Analyses

Profiles in the Applicability of Open Science Practices to Completed Research Projects Across Disciplines and Paradigms

[blinded]

2023-03-14

Contents

Packages & Setup

```
knitr::opts_chunk$set(
  warning = F, # don't show warnings during document generation
  message = F, # don't show messages during document generation
  error = F, # don't show errors during document generation
  echo = TRUE  # show R code
```

```
)

library(tidyverse)
library(viridis)
library(tidyLPA)
library(nnet)
library(kableExtra)
library(psych)
library(generics)
library(emmeans)
library(xfun)
library(here)
library(naniar)
library(stargazer)
library(ggpubr)
library(ggridges)

installed.packages()[names(sessionInfo()$otherPkgs), "Version"]
```

```
##    ggridges      ggpubr   stargazer      naniar        here        xfun
##     "0.5.4"     "0.6.0"     "5.2.3"     "1.0.0"     "1.0.1"      "0.37"
##     emmeans    generics       psych  kableExtra        nnet     tidyLPA
##     "1.8.2"     "0.1.3"     "2.2.9"     "1.3.4"    "7.3-18"     "1.1.0"
##     viridis viridisLite     forcats     stringr       dplyr       purrr
##     "0.6.2"     "0.4.1"     "1.0.0"     "1.5.0"     "1.1.0"     "1.0.1"
##       readr       tidyr      tibble     ggplot2   tidyverse
##     "2.1.3"     "1.3.0"     "3.1.8"     "3.4.1"     "1.3.2"
```

```
R.Version()
```

```
## $platform
## [1] "x86_64-w64-mingw32"
##
## $arch
## [1] "x86_64"
##
## $os
## [1] "mingw32"
##
## $crt
## [1] "ucrt"
##
## $system
## [1] "x86_64, mingw32"
##
## $status
## [1] ""
##
## $major
## [1] "4"
##
## $minor
## [1] "2.2"
##
```

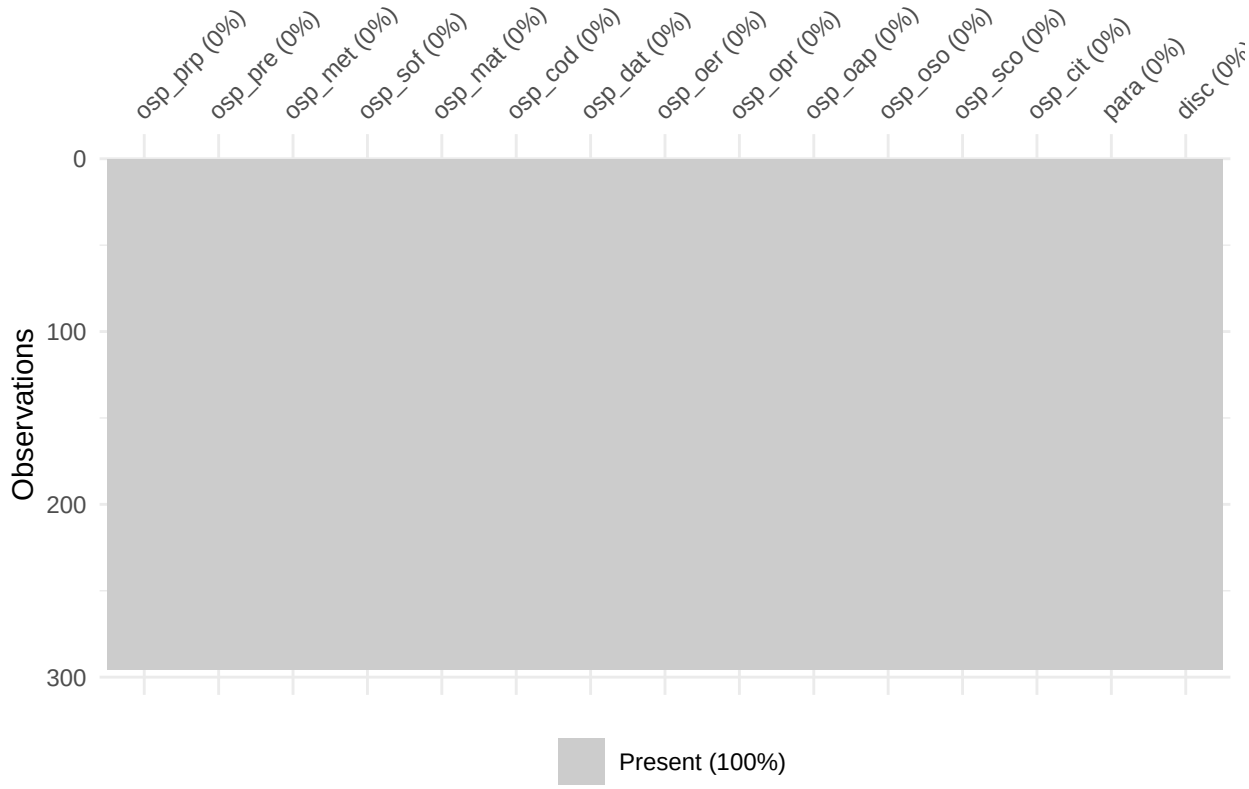
```
## $year
## [1] "2022"
##
## $month
## [1] "10"
##
## $day
## [1] "31"
##
## $`svn rev`
## [1] "83211"
##
## $language
## [1] "R"
##
## $version.string
## [1] "R version 4.2.2 (2022-10-31 ucrt)"
##
## $nickname
## [1] "Innocent and Trusting"
```

Data import

```
osc <- rio::import("https://zenodo.org/record/6834569/files/osc_data.csv") %>%
  dplyr::mutate(disc = disc_broad) %>%
  dplyr::select(-disc_broad)
```

Analysis of missingness

```
vis_miss(osc)
```



All items were set to mandatory, there are no missings in the data set.

Variable names and value labels

Below see the variable names and their description (not variable labels), as well as value labels.
The variable names are used throughout this document, for further details on the variables, see codebook on zenodo (not blinded).

```
names_labels <- data.frame(variable_name  = c(
                             "osp_cit",
                             "osp_cod",
                             "osp_dat",
                             "osp_mat",
                             "osp_met",
                             "osp_oap",
                             "osp_oer",
                             "osp_opr",
                             "osp_oso",
                             "osp_pre",
                             "osp_prp",
                             "osp_sco",
                             "osp_sof",
                             "disc",
                             "para"),
                           description =c(
                             "Citizen Science",
```

```
                              "Open Code",
                              "Open Data",
                              "Open Materials",
                              "Open Methodology",
                              "Open Access",
                              "OER",
                              "Open Peer Review",
                              "Open Source",
                              "Preregistration",
                              "Public Project Plan",
                              "Science Communication",
                              "Open Software",
                              "Discipline Cluster",
                              "Research paradigm"
                              ),
                            value_labels = c(
                              "1 = not applicable at all; 4 = highly applicable",
                              "1 = not applicable at all; 4 = highly applicable",
                              "1 = not applicable at all; 4 = highly applicable",
                              "1 = not applicable at all; 4 = highly applicable",
                              "1 = not applicable at all; 4 = highly applicable",
                              "1 = not applicable at all; 4 = highly applicable",
                              "1 = not applicable at all; 4 = highly applicable",
                              "1 = not applicable at all; 4 = highly applicable",
                              "1 = not applicable at all; 4 = highly applicable",
                              "1 = not applicable at all; 4 = highly applicable",
                              "1 = not applicable at all; 4 = highly applicable",
                              "1 = not applicable at all; 4 = highly applicable",
                              "1 = not applicable at all; 4 = highly applicable",
                              "agr = Agricultural and veterinary sciences, eng = Engineering and technol
                              "mixe = explicitly mixed-methodological (equally qualitative and quantitati
                            ))

names_labels %>%
  kbl() %>%
  kable_paper("hover")
```

Descriptive analyses

13 open science practices

```
descr <- describe(osc[,1:13])

descr %>%
  round(3) %>%
  kbl() %>%
  kable_paper("hover")
```

Research paradigm

```
table(osc$para) %>%
  round(3) %>%
```

variable_name	description	value_labels
osp_cit	Citizen Science	1 = not applicable at all; 4 = highly applicable
osp_cod	Open Code	1 = not applicable at all; 4 = highly applicable
osp_dat	Open Data	1 = not applicable at all; 4 = highly applicable
osp_mat	Open Materials	1 = not applicable at all; 4 = highly applicable
osp_met	Open Methodology	1 = not applicable at all; 4 = highly applicable
osp_oap	Open Access	1 = not applicable at all; 4 = highly applicable
osp_oer	OER	1 = not applicable at all; 4 = highly applicable
osp_opr	Open Peer Review	1 = not applicable at all; 4 = highly applicable
osp_oso	Open Source	1 = not applicable at all; 4 = highly applicable
osp_pre	Preregistration	1 = not applicable at all; 4 = highly applicable
osp_prp	Public Project Plan	1 = not applicable at all; 4 = highly applicable
osp_sco	Science Communication	1 = not applicable at all; 4 = highly applicable
osp_sof	Open Software	1 = not applicable at all; 4 = highly applicable
disc	Discipline Cluster	agr = Agricultural and veterinary sciences, eng = Engineering and technology
para	Research paradigm	mixe = explicitly mixed-methodological (equally qualitative and quantitative

	vars	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
osp_prp	1	295	2.837	1.021	3	2.920	1.483	1	4	3	-0.476	-0.900	0.059
osp_pre	2	295	2.508	1.100	3	2.511	1.483	1	4	3	-0.029	-1.324	0.064
osp_met	3	295	3.214	0.921	3	3.346	1.483	1	4	3	-0.927	-0.155	0.054
osp_sof	4	295	2.810	1.019	3	2.886	1.483	1	4	3	-0.327	-1.064	0.059
osp_mat	5	295	3.014	1.000	3	3.139	1.483	1	4	3	-0.739	-0.540	0.058
osp_cod	6	295	2.888	1.029	3	2.983	1.483	1	4	3	-0.486	-0.951	0.060
osp_dat	7	295	3.051	1.014	3	3.186	1.483	1	4	3	-0.667	-0.793	0.059
osp_oer	8	295	2.512	1.112	2	2.515	1.483	1	4	3	0.007	-1.354	0.065
osp_opr	9	295	2.871	1.039	3	2.962	1.483	1	4	3	-0.396	-1.092	0.060
osp_oap	10	295	3.088	1.016	3	3.232	1.483	1	4	3	-0.796	-0.577	0.059
osp_oso	11	295	2.281	1.183	2	2.228	1.483	1	4	3	0.280	-1.445	0.069
osp_sco	12	295	2.980	0.940	3	3.068	1.483	1	4	3	-0.498	-0.773	0.055
osp_cit	13	295	2.478	1.169	2	2.473	1.483	1	4	3	0.066	-1.474	0.068

Var1	Freq
mixe	119
none	12
qual	60
quan	104

Var1	Freq
agr	42
eng	50
hum	50
med	50
nat	52
soc	51

```
  kbl() %>%
  kable_paper("hover", full_width = F)
```

Discipline (broad classification)

```
table(osc$disc) %>%
  round(3) %>%
  kbl() %>%
  kable_paper("hover", full_width = F)
```

Latent profile analysis

Number of profiles

LPAs based on the 13 open science practices with different number of profiles (1 to 8). Solutions are compared using BIC.

```
osc %>%
  dplyr::select(osp_cit,
                osp_prp,
                osp_pre,
                osp_met,
                osp_sof,
                osp_mat,
                osp_cod,
                osp_dat,
                osp_oer,
                osp_opr,
                osp_oap,
                osp_oso,
                osp_sco) %>%
  mutate(across(osp_cit:osp_sco, as.numeric)) %>%
  estimate_profiles(1:8) %>%
  compare_solutions(statistics = c("BIC", "Entropy"))
```

```
## Compare tidyLPA solutions:
##
##  Model Classes BIC       Entropy
```

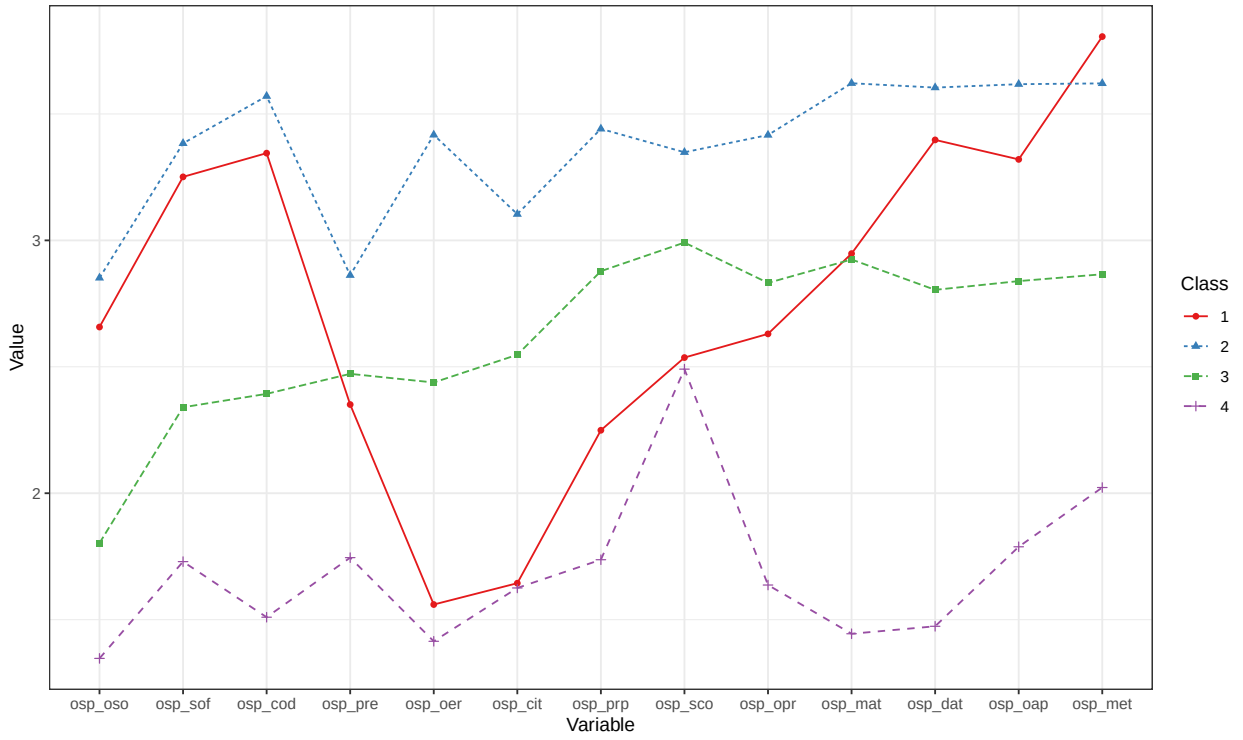
```
##  1     1       11323.689 1.000
##  1     2       10815.702 0.868
##  1     3       10706.692 0.829
##  1     4       10673.207 0.817
##  1     5       10672.462 0.789
##  1     6       10721.551 0.804
##  1     7       10775.950 0.802
##  1     8       10767.170 0.869
##
## Best model according to BIC is Model 1 with 5 classes.
## Best model according to Entropy is Model NA with NA classes.
##
## An analytic hierarchy process, based on the fit indices AIC, AWE, BIC, CLC, and KIC (Akogul & Erisogl
```

The BIC fit index is lowest for the solutions with 4 and 5 classes. For the 5 class solution, however, the entropy falls under the threshold of .80. We therefore decide for the 4 class solution with an acceptable entropy.

Characteristics of profiles

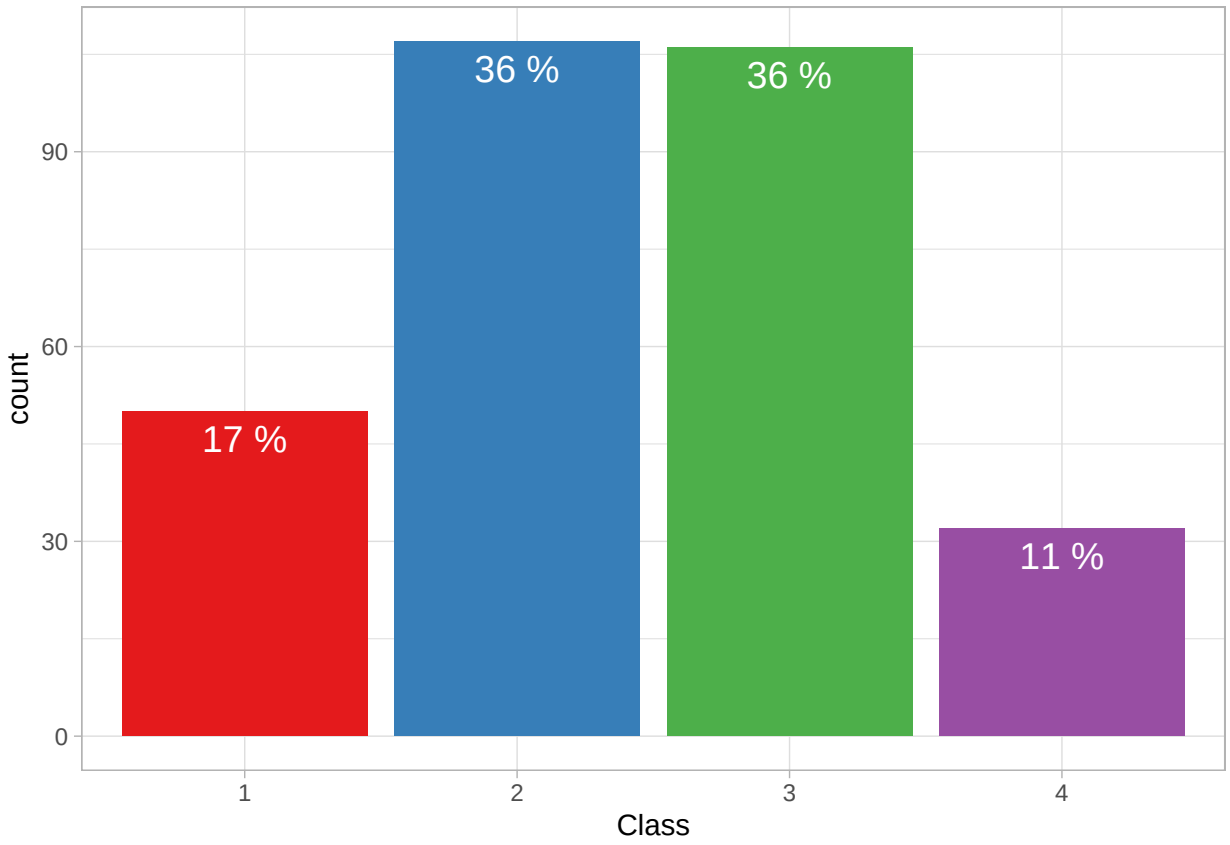
Line plot of profiles

```
osc_lpa <- osc %>%
  dplyr::select(osp_oso,
                osp_sof,
                osp_cod,
                osp_pre,
                osp_oer,
                osp_cit,
                osp_prp,
                osp_sco,
                osp_opr,
                osp_mat,
                osp_dat,
                osp_oap,
                osp_met) %>%
  mutate(across(osp_oso:osp_met, as.numeric)) %>%
  estimate_profiles(4)

osc_lpa%>%
  plot_profiles(ci = NULL,
                sd = F,
                add_line = T,
                rawdata = F)
```



Size of profiles

```
# get data from LPA
lpa_data <- get_data(osc_lpa)[,c(16:20)]
lpa_data <- cbind(osc, lpa_data) # bind with original data set


ggplot(lpa_data, aes(x=as.factor(Class), fill=as.factor(Class))) +
  # geom_bar() +
  geom_histogram(stat = "count") +
  geom_text(aes(label = paste(format((..count../295)*100,
                              digits=2,
                              drop0trailing=TRUE), "%"),
                y= ..count.. ), stat= "count", vjust = 1.5,
            size = 5, color = "white") +
  scale_fill_manual(values = c("#e41a1c", "#377eb8", "#4daf4a", "#984ea3")) +
  xlab("Class") +
  theme_light() +
  theme(legend.position = "none")
```

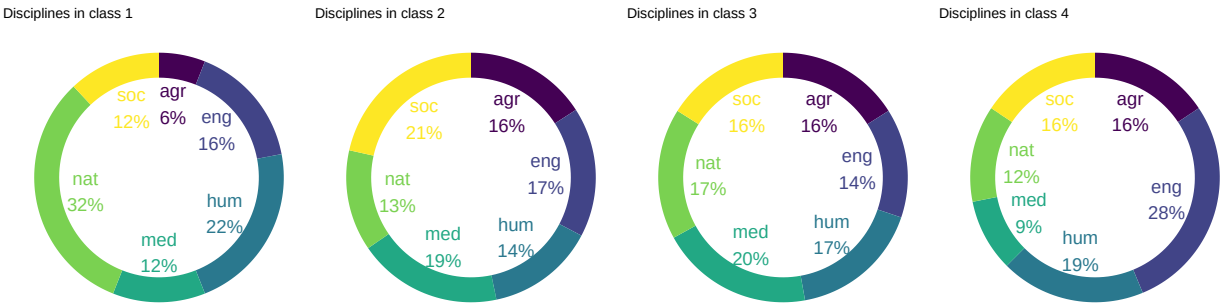


Donut plots

```
### donut plots ###################################################################

### disc ###
lpa_disc <- lpa_data %>%
  group_by(Class, disc) %>%
  dplyr::summarize(disc = disc[1],
                   disc_n = n()) %>%
  ungroup()

disc_N <- lpa_disc %>%
  group_by(Class) %>%
  summarize(N = sum(disc_n))

lpa_disc <- lpa_disc %>%
  mutate(fraction = case_when(
    Class == 1 ~ disc_n/disc_N$N[1],
    Class == 2 ~ disc_n/disc_N$N[2],
    Class == 3 ~ disc_n/disc_N$N[3],
    Class == 4 ~ disc_n/disc_N$N[4])) %>%
  group_by(Class) %>%
  mutate(ymax = cumsum(fraction),
         ymin = c(0, head(ymax, n=-1)),
         labelPosition = (ymax + ymin) / 2,
```

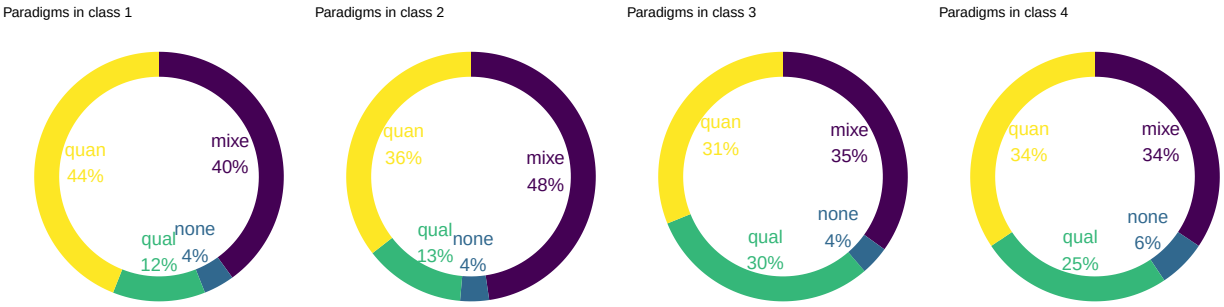
```
         label = paste0(disc, "\n", round(fraction * 100), "%"))



# plot for class 1
disc1 <-
ggplot(lpa_disc%>%filter(Class==1), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=disc)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=disc), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none") +
  ggtitle("Disciplines in class 1")


# plot for class 2
disc2 <-
ggplot(lpa_disc%>%filter(Class==2), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=disc)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=disc), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none")+
  ggtitle("Disciplines in class 2")

# plot for class 3
disc3 <-
ggplot(lpa_disc%>%filter(Class==3), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=disc)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=disc), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none")+
  ggtitle("Disciplines in class 3")

# plot for class 4
disc4 <-
ggplot(lpa_disc%>%filter(Class==4), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=disc)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=disc), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
```

```
  theme_void() +
  theme(legend.position = "none")+
  ggtitle("Disciplines in class 4")


ggarrange(disc1, disc2, disc3, disc4, ncol = 4, nrow = 1)
```




```
### disc ###
lpa_para <- lpa_data %>%
  group_by(Class, para) %>%
  dplyr::summarize(para = para[1],
                   para_n = n()) %>%
  ungroup()

para_N <- lpa_para %>%
  group_by(Class) %>%
  summarize(N = sum(para_n))

lpa_para <- lpa_para %>%
  mutate(fraction = case_when(
    Class == 1 ~ para_n/para_N$N[1],
    Class == 2 ~ para_n/para_N$N[2],
    Class == 3 ~ para_n/para_N$N[3],
    Class == 4 ~ para_n/para_N$N[4])) %>%
  group_by(Class) %>%
  mutate(ymax = cumsum(fraction),
         ymin = c(0, head(ymax, n=-1)),
         labelPosition = (ymax + ymin) / 2,
         label = paste0(para, "\n", round(fraction * 100), "%"))



# plot for class 1
para1 <-
ggplot(lpa_para%>%filter(Class==1), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=para)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=para), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
```

```
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none") +
  ggtitle("Paradigms in class 1")


# plot for class 2
para2 <-
ggplot(lpa_para%>%filter(Class==2), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=para)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=para), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none")+
  ggtitle("Paradigms in class 2")

# plot for class 3
para3 <-
ggplot(lpa_para%>%filter(Class==3), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=para)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=para), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none")+
  ggtitle("Paradigms in class 3")

# plot for class 4
para4 <-
ggplot(lpa_para%>%filter(Class==4), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=para)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=para), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none")+
  ggtitle("Paradigms in class 4")


ggarrange(para1, para2, para3, para4, ncol = 4, nrow = 1)
```



Predicting conditional probabilities

We predict the probability of class membership by the research paradigm and discipline of the research project. In case of significant results from a predictor, we additionally compute contrasts.

Predicting Class 1

```
# predicting conditional probability to be in class 1
fit_predict_class1 <- lm(scale(CPROB1) ~ as.factor(para) + as.factor(disc),
                         data = lpa_data)
summary(fit_predict_class1)
```

```
##
## Call:
## lm(formula = scale(CPROB1) ~ as.factor(para) + as.factor(disc),
##     data = lpa_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.06845 -0.54308 -0.38955 -0.04897  2.70817
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -0.29400    0.16372  -1.796  0.07358 .
## as.factor(para)none -0.07968    0.30085  -0.265  0.79133
## as.factor(para)qual -0.20250    0.15774  -1.284  0.20029
## as.factor(para)quan  0.17157    0.13231   1.297  0.19577
## as.factor(disc)eng   0.25256    0.20591   1.227  0.22100
## as.factor(disc)hum   0.43896    0.20883   2.102  0.03643 *
## as.factor(disc)med   0.11791    0.20637   0.571  0.56820
## as.factor(disc)nat   0.64324    0.20406   3.152  0.00179 **
## as.factor(disc)soc   0.15829    0.20520   0.771  0.44109
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9826 on 286 degrees of freedom
## Multiple R-squared:  0.06071,    Adjusted R-squared:  0.03444
## F-statistic: 2.311 on 8 and 286 DF,  p-value: 0.02049
```

```r
# pairwise comparisons
fit_predict_class1_em <- emmeans(fit_predict_class1, "para")
pwpm(fit_predict_class1_em)
```

```
##           mixe      none      qual      quan
## mixe [-0.0255]    0.9935    0.5741    0.5658
## none    0.0797 [-0.1052]    0.9794    0.8412
## qual    0.2025    0.1228 [-0.2280]    0.0979
## quan   -0.1716   -0.2512   -0.3741 [ 0.1461]
##
## Row and column labels: para
## Upper triangle: P values   adjust = "tukey"
## Diagonal: [Estimates] (emmean)
## Lower triangle: Comparisons (estimate)   earlier vs. later
```

```r
# fit_predict_class1_em <- emmeans(fit_predict_class1, "disc")
# pwpm(fit_predict_class1_em)
```

Predicting Class 2

```r
# predicting conditional probability to be in class 2
fit_predict_class2 <- lm(scale(CPROB2) ~ as.factor(para) + as.factor(disc),
                         data = lpa_data)
summary(fit_predict_class2)
```

```
##
## Call:
## lm(formula = scale(CPROB2) ~ as.factor(para) + as.factor(disc),
##     data = lpa_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.1653 -0.7942 -0.5275  1.1580  1.8873
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)           0.12988    0.16551   0.785   0.4332
## as.factor(para)none -0.12844    0.30413  -0.422   0.6731
## as.factor(para)qual -0.36913    0.15947  -2.315   0.0213 *
## as.factor(para)quan -0.14098    0.13376  -1.054   0.2928
## as.factor(disc)eng  -0.06711    0.20816  -0.322   0.7474
## as.factor(disc)hum  -0.07362    0.21111  -0.349   0.7276
## as.factor(disc)med   0.15227    0.20863   0.730   0.4661
## as.factor(disc)nat  -0.22497    0.20629  -1.091   0.2764
## as.factor(disc)soc   0.21875    0.20744   1.055   0.2925
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9934 on 286 degrees of freedom
## Multiple R-squared:  0.04008,    Adjusted R-squared:  0.01323
## F-statistic: 1.493 on 8 and 286 DF,  p-value: 0.1594
```

```r
# # pairwise comparisons
# fit_predict_class2_em <- emmeans(fit_predict_class2, "para")
# pwpm(fit_predict_class2_em)
```

```
# fit_predict_class2_em <- emmeans(fit_predict_class2, "disc")
# pwpm(fit_predict_class2_em)
```

Predicting Class 3

```
# predicting conditional probability to be in class 3
fit_predict_class3 <- lm(scale(CPROB3) ~ as.factor(para) + as.factor(disc),
                         data = lpa_data)
summary(fit_predict_class3)
```

```
##
## Call:
## lm(formula = scale(CPROB3) ~ as.factor(para) + as.factor(disc),
##     data = lpa_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4120 -0.7460 -0.5831  1.0791  1.7697
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)           0.11703    0.16510   0.709  0.47901
## as.factor(para)none   0.04090    0.30339   0.135  0.89285
## as.factor(para)qual   0.44551    0.15908   2.801  0.00545 **
## as.factor(para)quan  -0.03949    0.13343  -0.296  0.76750
## as.factor(disc)eng   -0.29467    0.20765  -1.419  0.15698
## as.factor(disc)hum   -0.22557    0.21059  -1.071  0.28503
## as.factor(disc)med   -0.12286    0.20811  -0.590  0.55544
## as.factor(disc)nat   -0.17782    0.20579  -0.864  0.38826
## as.factor(disc)soc   -0.31837    0.20693  -1.539  0.12502
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9909 on 286 degrees of freedom
## Multiple R-squared:  0.04478,    Adjusted R-squared:  0.01806
## F-statistic: 1.676 on 8 and 286 DF,  p-value: 0.1039
```

```
# pairwise comparisons
# fit_predict_class3_em <- emmeans(fit_predict_class3, "para")
# pwpm(fit_predict_class3_em)
fit_predict_class3_em <- emmeans(fit_predict_class3, "disc")
pwpm(fit_predict_class3_em)
```

```
##            agr        eng        hum        med        nat        soc
## agr [ 0.22876]     0.7155     0.8925     0.9916     0.9547     0.6396
## eng     0.2947 [-0.06591]     0.9994     0.9549     0.9914     1.0000
## hum     0.2256    -0.0691 [ 0.00319]     0.9957     0.9999     0.9972
## med     0.1229    -0.1718    -0.1027 [ 0.10590]     0.9998     0.9209
## nat     0.1778    -0.1168    -0.0477     0.0550 [ 0.05094]     0.9795
## soc     0.3184     0.0237     0.0928     0.1955     0.1406 [-0.08961]
##
## Row and column labels: disc
## Upper triangle: P values   adjust = "tukey"
## Diagonal: [Estimates] (emmean)
```

```
## Lower triangle: Comparisons (estimate)   earlier vs. later
```

Predicting Class 4

```
# predicting conditional probability to be in class 4
fit_predict_class4 <- lm(scale(CPROB4) ~ as.factor(para) + as.factor(disc),
                         data = lpa_data)
summary(fit_predict_class4)
```

```
##
## Call:
## lm(formula = scale(CPROB4) ~ as.factor(para) + as.factor(disc),
##     data = lpa_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7958 -0.4183 -0.3099 -0.1999  3.1809
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -0.02189    0.16716  -0.131    0.896
## as.factor(para)none  0.22130    0.30717   0.720    0.472
## as.factor(para)qual  0.14588    0.16106   0.906    0.366
## as.factor(para)quan  0.06801    0.13509   0.503    0.615
## as.factor(disc)eng   0.22592    0.21024   1.075    0.283
## as.factor(disc)hum  -0.07252    0.21322  -0.340    0.734
## as.factor(disc)med  -0.18457    0.21071  -0.876    0.382
## as.factor(disc)nat  -0.14872    0.20836  -0.714    0.476
## as.factor(disc)soc  -0.05353    0.20951  -0.256    0.799
##
## Residual standard error: 1.003 on 286 degrees of freedom
## Multiple R-squared:  0.02078,    Adjusted R-squared:  -0.006607
## F-statistic: 0.7588 on 8 and 286 DF,  p-value: 0.6395
```

```
# pairwise comparisons
fit_predict_class4_em <- emmeans(fit_predict_class4, "para")
pwpm(fit_predict_class4_em)
```

```
##            mixe       none       qual       quan
## mixe [-0.06080]     0.8889     0.8018     0.9582
## none    -0.2213 [ 0.16050]     0.9953     0.9602
## qual    -0.1459     0.0754 [ 0.08508]     0.9653
## quan    -0.0680     0.1533     0.0779 [ 0.00721]
##
## Row and column labels: para
## Upper triangle: P values   adjust = "tukey"
## Diagonal: [Estimates] (emmean)
## Lower triangle: Comparisons (estimate)   earlier vs. later
```

```
# fit_predict_class4_em <- emmeans(fit_predict_class4, "disc")
# pwpm(fit_predict_class4_em)
```

[additional: as preregistered]

latent class models (2 to 7)

A loop to compute latent class models with 2 to 7 classes.

```
library(poLCA)
library(ggridges)

f <- with(osc, cbind(osp_cit,
                     osp_prp,
                     osp_pre,
                     osp_met,
                     osp_sof,
                     osp_mat,
                     osp_cod,
                     osp_dat,
                     osp_oer,
                     osp_opr,
                     osp_oap,
                     osp_oso,
                     osp_sco) ~ 1)

# run a sequence of models with 1-7 classes and print out the model with the lowest BIC
max_II <- -100000
min_bic <- 100000
for(i in 2:7){
  lc <- poLCA(f, osc,
              nclass=i,
              maxiter=3000,
              tol=1e-5,
              na.rm=FALSE,
              nrep=10,
              verbose=TRUE,
              calc.se=TRUE)
  if(lc$bic < min_bic){
    min_bic <- lc$bic
    LCA_best_model<-lc
  }
}
```

Extracting the best model

```
# extracting best model
LCA_best_model
```

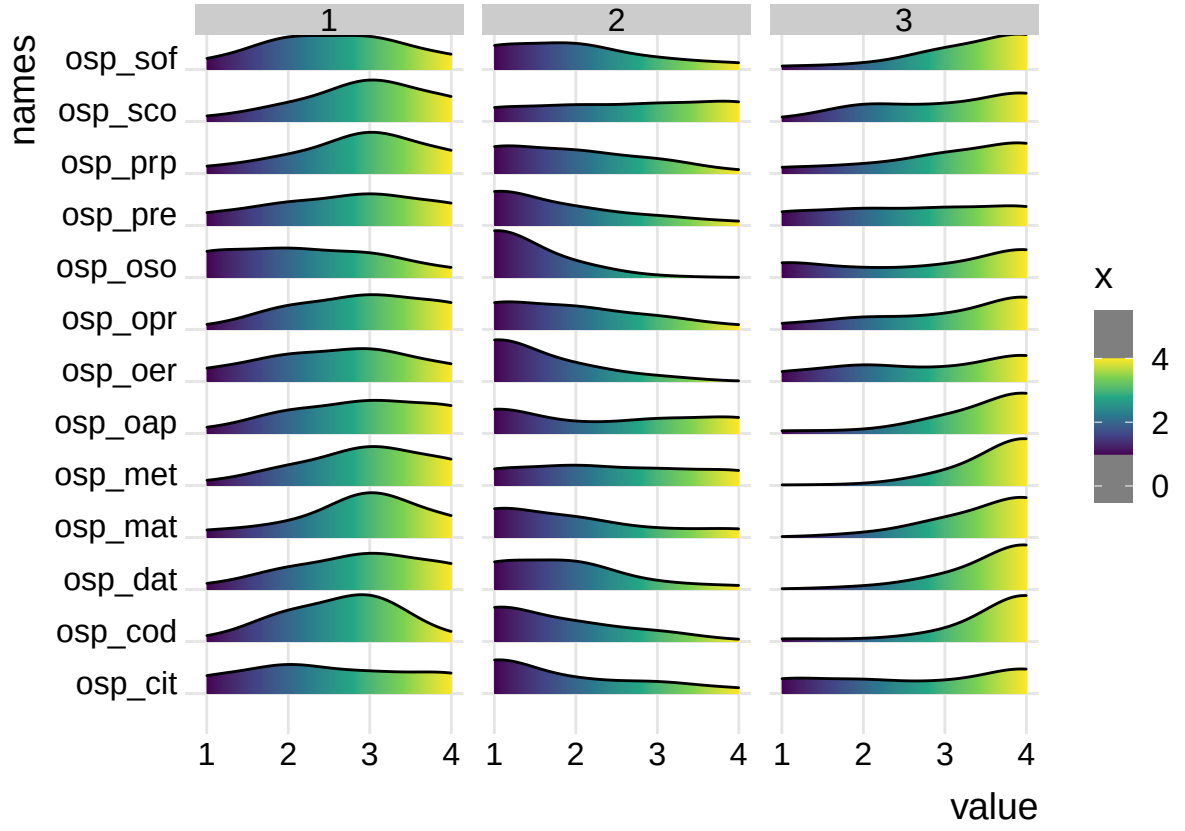
```
## Conditional item response (column) probabilities,
##  by outcome variable, for each class (row)
##
## $osp_cit
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.1888 0.3283 0.2342 0.2487
```

```
## class 2:  0.5906 0.1840 0.1556 0.0698
## class 3:  0.2286 0.2063 0.1577 0.4074
##
## $osp_prp
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.0788 0.1945 0.4935 0.2332
## class 2:  0.4228 0.3134 0.2052 0.0585
## class 3:  0.0883 0.1230 0.2925 0.4962
##
## $osp_pre
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.1509 0.2549 0.3525 0.2417
## class 2:  0.5545 0.2467 0.1423 0.0565
## class 3:  0.2000 0.2399 0.2494 0.3108
##
## $osp_met
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.0390 0.2075 0.4408 0.3127
## class 2:  0.2481 0.2765 0.2388 0.2366
## class 3:  0.0119 0.0239 0.1818 0.7824
##
## $osp_sof
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.0972 0.3750 0.3745 0.1534
## class 2:  0.3640 0.3760 0.1485 0.1116
## class 3:  0.0521 0.0739 0.2994 0.5745
##
## $osp_mat
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.0726 0.1479 0.5588 0.2208
## class 2:  0.4823 0.2803 0.1104 0.1270
## class 3:  0.0101 0.0600 0.2561 0.6738
##
## $osp_cod
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.0426 0.3332 0.5391 0.0851
## class 2:  0.5448 0.2687 0.1574 0.0292
## class 3:  0.0400 0.0379 0.1553 0.7668
##
## $osp_dat
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.0516 0.2459 0.4153 0.2872
## class 2:  0.4199 0.4173 0.0998 0.0630
## class 3:  0.0086 0.0406 0.1844 0.7663
##
## $osp_oer
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.1512 0.2957 0.3634 0.1897
## class 2:  0.6742 0.2444 0.0814 0.0000
## class 3:  0.1394 0.2484 0.1798 0.4325
##
## $osp_opr
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.0358 0.2639 0.3956 0.3047
```

```
## class 2:  0.4269 0.3184 0.1874 0.0672
## class 3:  0.0814 0.1909 0.1843 0.5434
##
## $osp_oap
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.0446 0.2592 0.3668 0.3294
## class 2:  0.4085 0.1380 0.2144 0.2391
## class 3:  0.0463 0.0376 0.2369 0.6793
##
## $osp_oso
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.3173 0.3126 0.2631 0.1070
## class 2:  0.7884 0.1910 0.0206 0.0000
## class 3:  0.2351 0.1271 0.1771 0.4608
##
## $osp_sco
##            Pr(1)  Pr(2)  Pr(3)  Pr(4)
## class 1:  0.0557 0.1892 0.4792 0.2760
## class 2:  0.2080 0.2417 0.2508 0.2995
## class 3:  0.0369 0.2610 0.2372 0.4649
##
## Estimated class population shares
##  0.4247 0.1737 0.4016
##
## Predicted class memberships (by modal posterior prob.)
##  0.4169 0.1763 0.4068
##
## =========================================================
## Fit for 3 latent classes:
## =========================================================
## number of observations: 295
## number of estimated parameters: 119
## residual degrees of freedom: 176
## maximum log-likelihood: -4620.588
##
## AIC(3): 9479.177
## BIC(3): 9917.927
## G^2(3): 5899.724 (Likelihood ratio/deviance statistic)
## X^2(3): 152205664 (Chi-square goodness of fit)
##
```

```r
# append class to data set
osc <- data.frame(osc, class = LCA_best_model$predclass)

# plotting density of participants in each class as ridges
osc_plot <- osc %>%
  dplyr::select(-c(disc, para)) %>%
  pivot_longer(1:13, names_to = "names", values_to = "value")

ggplot(osc_plot, aes(x=value, y=names, fill = stat(x))) +
  geom_density_ridges_gradient(scale = 0.9, bandwidth = .5) +
  scale_fill_viridis_c(values = c(.25,.75), option = "D") +
  scale_x_continuous(limits = c(1,4), breaks = c(1,2,3,4)) +
  theme_ridges() +
```

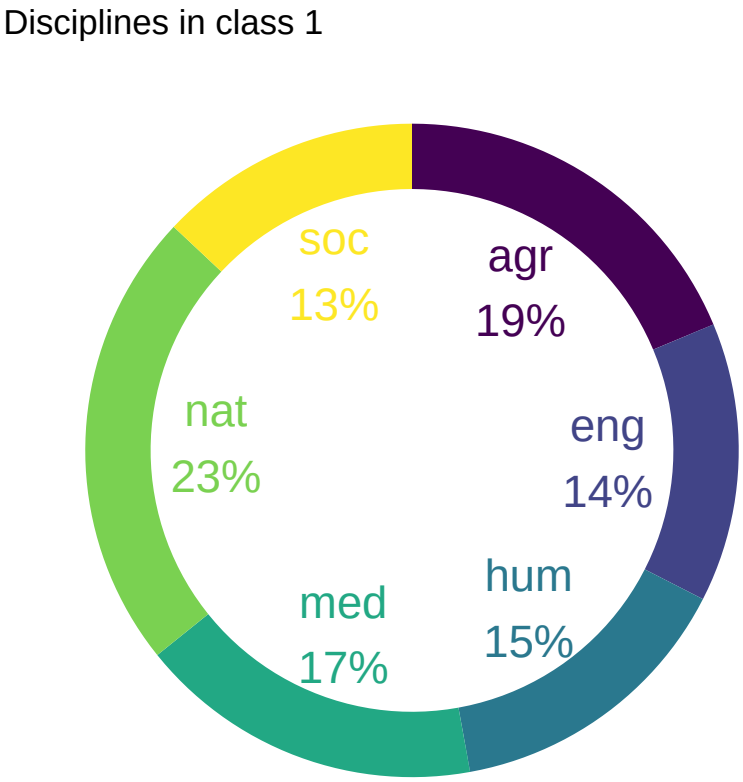
```
  facet_wrap(~class)
```



donut plots

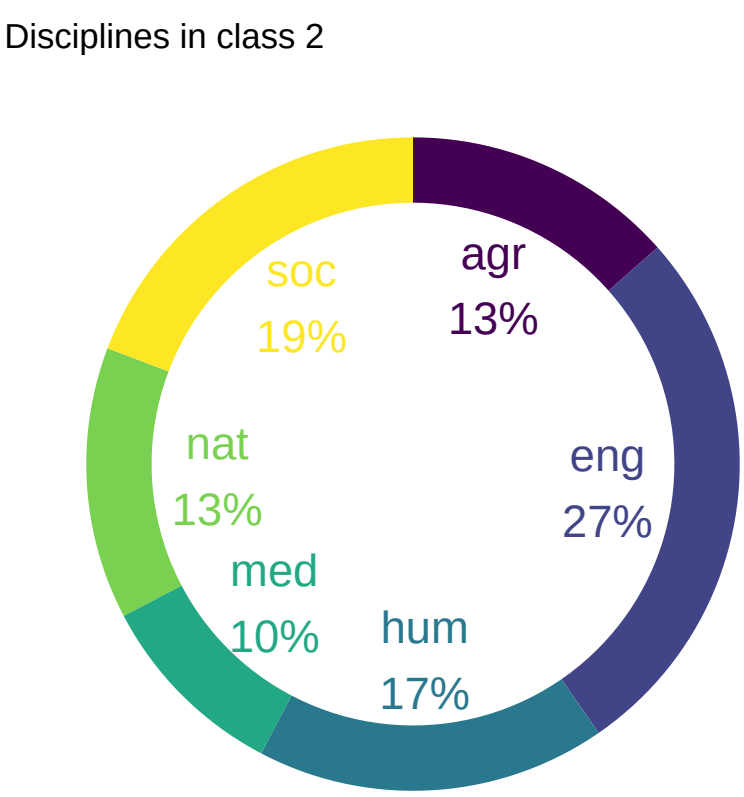
```
### donut plots ############### #
# disc
osc_disc <- osc %>%
  group_by(class, disc) %>%
  summarize(disc = disc[1],
            disc_n = n()) %>%
  ungroup()

disc_N <- osc_disc %>%
  group_by(class) %>%
  summarize(N = sum(disc_n))

osc_disc <- osc_disc %>%
  mutate(fraction = case_when(
                        class == 1 ~ disc_n/disc_N$N[1],
                        class == 2 ~ disc_n/disc_N$N[2],
                        class == 3 ~ disc_n/disc_N$N[3])) %>%
  group_by(class) %>%
  mutate(ymax = cumsum(fraction),
         ymin = c(0, head(ymax, n=-1)),
         labelPosition = (ymax + ymin) / 2,
```

```
        label = paste0(disc, "\n", round(fraction * 100), "%"))



# plot for class 1
ggplot(osc_disc%>%filter(class==1), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=disc)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=disc), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none") +
  ggtitle("Disciplines in class 1")
```

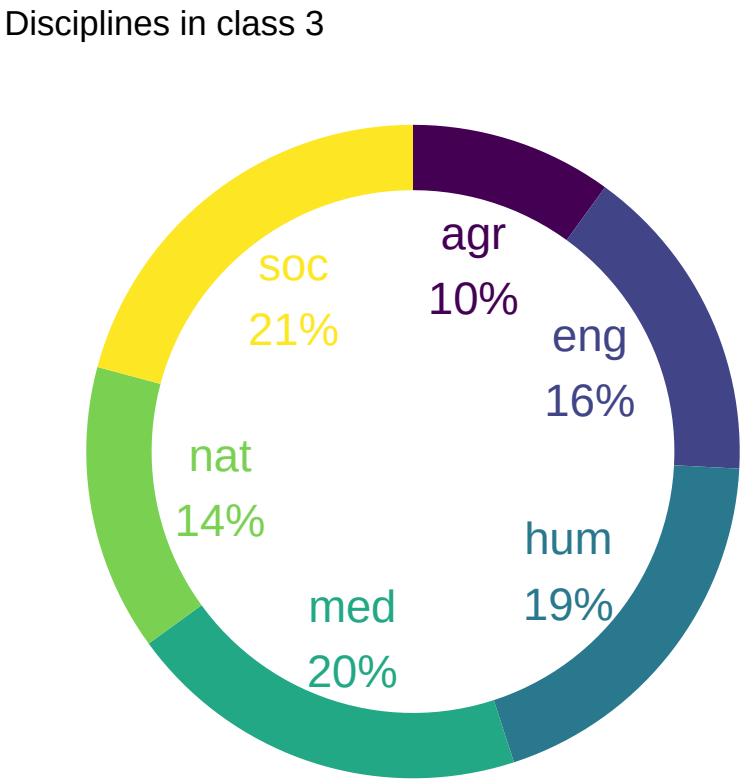



```
# plot for class 2
ggplot(osc_disc%>%filter(class==2), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=disc)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=disc), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none")+
```

```
  ggtitle("Disciplines in class 2")
```

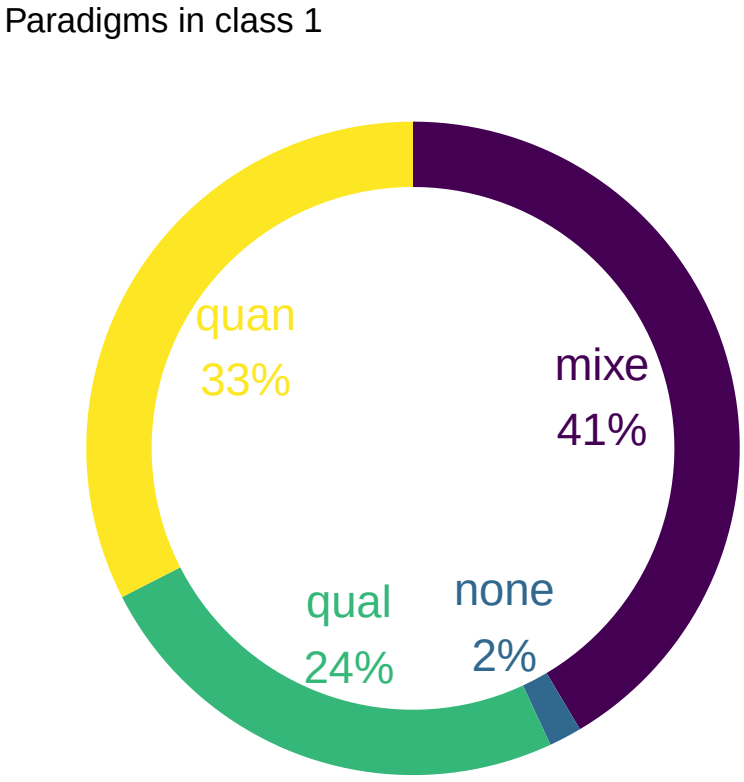



```
# plot for class 3
ggplot(osc_disc%>%filter(class==3), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=disc)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=disc), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none")+
  ggtitle("Disciplines in class 3")
```

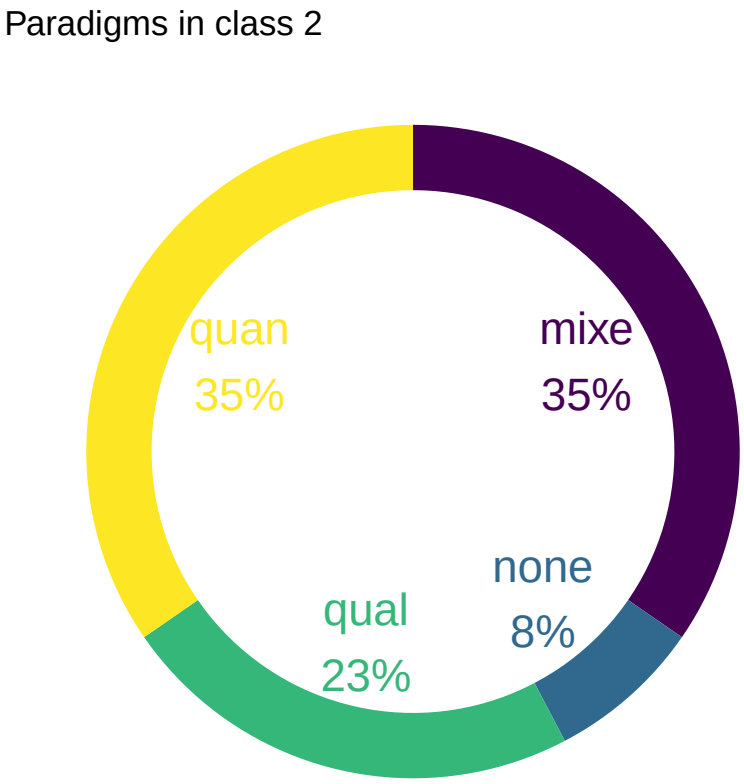



```
# para
osc_para <- osc %>%
  group_by(class, para) %>%
  summarize(para = para[1],
            para_n = n()) %>%
  ungroup()

para_N <- osc_para %>%
  group_by(class) %>%
  summarize(N = sum(para_n))

osc_para <- osc_para %>%
  mutate(fraction = case_when(
    class == 1 ~ para_n/para_N$N[1],
    class == 2 ~ para_n/para_N$N[2],
    class == 3 ~ para_n/para_N$N[3])) %>%
  group_by(class) %>%
  mutate(ymax = cumsum(fraction),
         ymin = c(0, head(ymax, n=-1)),
         labelPosition = (ymax + ymin) / 2,
         label = paste0(para, "\n", round(fraction * 100), "%"))

#plot for class 1
ggplot(osc_para%>%filter(class==1), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=para)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=para), size=6) + # x here controls label posi
```

```
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none") +
  ggtitle("Paradigms in class 1")
```




```
#plot for class 2
ggplot(osc_para%>%filter(class==2), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=para)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=para), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none") +
  ggtitle("Paradigms in class 2")
```




```
#plot for class 3
ggplot(osc_para%>%filter(class==3), aes(ymax=ymax, ymin=ymin, xmax=4, xmin=3, fill=para)) +
  geom_rect() +
  geom_text( x=2, aes(y=labelPosition, label=label, color=para), size=6) + # x here controls label posi
  scale_fill_viridis_d() +
  scale_color_viridis_d() +
  coord_polar(theta="y") +
  xlim(c(-1, 4)) +
  theme_void() +
  theme(legend.position = "none") +
  ggtitle("Paradigms in class 3")
```

Paradigms in class 3