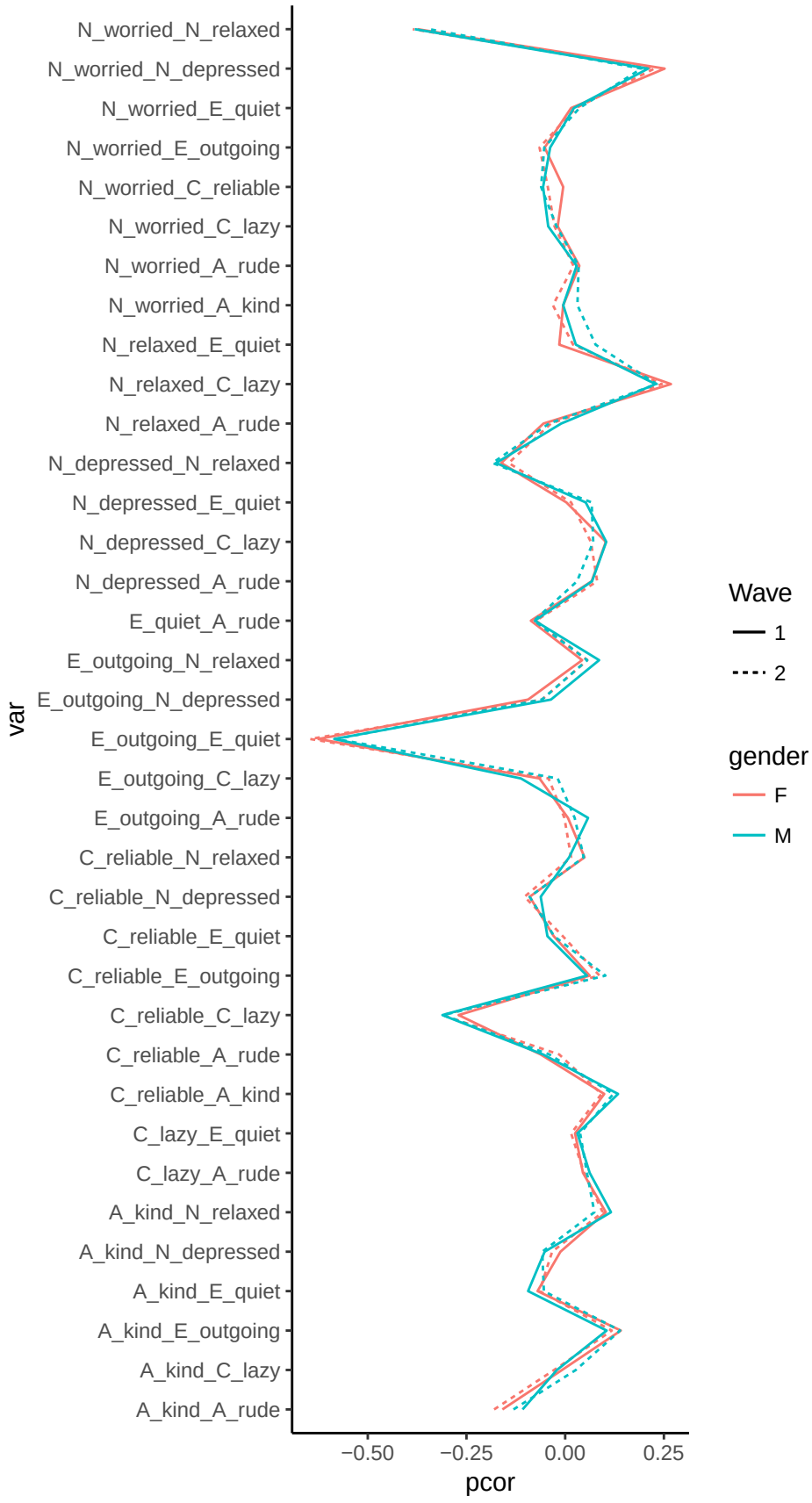
```
## # A tibble: 2 × 2
##   gender         r
##    <chr>     <dbl>
## 1      F 0.5424656
## 2      M 0.5429562
```

```r
# profile correlation across genders for each wave
gender_temp.df %>%
  select(Wave:var, fixed) %>%
  spread(key = gender, value = fixed) %>%
  group_by(Wave) %>%
  summarize(r = cor(F, M, use = "pairwise"))
```

```
## # A tibble: 2 × 2
##    Wave         r
##   <chr>     <dbl>
## 1     1 0.6803931
## 2     2 0.4218760
```

```r
# extract contemporaneous effects and unnest into data frame
gender_contemp.df <- gender_dat %>%
  mutate(temp.df = map(mlvar, possibly(contemp_eff_fun, NA_real_))) %>%
  unnest(temp.df, .drop = T) %>%
  unite(var, node1, node2, remove = F)
```

```r
# plot profiles of each gender @ wave
gender_contemp.df %>%
  ggplot(aes(x = var, y = pcor)) +
    geom_line(data = filter(gender_contemp.df, gender == "F"),
              aes(linetype = Wave, group = Wave, color = gender)) +
    geom_line(data = filter(gender_contemp.df, gender == "M"),
              aes(linetype = Wave, group = Wave, color = gender)) +
    coord_flip() +
    theme_classic()
```


N_worried_N_relaxed
N_worried_N_depressed
N_worried_E_quiet
N_worried_E_outgoing
N_worried_C_reliable
N_worried_C_lazy
N_worried_A_rude
N_worried_A_kind
N_relaxed_E_quiet
N_relaxed_C_lazy
N_relaxed_A_rude
N_depressed_N_relaxed
N_depressed_E_quiet
N_depressed_C_lazy
N_depressed_A_rude
E_quiet_A_rude
E_outgoing_N_relaxed
E_outgoing_N_depressed
E_outgoing_E_quiet
E_outgoing_C_lazy
E_outgoing_A_rude
C_reliable_N_relaxed
C_reliable_N_depressed
C_reliable_E_quiet
C_reliable_E_outgoing
C_reliable_C_lazy
C_reliable_A_rude
C_reliable_A_kind
C_lazy_E_quiet
C_lazy_A_rude
A_kind_N_relaxed
A_kind_N_depressed
A_kind_E_quiet
A_kind_E_outgoing
A_kind_C_lazy
A_kind_A_rude
var
−0.50
−0.25
0.00
0.25
pcor
Wave
1
2
gender
F
M

```r
# profile correlation of each gender across waves
gender_contemp.df %>%
  select(Wave:var, pcor) %>%
  spread(key = Wave, value = pcor) %>%
  group_by(gender) %>%
  summarize(r = cor(`1`, `2`, use = "pairwise"))
```

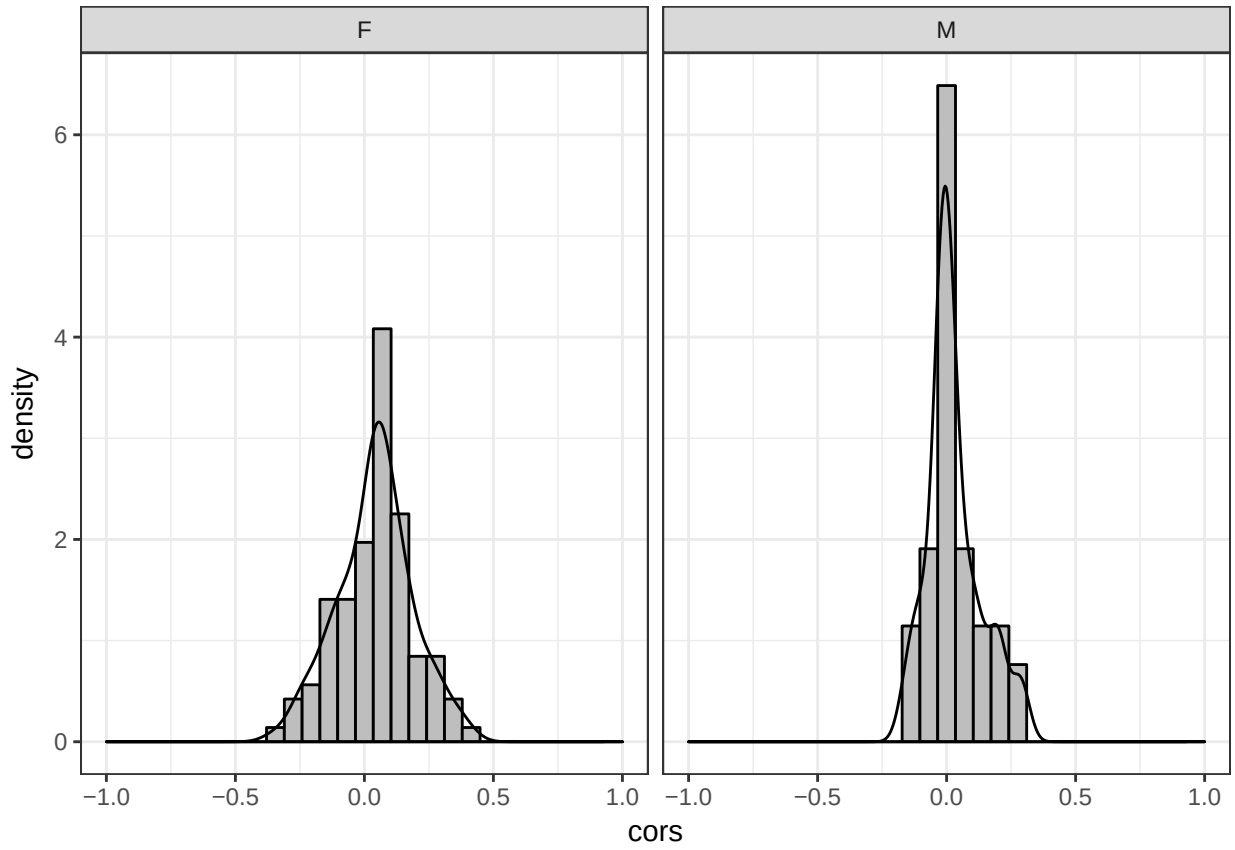
```
## # A tibble: 2 × 2
##   gender         r
##    <chr>     <dbl>
## 1      F 0.9895826
## 2      M 0.9798360
```

```r
# profile correlation of each wave across genders
gender_contemp.df %>%
  select(Wave:var, pcor) %>%
  spread(key = gender, value = pcor) %>%
  group_by(Wave) %>%
  summarize(r = cor(F, M, use = "pairwise"))
```

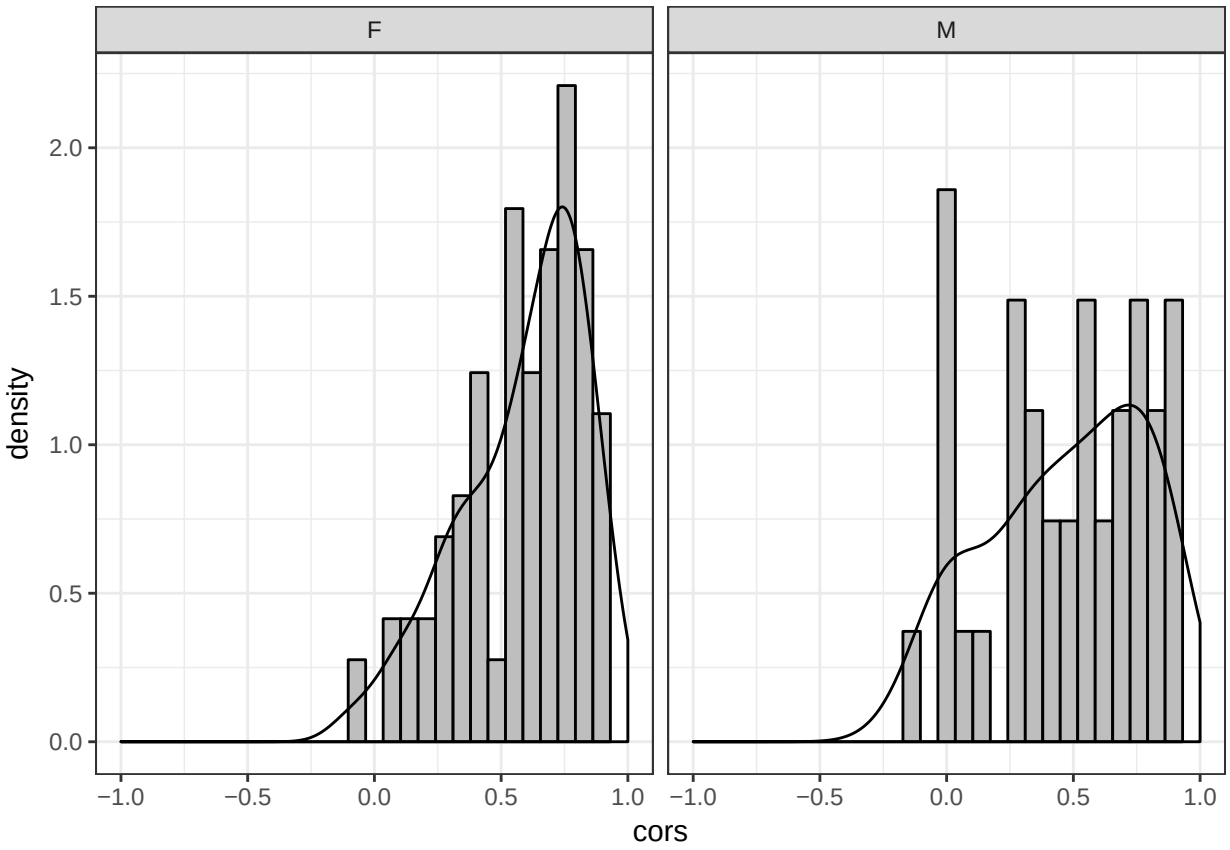
```
## # A tibble: 2 × 2
##    Wave         r
##   <chr>     <dbl>
## 1     1 0.9787086
## 2     2 0.9833809
```

5.1.2 Idiographic

```r
ip_cors <- ip_cors %>%
  mutate(SID = factor(SID)) %>%
  full_join(target.ratings.initial.w1, by = "SID")

# Temporal
ip_cors %>%
 filter(!(is.na(gender)) & !(is.na(cors)) & type2 == "Temporal") %>%
  ggplot(aes(x = cors)) +
    geom_histogram(aes(y = ..density..), color = "black", fill = "gray") +
    geom_density() +
    scale_x_continuous(limits = c(-1,1), breaks = seq(-1, 1,.5)) +
    facet_grid(.~gender) +
    theme_bw()
```




```
# Contemporaneous
ip_cors %>%
 filter(!(is.na(gender)) & !(is.na(cors)) & type2 == "Contemporaneous") %>%
  ggplot(aes(x = cors)) +
    geom_histogram(aes(y = ..density..), color = "black", fill = "gray") +
    geom_density() +
    scale_x_continuous(limits = c(-1,1), breaks = seq(-1, 1,.5)) +
    facet_grid(.~gender) +
    theme_bw()
```



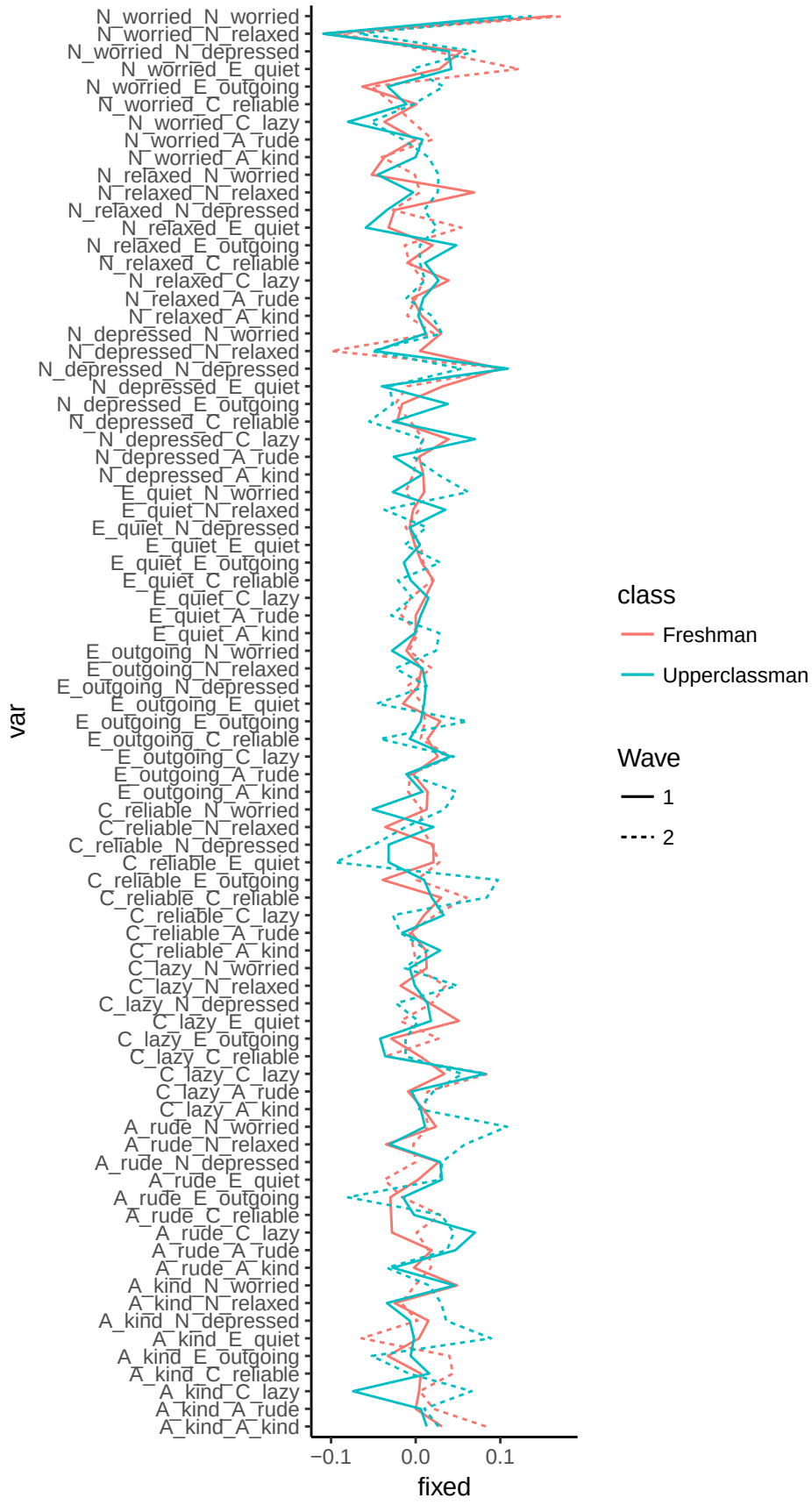
5.2 Class Year

5.2.1 Population

```
# keep only necessary variables and run mlVAR for each group@wave
class_dat <- dat %>%
  select(SID, A_rude:SID2, Wave, class) %>%
  filter(!(is.na(Wave)) & !(is.na(class))) %>%
  group_by(Wave, class) %>%
  nest() %>%
  mutate(mlvar = map(data, possibly(mlVAR_fun, NA_real_)))
```

```
# extract temporal effects and unnest into data frame
class_temp.df <- class_dat %>%
  mutate(temp.df = map(mlvar, possibly(temp_eff_fun, NA_real_))) %>%
  unnest(temp.df, .drop = T) %>%
  unite(var, from, to, remove = F)
```

```
# plot profiles of each class@wave
class_temp.df %>%
  ggplot(aes(x = var, y = fixed)) +
    geom_line(data = filter(class_temp.df, class == "Freshman"),
              aes(linetype = Wave, group = Wave, color = class)) +
    geom_line(data = filter(class_temp.df, class == "Upperclassman"),
              aes(linetype = Wave, group = Wave, color = class)) +
    coord_flip() +
    theme_classic()
```


N_worried_N_worried
N_worried_N_relaxed
N_worried_N_depressed
N_worried_E_quiet
N_worried_E_outgoing
N_worried_C_reliable
N_worried_C_lazy
N_worried_A_rude
N_worried_A_kind
N_relaxed_N_worried
N_relaxed_N_relaxed
N_relaxed_N_depressed
N_relaxed_E_quiet
N_relaxed_E_outgoing
N_relaxed_C_reliable
N_relaxed_C_lazy
N_relaxed_A_rude
N_relaxed_A_kind
N_depressed_N_worried
N_depressed_N_relaxed
N_depressed_N_depressed
N_depressed_E_quiet
N_depressed_E_outgoing
N_depressed_C_reliable
N_depressed_C_lazy
N_depressed_A_rude
N_depressed_A_kind
E_quiet_N_worried
E_quiet_N_relaxed
E_quiet_N_depressed
E_quiet_E_quiet
E_quiet_E_outgoing
E_quiet_C_reliable
E_quiet_C_lazy
E_quiet_A_rude
E_quiet_A_kind
E_outgoing_N_worried
E_outgoing_N_relaxed
E_outgoing_N_depressed
E_outgoing_E_quiet
E_outgoing_E_outgoing
E_outgoing_C_reliable
E_outgoing_C_lazy
E_outgoing_A_rude
E_outgoing_A_kind
C_reliable_N_worried
C_reliable_N_relaxed
C_reliable_N_depressed
C_reliable_E_quiet
C_reliable_E_outgoing
C_reliable_C_reliable
C_reliable_C_lazy
C_reliable_A_rude
C_reliable_A_kind
C_lazy_N_worried
C_lazy_N_relaxed
C_lazy_N_depressed
C_lazy_E_quiet
C_lazy_E_outgoing
C_lazy_C_reliable
C_lazy_C_lazy
C_lazy_A_rude
C_lazy_A_kind
A_rude_N_worried
A_rude_N_relaxed
A_rude_N_depressed
A_rude_E_quiet
A_rude_E_outgoing
A_rude_C_reliable
A_rude_C_lazy
A_rude_A_rude
A_rude_A_kind
A_kind_N_worried
A_kind_N_relaxed
A_kind_N_depressed
A_kind_E_quiet
A_kind_E_outgoing
A_kind_C_reliable
A_kind_C_lazy
A_kind_A_rude
A_kind_A_kind
var
−0.1
0.0
0.1
fixed
class
Freshman
Upperclassman
Wave
1
2

```r
# profile correlation across waves for each class
class_temp.df %>%
  select(Wave:var, fixed) %>%
  spread(key = Wave, value = fixed) %>%
  group_by(class) %>%
  summarize(r = cor(`1`, `2`, use = "pairwise"))
```

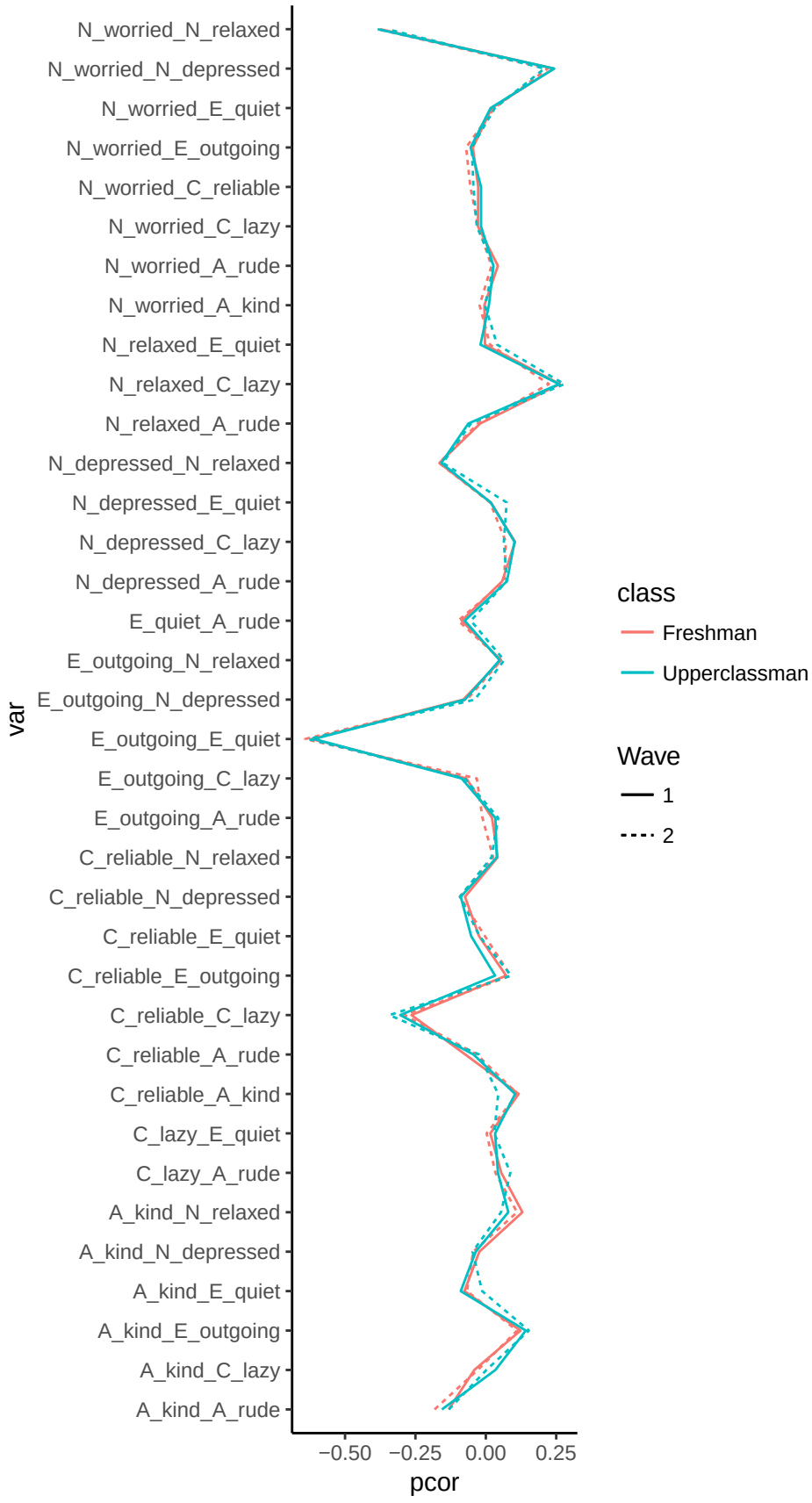
```
## # A tibble: 2 × 2
##           class         r
##           <chr>     <dbl>
## 1      Freshman 0.5829956
## 2 Upperclassman 0.3677324
```

```r
# profile correlation across classes for each wave
class_temp.df %>%
  select(Wave:var, fixed) %>%
  spread(key = class, value = fixed) %>%
  group_by(Wave) %>%
  summarize(r = cor(Freshman, Upperclassman, use = "pairwise"))
```

```
## # A tibble: 2 × 2
##    Wave         r
##   <chr>     <dbl>
## 1     1 0.6356068
## 2     2 0.3098215
```

```r
# extract contemporaneous effects and unnest into data frame
class_contemp.df <- class_dat %>%
  mutate(temp.df = map(mlvar, possibly(contemp_eff_fun, NA_real_))) %>%
  unnest(temp.df, .drop = T) %>%
  unite(var, node1, node2, remove = F)
```

```r
# plot profiles of each class @ wave
class_contemp.df %>%
  ggplot(aes(x = var, y = pcor)) +
    geom_line(data = filter(class_contemp.df, class == "Freshman"),
              aes(linetype = Wave, group = Wave, color = class)) +
    geom_line(data = filter(class_contemp.df, class == "Upperclassman"),
              aes(linetype = Wave, group = Wave, color = class)) +
    coord_flip() +
    theme_classic()
```


N_worried_N_relaxed
N_worried_N_depressed
N_worried_E_quiet
N_worried_E_outgoing
N_worried_C_reliable
N_worried_C_lazy
N_worried_A_rude
N_worried_A_kind
N_relaxed_E_quiet
N_relaxed_C_lazy
N_relaxed_A_rude
N_depressed_N_relaxed
N_depressed_E_quiet
N_depressed_C_lazy
N_depressed_A_rude
E_quiet_A_rude
E_outgoing_N_relaxed
E_outgoing_N_depressed
E_outgoing_E_quiet
E_outgoing_C_lazy
E_outgoing_A_rude
C_reliable_N_relaxed
C_reliable_N_depressed
C_reliable_E_quiet
C_reliable_E_outgoing
C_reliable_C_lazy
C_reliable_A_rude
C_reliable_A_kind
C_lazy_E_quiet
C_lazy_A_rude
A_kind_N_relaxed
A_kind_N_depressed
A_kind_E_quiet
A_kind_E_outgoing
A_kind_C_lazy
A_kind_A_rude
var
−0.50
−0.25
0.00
0.25
pcor
class
Freshman
Upperclassman
Wave
1
2

```r
# profile correlation of each class across waves
class_contemp.df %>%
  select(Wave:var, pcor) %>%
  spread(key = Wave, value = pcor) %>%
  group_by(class) %>%
  summarize(r = cor(`1`, `2`, use = "pairwise"))
```

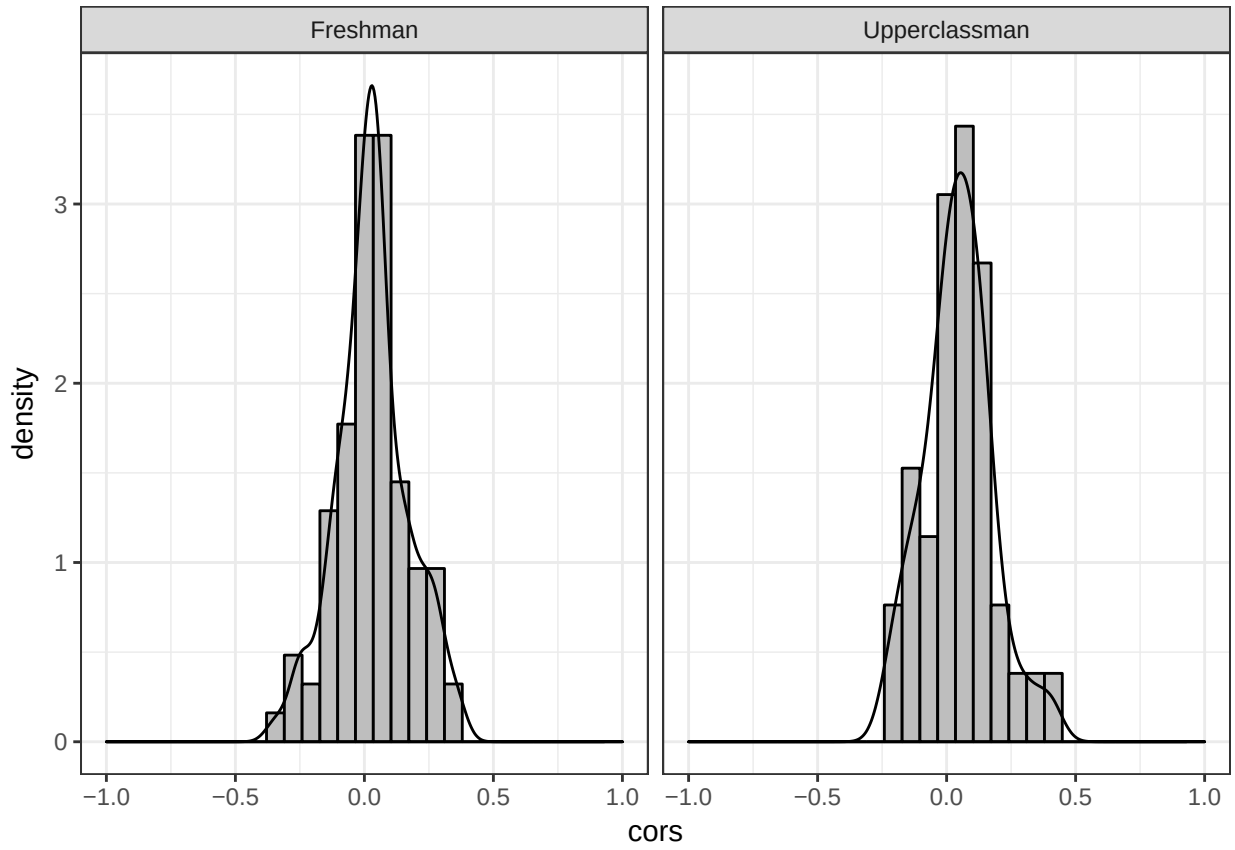
```
## # A tibble: 2 × 2
##           class         r
##           <chr>     <dbl>
## 1      Freshman 0.9914466
## 2 Upperclassman 0.9806495
```

```r
# profile correlation of each wave across classes
class_contemp.df %>%
  select(Wave:var, pcor) %>%
  spread(key = class, value = pcor) %>%
  group_by(Wave) %>%
  summarize(r = cor(Freshman, Upperclassman, use = "pairwise"))
```

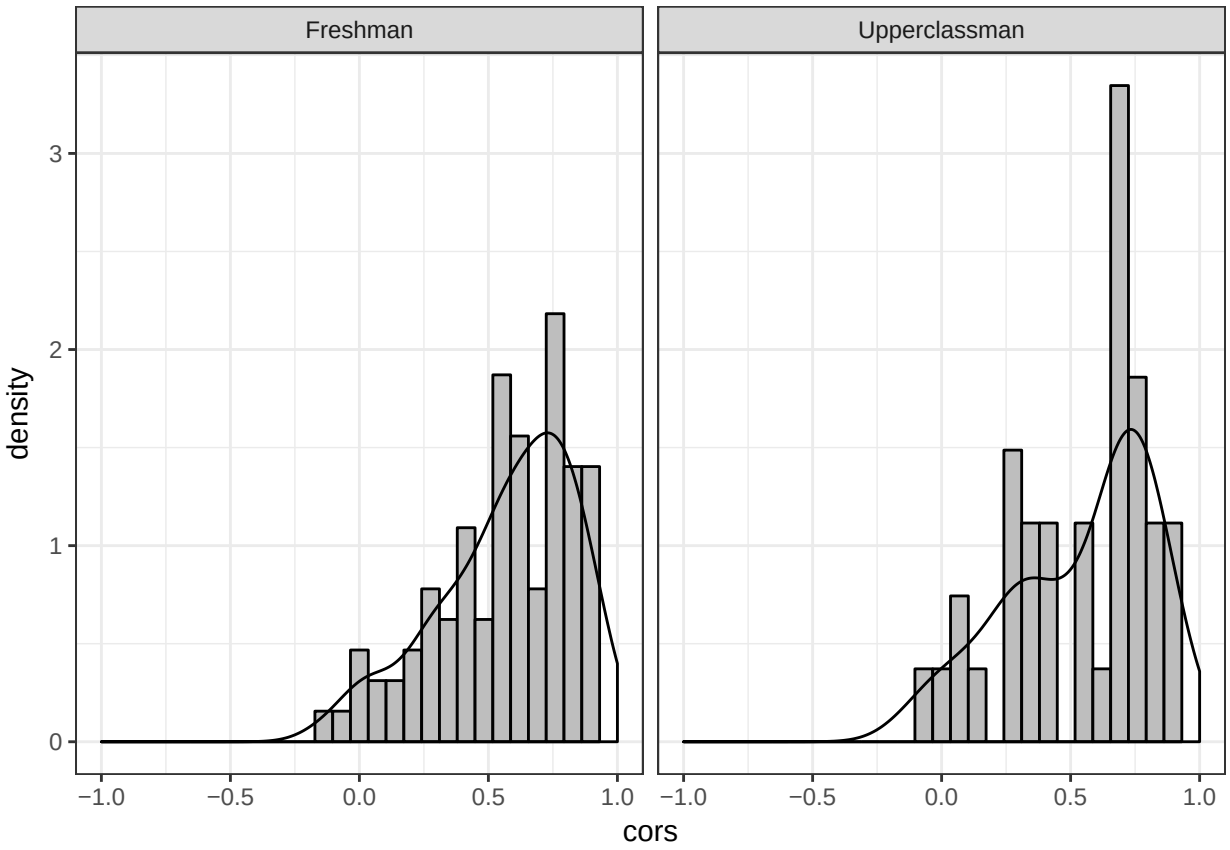
```
## # A tibble: 2 × 2
##    Wave         r
##   <chr>     <dbl>
## 1     1 0.9894037
## 2     2 0.9785398
```

5.2.2 Idiographic

```r
# Temporal
ip_cors %>%
 filter(!(is.na(class)) & !(is.na(cors)) & type2 == "Temporal") %>%
  ggplot(aes(x = cors)) +
    geom_histogram(aes(y = ..density..), color = "black", fill = "gray") +
    geom_density() +
    scale_x_continuous(limits = c(-1,1), breaks = seq(-1, 1,.5)) +
    facet_grid(.~class) +
    theme_bw()
```




```
# Contemporaneous
ip_cors %>%
 filter(!(is.na(class)) & !(is.na(cors)) & type2 == "Contemporaneous") %>%
  ggplot(aes(x = cors)) +
    geom_histogram(aes(y = ..density..), color = "black", fill = "gray") +
    geom_density() +
    scale_x_continuous(limits = c(-1,1), breaks = seq(-1, 1,.5)) +
    facet_grid(.~class) +
    theme_bw()
```



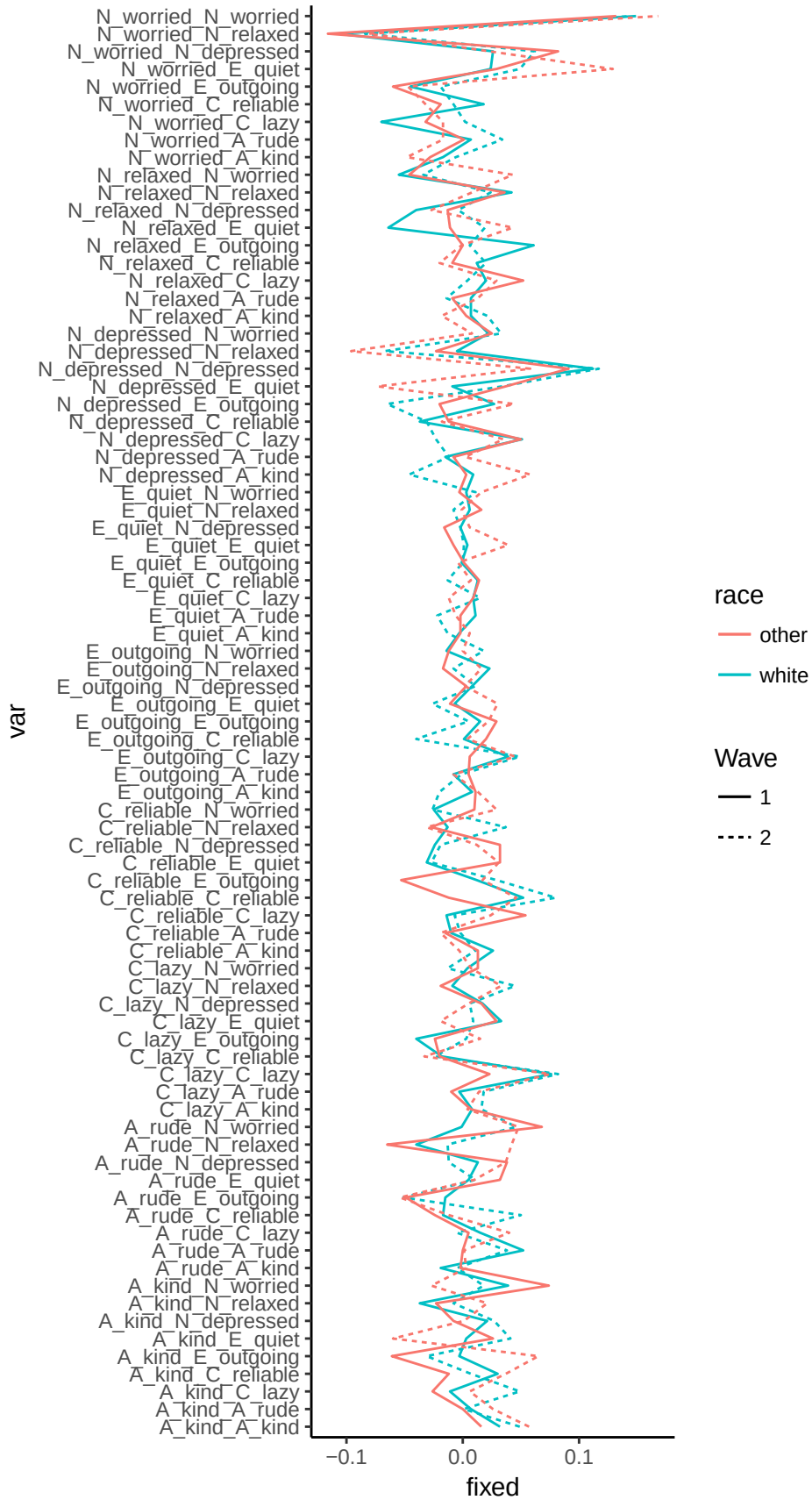
5.3 Race

5.3.1 Population

```
# keep only necessary variables and run mlVAR for each group@wave
race_dat <- dat %>%
  select(SID, A_rude:SID2, Wave, race) %>%
  filter(!(is.na(Wave)) & !(is.na(race))) %>%
  group_by(Wave, race) %>%
  nest() %>%
  mutate(mlvar = map(data, possibly(mlVAR_fun, NA_real_)))
```

```
# extract temporal effects and unnest into data frame
race_temp.df <- race_dat %>%
  mutate(temp.df = map(mlvar, possibly(temp_eff_fun, NA_real_))) %>%
  unnest(temp.df, .drop = T) %>%
  unite(var, from, to, remove = F)
```

```
# plot profiles of each race@wave
race_temp.df %>%
  ggplot(aes(x = var, y = fixed)) +
    geom_line(data = filter(race_temp.df, race == "white"),
              aes(linetype = Wave, group = Wave, color = race)) +
    geom_line(data = filter(race_temp.df, race == "other"),
              aes(linetype = Wave, group = Wave, color = race)) +
    coord_flip() +
    theme_classic()
```


N_worried_N_worried
N_worried_N_relaxed
N_worried_N_depressed
N_worried_E_quiet
N_worried_E_outgoing
N_worried_C_reliable
N_worried_C_lazy
N_worried_A_rude
N_worried_A_kind
N_relaxed_N_worried
N_relaxed_N_relaxed
N_relaxed_N_depressed
N_relaxed_E_quiet
N_relaxed_E_outgoing
N_relaxed_C_reliable
N_relaxed_C_lazy
N_relaxed_A_rude
N_relaxed_A_kind
N_depressed_N_worried
N_depressed_N_relaxed
N_depressed_N_depressed
N_depressed_E_quiet
N_depressed_E_outgoing
N_depressed_C_reliable
N_depressed_C_lazy
N_depressed_A_rude
N_depressed_A_kind
E_quiet_N_worried
E_quiet_N_relaxed
E_quiet_N_depressed
E_quiet_E_quiet
E_quiet_E_outgoing
E_quiet_C_reliable
E_quiet_C_lazy
E_quiet_A_rude
E_quiet_A_kind
E_outgoing_N_worried
E_outgoing_N_relaxed
E_outgoing_N_depressed
E_outgoing_E_quiet
E_outgoing_E_outgoing
E_outgoing_C_reliable
E_outgoing_C_lazy
E_outgoing_A_rude
E_outgoing_A_kind
C_reliable_N_worried
C_reliable_N_relaxed
C_reliable_N_depressed
C_reliable_E_quiet
C_reliable_E_outgoing
C_reliable_C_reliable
C_reliable_C_lazy
C_reliable_A_rude
C_reliable_A_kind
C_lazy_N_worried
C_lazy_N_relaxed
C_lazy_N_depressed
C_lazy_E_quiet
C_lazy_E_outgoing
C_lazy_C_reliable
C_lazy_C_lazy
C_lazy_A_rude
C_lazy_A_kind
A_rude_N_worried
A_rude_N_relaxed
A_rude_N_depressed
A_rude_E_quiet
A_rude_E_outgoing
A_rude_C_reliable
A_rude_C_lazy
A_rude_A_rude
A_rude_A_kind
A_kind_N_worried
A_kind_N_relaxed
A_kind_N_depressed
A_kind_E_quiet
A_kind_E_outgoing
A_kind_C_reliable
A_kind_C_lazy
A_kind_A_rude
A_kind_A_kind
var
−0.1
0.0
0.1
fixed
race
other
white
Wave
1
2

```r
# profile correlation across waves for each race
race_temp.df %>%
  select(Wave:var, fixed) %>%
  spread(key = Wave, value = fixed) %>%
  group_by(race) %>%
  summarize(r = cor(`1`, `2`, use = "pairwise"))
```

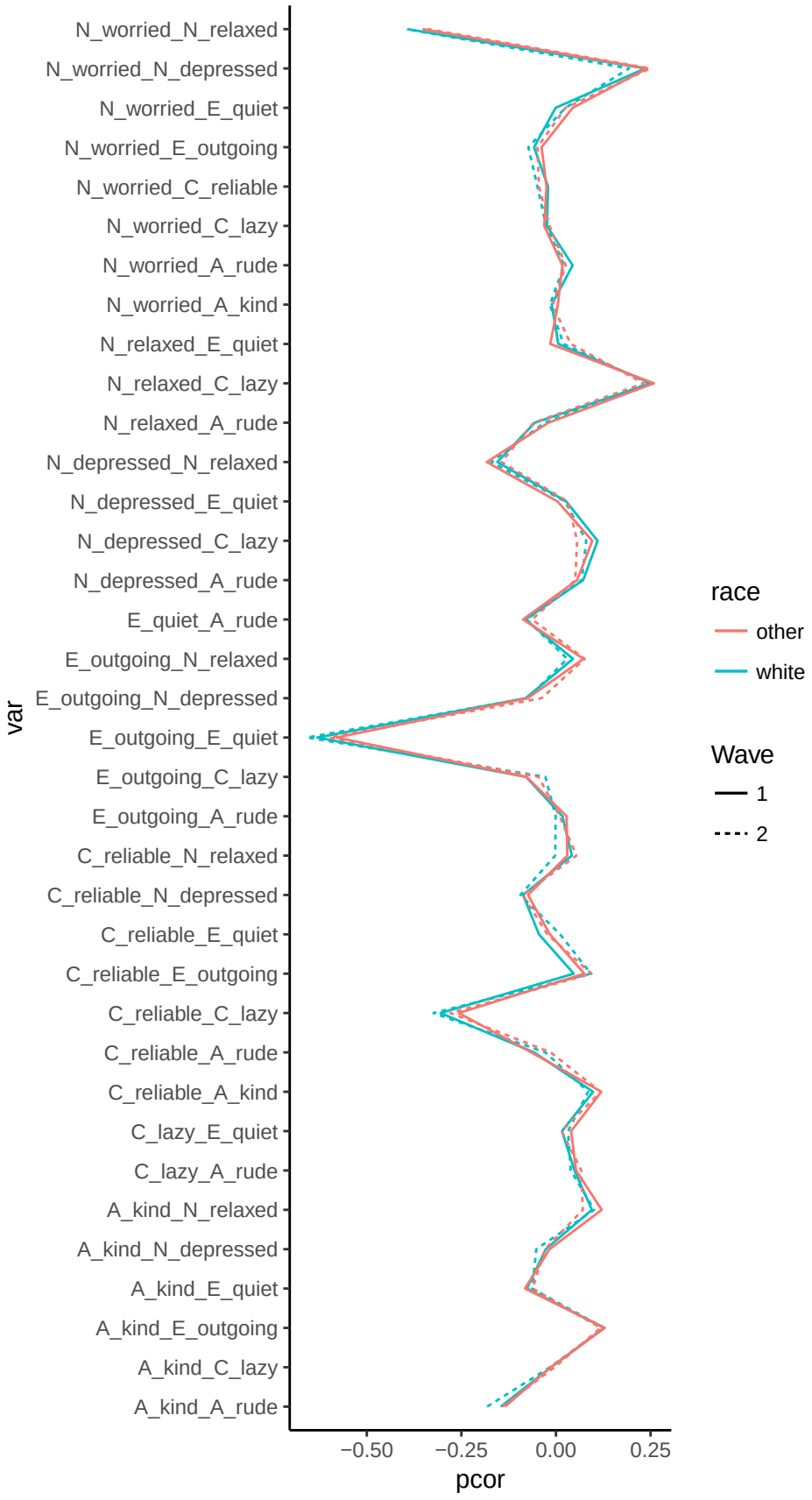
```
## # A tibble: 2 × 2
##    race         r
##   <chr>     <dbl>
## 1 other 0.4111073
## 2 white 0.6202292
```

```r
# profile correlation across races for each wave
race_temp.df %>%
  select(Wave:var, fixed) %>%
  spread(key = race, value = fixed) %>%
  group_by(Wave) %>%
  summarize(r = cor(white, other, use = "pairwise"))
```

```
## # A tibble: 2 × 2
##    Wave         r
##   <chr>     <dbl>
## 1     1 0.6647653
## 2     2 0.4299743
```

```r
# extract contemporaneous effects and unnest into data frame
race_contemp.df <- race_dat %>%
  mutate(temp.df = map(mlvar, possibly(contemp_eff_fun, NA_real_))) %>%
  unnest(temp.df, .drop = T) %>%
  unite(var, node1, node2, remove = F)
```

```r
# plot profiles of each race @ wave
race_contemp.df %>%
  ggplot(aes(x = var, y = pcor)) +
    geom_line(data = filter(race_contemp.df, race == "white"),
              aes(linetype = Wave, group = Wave, color = race)) +
    geom_line(data = filter(race_contemp.df, race == "other"),
              aes(linetype = Wave, group = Wave, color = race)) +
    coord_flip() +
    theme_classic()
```


N_worried_N_relaxed
N_worried_N_depressed
N_worried_E_quiet
N_worried_E_outgoing
N_worried_C_reliable
N_worried_C_lazy
N_worried_A_rude
N_worried_A_kind
N_relaxed_E_quiet
N_relaxed_C_lazy
N_relaxed_A_rude
N_depressed_N_relaxed
N_depressed_E_quiet
N_depressed_C_lazy
N_depressed_A_rude
E_quiet_A_rude
E_outgoing_N_relaxed
E_outgoing_N_depressed
E_outgoing_E_quiet
E_outgoing_C_lazy
E_outgoing_A_rude
C_reliable_N_relaxed
C_reliable_N_depressed
C_reliable_E_quiet
C_reliable_E_outgoing
C_reliable_C_lazy
C_reliable_A_rude
C_reliable_A_kind
C_lazy_E_quiet
C_lazy_A_rude
A_kind_N_relaxed
A_kind_N_depressed
A_kind_E_quiet
A_kind_E_outgoing
A_kind_C_lazy
A_kind_A_rude
var
−0.50
−0.25
0.00
0.25
pcor
race
other
white
Wave
1
2

```r
# profile correlation of each race across waves
race_contemp.df %>%
  select(Wave:var, pcor) %>%
  spread(key = Wave, value = pcor) %>%
  group_by(race) %>%
  summarize(r = cor(`1`, `2`, use = "pairwise"))
```

```
## # A tibble: 2 × 2
##    race         r
##   <chr>     <dbl>
## 1 other 0.9857999
## 2 white 0.9885170
```

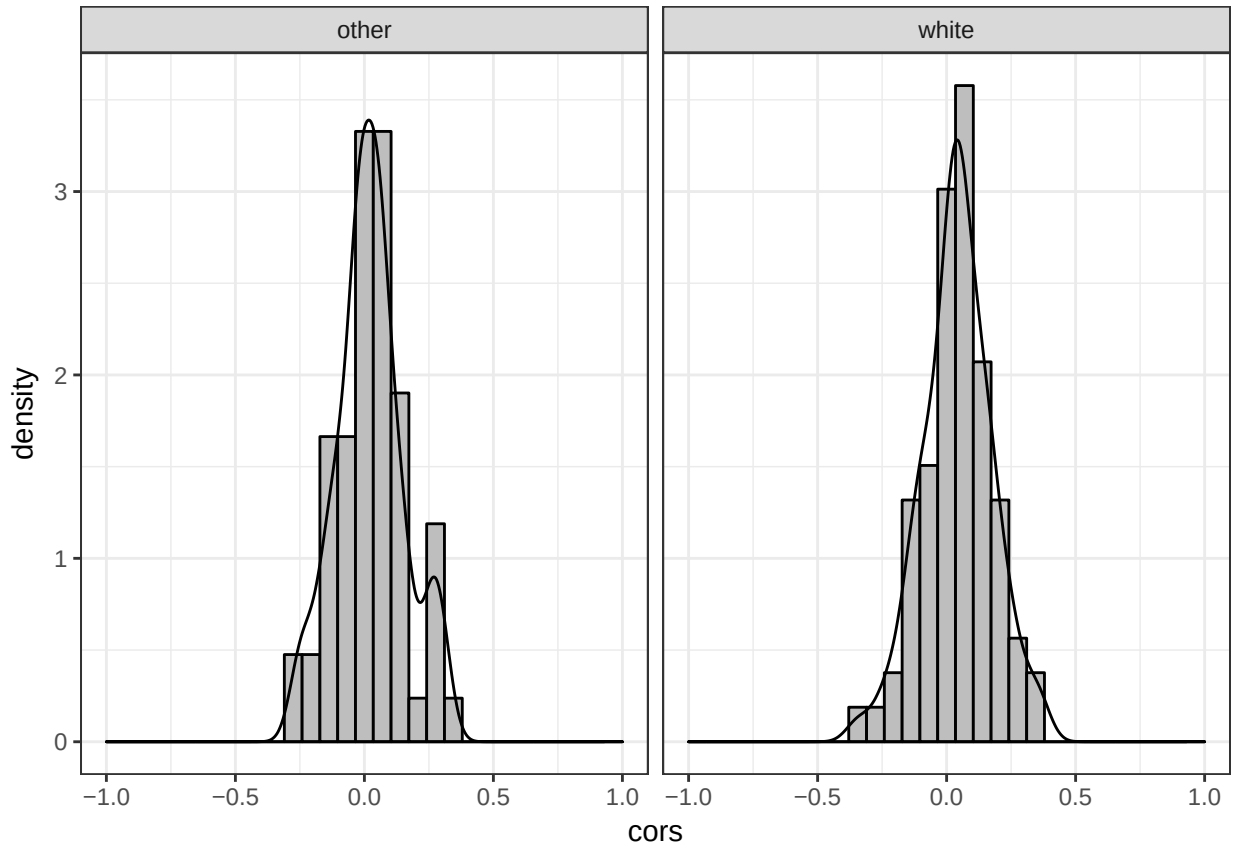
```r
# profile correlation of each wave across races
race_contemp.df %>%
  select(Wave:var, pcor) %>%
  spread(key = race, value = pcor) %>%
  group_by(Wave) %>%
  summarize(r = cor(white, other, use = "pairwise"))
```

```
## # A tibble: 2 × 2
##    Wave         r
##   <chr>     <dbl>
## 1     1 0.9918624
## 2     2 0.9885678
```

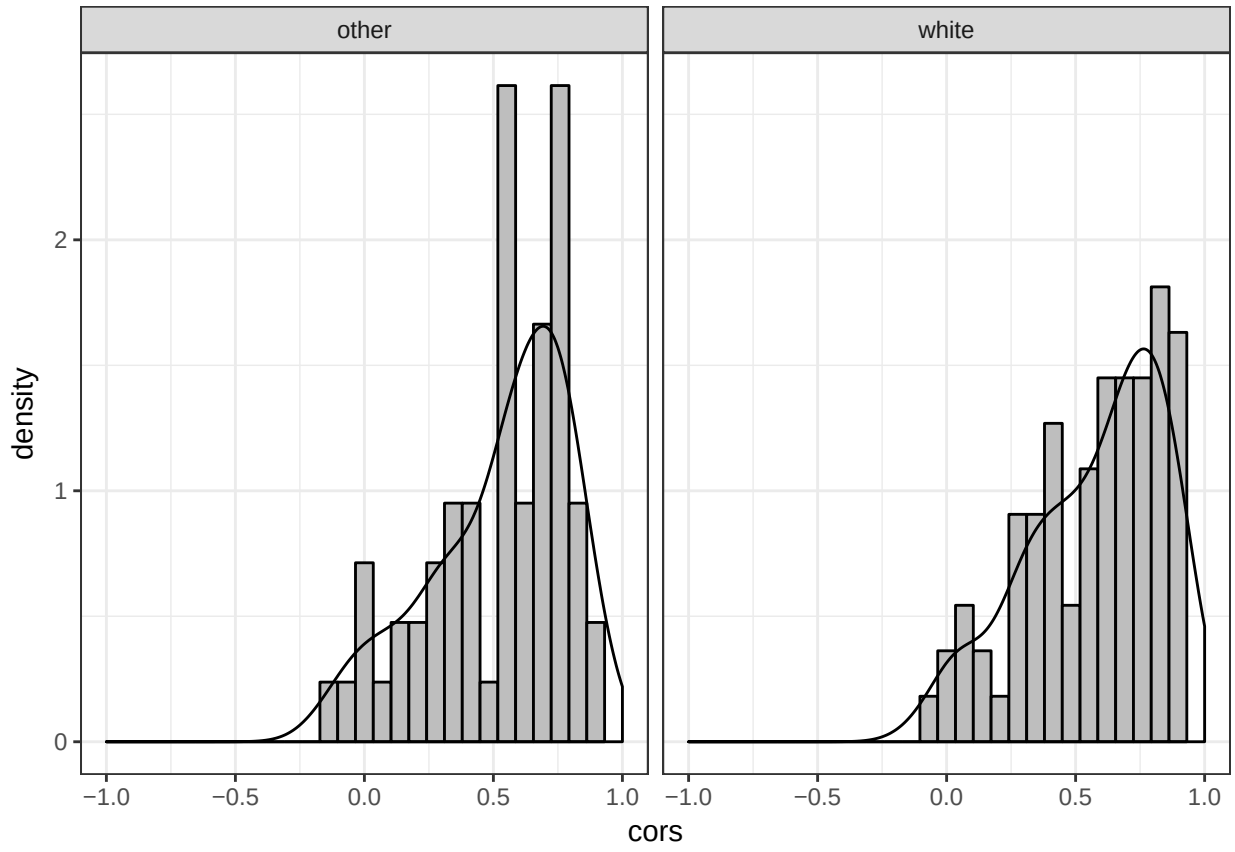
```r
save(gender_dat, class_dat, race_dat,
     file = "~/Box Sync/network/PAIRS/Brown Bag Presentation 3.31/moderator_mlVAR.RData")
```

5.3.2 Idiographic

```r
# Temporal
ip_cors %>%
 filter(!(is.na(race)) & !(is.na(cors)) & type2 == "Temporal") %>%
  ggplot(aes(x = cors)) +
    geom_histogram(aes(y = ..density..), color = "black", fill = "gray") +
    geom_density() +
    scale_x_continuous(limits = c(-1,1), breaks = seq(-1, 1,.5)) +
    facet_grid(.~race) +
    theme_bw()
```




```
# Contemporaneous
ip_cors %>%
 filter(!(is.na(race)) & !(is.na(cors)) & type2 == "Contemporaneous") %>%
  ggplot(aes(x = cors)) +
    geom_histogram(aes(y = ..density..), color = "black", fill = "gray") +
    geom_density() +
    scale_x_continuous(limits = c(-1,1), breaks = seq(-1, 1,.5)) +
    facet_grid(.~race) +
    theme_bw()
```


other
white
density
cors
0
1
2
−1.0
−0.5
0.0
0.5
1.0