Problem Set 1

Data Visualisation for Social Scientists

Due: January 28, 2026

Instructions

- Please show your work! You may lose points by simply writing in the answer. If the problem requires you to execute commands in `R`, please include the code you used to get your answers. Please also include the `.R` file that contains your code. If you are not sure if work needs to be shown for a particular problem, please ask.
- Your homework should be submitted electronically on GitHub.
- This problem set is due before 23:59 on Wednesday January 28, 2026. No late assignments will be accepted.

Roll Call Votes in the European Parliament

Data Manipulation

First, you need to download data from the first six elected European Parliaments on each MEP and how they voted in each recorded roll-call vote.

1. Load these datasets into your global environment:
   - `mep_info_26Jul11.xls` (MEP characteristics, EP1–EP5)
   - `rcv_ep1.txt` (EP1 roll-call votes)

   First step is to set up and load necessary packages

```
lapply(c("tidyverse", "ggplot2", "readr", "readxl", "purrr", "ggridges"),
       pkgTest)
```

   For the Information on MEP's explore .xls file and see that there are many sheets from which we wish to extract the info from the 5 parliamentary years. Use a function to extract the relevant sheets and store the data, checking important factors like the dimensions are numeric. Rename and tidy into a dataframe

```r
## Load Data to Global environment ##
excel_sheets('mep_info_26Jul11.xls') #look at the different sheets in xls
EPs <- c("EP1", "EP2", "EP3", "EP4", "EP5") #names of sheets we care to load
MEP_info <-  lapply( #apply to multiple
  EPs,
  function(i) { #function to combine each year from respective sheets
    readxl::read_excel('mep_info_26Jul11.xls', sheet = i) %>% #each i is in EPs
      mutate(
        `NOM-D1`= as.numeric(`NOM-D1`), #ensure specified as numeric info
        `NOM-D2` = as.numeric(`NOM-D2`)
      ) %>%
      mutate(EP = i) #add name
    }
)
names(MEP_info) <- EPs #name the elements as years
MEP_df <- bind_rows(MEP_info) #transofrm to dataframe
```

Read in 'Rolling Call Vote' files for each EP, as they are .txt files deliminated by commas, use read delim and specify encoding to make sure characters appear correctly.

```r
#Read in RCV
RCV1 <- read_delim("rcv_ep1.txt", delim = ",", locale = locale(encoding = "UTF-8"))
RCV2 <- read_delim("rcv_ep2.txt", delim = ",", locale = locale(encoding = "UTF-8"))
RCV3 <- read_delim("rcv_ep3.txt", delim = ",", locale = locale(encoding = "UTF-8"))
RCV4 <- read_delim("rcv_ep4.txt", delim = ",", locale = locale(encoding = "UTF-8"))
RCV5 <- read_delim("rcv_ep5.txt", delim = ",", locale = locale(encoding = "UTF-8"))
```

2. Briefly describe (2–3 sentences each) the unit of analysis and key variables in each of these two datasets.

   The MEP info dataset has detailed information about 3222 MEP's in the European parliament, identified by an MEP Id number, and their name. There is additional information such as a letter code for their member state (which of the 15 EU countries they come from), National Party a four digit code relating to their specific political party (out of 169 parties) and an 'EP group' code, a letter that relates to a one of 13 identified groups of similar political leaning/ideology. It also includes each MEP's NOMINATE coordinates (stored as numerical coordinates for each of the two dimensions) on a scale from -1 to +1.

   The RCV1 dataset stores voting data for the first elected European Parliament (1979-1984). It contains a row for each MEP with some metadata similar to the MEP info (e.g. name, id, National party etc.) as well as how they vote in every EP vote (V1-V886) coded as a single number that corresponds to a specific outcome (i.e. 'Yes', 'No'

'Absent' 'Abstain' etc.). RCV2-5 follow a similar format/content.

3. The `rcv_ep1` data are in a wide format, with V1, V2, ..., Vn as separate vote columns.

   - Identify which columns are ID/metadata (*MEPID, MEPNAME, MS, NP, EPG*) and which columns are vote decisions ($V_1 \ldots V_n$). Tidy the voting data such that each row/observation is a single vote for a single MEP.

     All but the first five columns are vote decisions. Therefore to tidy the day use pivot longer with all but the first 5 columns, to create a new variable 'Votes' that refers to the specific vote (V1,...Vn)and 'Outcome' (coded 0:5)that is the code for the MEP action.

```
## Tidy Votes into long format ##
RCV1 <- RCV1 %>% pivot_longer(cols = -(1:5), names_to = "Votes",
    values_to = "Outcome")
RCV2 <- RCV2 %>% pivot_longer(cols = -(1:5), names_to = "Votes",
    values_to = "Outcome")
RCV3 <- RCV3 %>% pivot_longer(cols = -(1:5), names_to = "Votes",
    values_to = "Outcome")
RCV4 <- RCV4 %>% pivot_longer(cols = -(1:5), names_to = "Votes",
    values_to = "Outcome")
RCV5 <- RCV5 %>% pivot_longer(cols = -(1:5), names_to = "Votes",
    values_to = "Outcome")
```

   - Create a summary table of counts of decision categories (e.g. Yes/No/Abstain/Present but did not vote/Absent) across all votes.
     Next step is to create categorical variable of 'Outcome' that refers to the MEP's behaviour. As the code for EP 3 and 4 are different (according to source) use two different coding schemes to create the new variable. By using mutate, we are able to create a new 'OutcomeCategory' variable that has the decision as the word e.g. 'Yes'

```
##  MEP Decision Categories ##
decision125 <- c( #from the data link
  "0" = "Absent",
  "1" = "Yes",
  "2" = "No",
  "3" = "Abstain",
  "4" = "Present but no vote",
  "5" = "Not an MEP")

decision34 <- c( #from the data link the codes for EP 3/4
  "0" = "Absent or Not an MEP",
  "1" = "Yes",
  "2" = "No",
  "3" = "Abstain",
  "4" = "Present but no vote")

```

```r

RCV1 <- RCV1 %>% mutate(Outcome_Category =  decision125[as.character(Outcome)])
RCV2 <- RCV2 %>% mutate(Outcome_Category =  decision125[as.character(Outcome)])
RCV3 <- RCV3 %>% mutate(Outcome_Category =  decision34[as.character(Outcome)])
RCV4 <- RCV4 %>% mutate(Outcome_Category =  decision34[as.character(Outcome)])
RCV5 <- RCV5 %>% mutate(Outcome_Category =  decision125[as.character(Outcome)])
#creates new categorical variable that codes the outcome referencing decision code
```

Create a count table for EP 1, that sees the number of outcomes of different types across all the votes. Using dplyr to group by OutcomeCategory then presenting a table with the counts in each category.

```r
#Summary EP 1 of Counts across all votes
RCV1 %>%
  group_by(Outcome_Category) %>% #different groups to count
  summarise(count = n()) %>% #count of each subgroup
  arrange(desc(count)) #descending order to see popularity
```

```
A tibble: 6 × 2
Outcome_Category     count
<chr>                <int>
1 Present but no vote 109224
2 Not an MEP          103618
3 Absent               99753
4 Yes                  88185
5 No                   75171
6 Abstain               9577
```

From this we can see in the first European Parliment that most of the time MEP are present but do not vote, are not and MEP or are absent.

4. Construct a new dataset that combines MEP-level information with their vote decisions from EP1 in long format (from part 3). Check for missingness.
First collect all RCV dataframes together by binding rows, change the name of 'MEP id' from MEP info dataframe so that they can be matched when combining. Join the dataframes together using left join that matching common key (the EP and Member Id) such that the additional info on the Nominate Dimensions are added to the correct MEPs.

```r
## Contruct new combined dataset ##
#All EP's
RCV_all <- bind_rows(
```

```
  RCV1 %>% mutate(EP = "EP1"),#create new variable with codes for EP
    session
  RCV2 %>% mutate(EP = "EP2"),
  RCV3 %>% mutate(EP = "EP3"),
  RCV4 %>% mutate(EP = "EP4"),
  RCV5 %>% mutate(EP = "EP5"))

# Rename ID to match and join data
MEP_df <- MEP_df %>% rename('MEPID' =`MEP id`) #make them identical to
    combine

RCV_com <- RCV_all %>%
  left_join(MEP_df %>% select(EP, `MEPID`, `NOM-D1`,`NOM-D2`), #merge
    these variables
            by = c("EP", "MEPID")) #common key is ID and EP to mesh them
    together
```

Use summarise to check for missing data across the whole new dataset. Can see some evidence of missing values.

```
#Check for NA's
print(RCV_com %>%
  summarise(across(everything(), ~sum(is.na(.)))))
```

```
A tibble: 1 × 11
MEPID MEPNAME    MS    NP   EPG Votes Outcome Outcome_Category  Year `NOM-D1` `NOM
<int>   <int> <int> <int> <int> <int>   <int>            <int> <int>    <int>    <
1     0       0     0     0     0     0       0             2733     0   202844
```

Tidy data by removing them and seeing about 20,000 removed.

```
#See the starting rows
RCV_com %>% summarise(n_rows = nrow(.))

#Remove any rows with na()
RCV_com <- RCV_com %>%
  tidyr::drop_na()

#Check new entries
RCV_com %>% summarise(n_rows = nrow(.))
```

```
A tibble: 1 × 1
n_rows
<int>
1 10172184

# A tibble: 1 × 1
n_rows
```

```
<int>
1 9968833
```

5. Compute, for each EP group in EP1:

   First of all select EP1, group the data by EP group and see roughly how many votes fall into each category.

```r
##  Comparison of EP Groups in EP1 ##
EP1_grouped <- RCV_com %>%  #make a group just of EP1
  filter(EP == "EP1") %>% #work with first parliment only
  group_by(EPG)

EP1_grouped %>%
  summarise(count = n()) %>%  #count the total number of votes/
    decisions in each group
  arrange(desc(count)) #see most populated group
```

- The mean rate of Yes votes (Yes over Yes+No+Abstain) across all roll calls.

```r
# Yes rate #
EP1_grouped %>%
  summarise( #create a Yes_rate variable summimg up outcome
    categories (as grouped EPG)
     Yes_Rate = sum(Outcome_Category == "Yes") / sum(Outcome_Category
    %in% c("Yes", "No", "Abstain"))
     )
```

```
# A tibble: 8 × 2
EPG   Yes_Rate
<chr>    <dbl>
1 C        0.415
2 E        0.509
3 G        0.517
4 L        0.487
5 M        0.529
6 N        0.582
7 R        0.457
8 S        0.576
```

Can see that group N has the highest Yes rate with group C the lowest.

- The mean abstention rate.

```r
# Abstain Rate #
EP1_grouped %>%
  summarise( #repeat for abstain rate
```

```
    Abstain_Rate = sum(Outcome_Category == "Abstain") / n() # divall
    categories
    )
```

```
# A tibble: 8 × 2
EPG   Abstain_Rate
<chr>        <dbl>
1 C          0.0468
2 E          0.00970
3 G          0.0141
4 L          0.0200
5 M          0.0249
6 N          0.0103
7 R          0.0505
8 S          0.0214
```

Can see much lower rates of Abstaining with group E abstaining the least.

- The mean vote preferences along the two contested dimensions (NOM-D1 and NOM-D2).
  Look at Yes, No and Abstain by filtering the outcome category and finding the mean of each dimensions within these subgroups. This gives insight into the coordinates of Nominate dimension for each group within category outcomes. Can see that the dimensions averages within groups for different outcomes is similar but slightly different (e.g. group M average D1 for Yes is -.284 for Yes but -.311 for No). May give insight within groups as to voting behaviour of MEP's.

```
# 150 EP1 Mean preference along dimensions NOM-D1 and NOM-D2 #
EP1_grouped %>%
  filter(Outcome_Category == "Yes") %>% #isolate just within those
   with yes outcome
  summarise(
    mean_D1 = mean(`NOM-D1`), #average dimension at mean for each EP
    group
    mean_D2 = mean(`NOM-D2`)
  )
```

```
# A tibble: 8 × 3
EPG   mean_D1 mean_D2
<chr>   <dbl>   <dbl>
1 C       0.811    0.532
2 E       0.510   -0.267
3 G       0.287   -0.823
4 L       0.419   -0.305
5 M      -0.284   -0.149
6 N       0.186   -0.179
```

```
7 R      -0.501   -0.150
8 S      -0.0921   0.387
```

'No' dimension averages

```
EP1_grouped %>%
  filter(Outcome_Category == "Abstain") %>% #and abstain
  summarise(
    mean_D1 = mean(`NOM-D1`),
    mean_D2 = mean(`NOM-D2`)
    )
```

```
# A tibble: 8 × 3
EPG   mean_D1 mean_D2
<chr>   <dbl>   <dbl>
1 C       0.811   0.530
2 E       0.516  -0.269
3 G       0.292  -0.811
4 L       0.422  -0.300
5 M      -0.311  -0.155
6 N       0.237  -0.242
7 R      -0.607  -0.0533
8 S      -0.0860  0.399
```

Abstain preferences

```
EP1_grouped %>%
  filter(Outcome_Category == "Abstain") %>% #and abstain
  summarise(
    mean_D1 = mean(`NOM-D1`),
    mean_D2 = mean(`NOM-D2`)
    )
```

```
# A tibble: 8 × 3
EPG   mean_D1 mean_D2
<chr>   <dbl>   <dbl>
1 C       0.811   0.526
2 E       0.495  -0.268
3 G       0.286  -0.807
4 L       0.421  -0.276
5 M      -0.343  -0.111
6 N       0.146  -0.0589
7 R      -0.710  -0.0126
8 S      -0.106   0.351
```

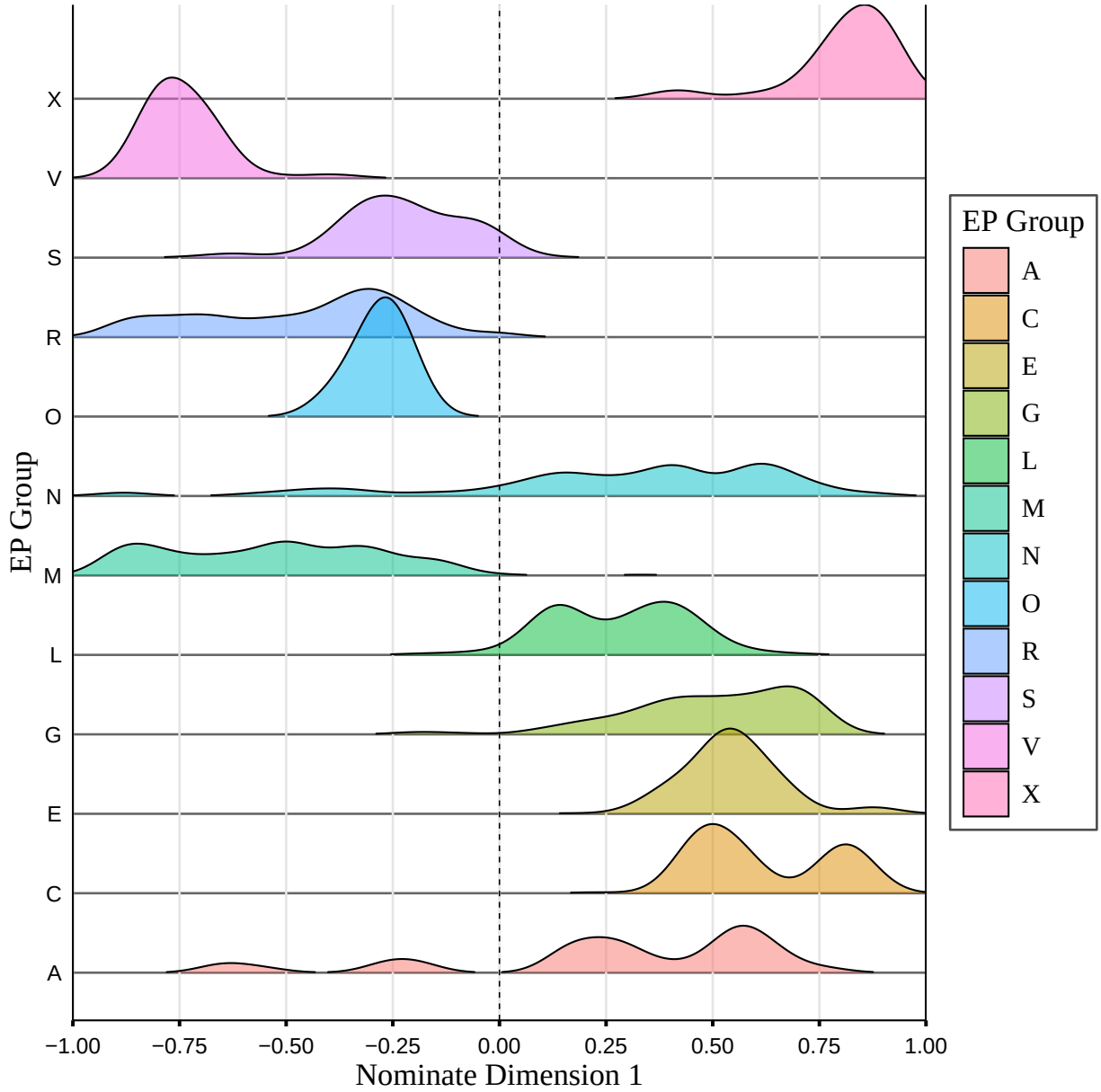
Data Visualization

1. Plot the distribution of the first NOMINATE dimension by EP group, and explain any trends you see.
   Aim to create dataset of MEP's grouped by EPG and use a density ridge plot to show the distribution of MEP's within different EP groups on the distribution of D1 scores. Can get insight into how similar/skewed the preferences of each group are and where they lie on the scale from -1 to 1.

```
##————————— Plot 1 —————————## 175

MEPs <- RCV_com %>% #make a dataframe for just the MEP's (not longform votes)
  group_by(MEPID, EP, EPG) %>% #capture id, group and parliment
  summarise(
    D1 = first(`NOM-D1`),  #capture the first value of D1
    D2 = first(`NOM-D2`)
  ) %>%
  ungroup() #ungroup to explore as whole set

pdf("PS01_Fig1.pdf")
ggplot(MEPs, aes(y = EPG, x = D1 , fill = EPG)) +
  # https://cran.r-project.org/web/packages/ggridges/vignettes/introduction.html
  geom_density_ridges( #to see multiple density plots in one
    color = "black", #line colour
    alpha = 0.5, #transparency
    scale = 1.5,
    rel_min_height = 0.005,#hight of distribution
    linewidth = 0.4 #size of line
  ) +
  scale_x_continuous(limits = c(-1, 1), #start and end at dimension max/min
                     breaks = seq(-1 ,1 ,0.25), #sequence for x axis
                     minor_breaks = NULL, #reduce some of clustered grid
                     expand = c(0,0)) + #stop grid going beyond limits
  geom_vline(xintercept = 0, linetype = "dashed", color = "black", size = 0.3) + #add line at zero
  geom_vline(xintercept = 1, color = "black", size = 0.5) + #add axis at +1
  theme_minimal() +
  labs(x = "Nominate Dimension 1", y = "EP Group", fill = "EP Group") + #labels
  theme(
    axis.title = element_text(family = "serif", size = 14), #title font
    axis.line.x = element_line(size = 0.4), #axis lines
    axis.line.y = element_line(size = 0.4),
    axis.text = element_text(colour = "black", size = 10), #axis text change
    axis.ticks.x = element_line(colour = "black", size = 0.75),
```

```
35     panel.grid.major.y = element_line(colour = "grey40"), #change horiz
       grid to stronger
36     panel.grid.major.x = element_line(colour = "grey90"), #lighter
       vertical gridlines
37     legend.title = element_text(family = 'serif', size = 14, colour = '
       black'), #legend
38     legend.text =  element_text(family = 'serif', size = 12, colour = '
       black'),# font change
39     legend.key.height = unit(1.5, "lines"),#spacing
40     legend.key.width = unit(1.5, "lines"),
41     legend.spacing.y = unit(1, "lines"),
42     legend.background = element_rect(
43       fill = "white",
44       colour = "grey30", #make box aound legend to align it
```

Figure 1: MEP Nominate Dimension 1 by EP Group



Notes: From MEP's across the first five European Parliaments.

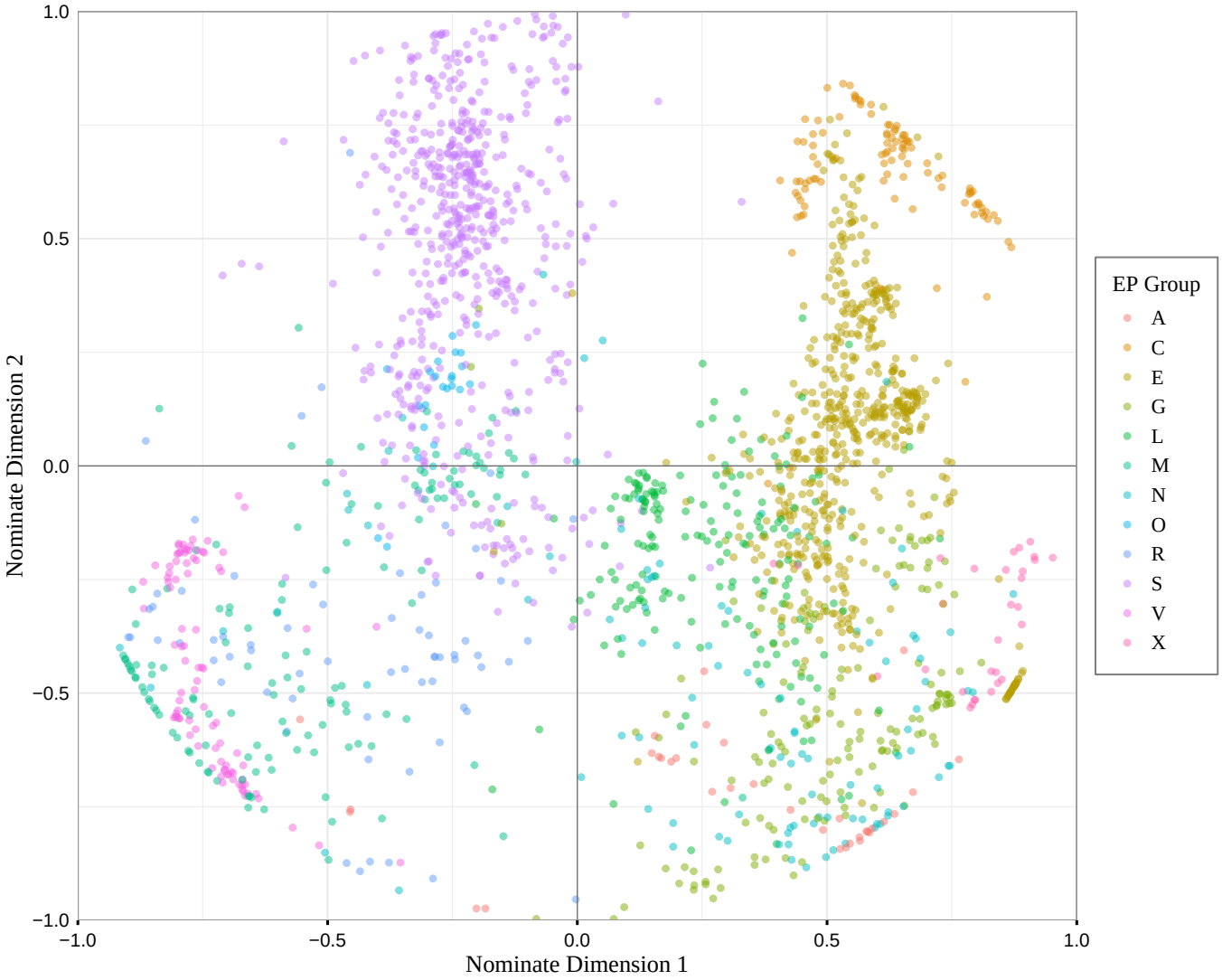
From this we can see group X and V are the most clearly polarised at the top and bottom of the dimension respectively. Groups S, R, O and M generally have MEP's scoring negatively on the dimension whereas N, L, G, E, C and A tend to contain MEPs with positive scores on the dimension. Group N appears to have the widest range of MEP's on the dimension. EP groups O, and E have the most 'normal' distributions, Group C and L appear to have bimodal distributions, whilst group A seems to be tetramodal.

2. Make a scatterplot of *nomdim1* (x-axis) and *nomdim2* (y-axis), with one point per MEP and color by EP group.

```
##————————— Plot 2 —————————## 225

MEP_ind <- MEPs %>%  #isolate MEP individuals (so MEP's in multiple EP's appear once
  group_by(MEPID, EPG) %>% #one id per mep
  summarise(D1 = mean(D1),D2 = mean(D2)) %>% #summarise their dimension as mean (if many)
  ungroup()

pdf("PS01_Fig2.pdf", width = 10, height = 8)
ggplot(MEP_ind, aes(x = D1, y = D2, color = EPG))+ #plot each individual MEP, colour by EPG
  geom_point(alpha = 0.5, size = 1.5, na.rm = TRUE) + #add points to make scatter
  geom_hline(yintercept = 0, color = "grey40", size = 0.3) + #add grid axis at 0,0
  geom_vline(xintercept = 0,color = "grey40", size = 0.3) +
  geom_vline(xintercept = 1,color = "grey40", size = 0.3) +
  scale_x_continuous(limits = c(-1,1), breaks = seq(-1,1, 0.5), expand = c(0,0)) + #make scale fit -1 to +1
  scale_y_continuous(limits = c(-1,1), breaks = seq(-1,1, 0.5), expand = c(0,0)) + #same again
  labs(x = "Nominate Dimension 1",y = "Nominate Dimension 2", colour = "EP Group") +
  theme_minimal()+
  theme(
    panel.border = element_rect(colour = "grey60", fill = NA, size = 0.5),#add grey border
    axis.title = element_text(family = "serif", size = 14), #title font
    axis.text = element_text(colour = "black", size = 11), #axis text change
    axis.ticks.x = element_line(colour = "black", size = 0.75),
    legend.title = element_text(family = 'serif', size = 13, colour = 'black'), #legend
    legend.text =  element_text(family = 'serif', size = 12, colour = 'black'),# font change
    legend.position = "right", #position right
    legend.margin = margin(10,10,10,10), #add space around
    legend.background = element_rect(
      fill = "white",
```

```
        colour = "grey50", #make box aound legend to align it
        size = 0.5,
        linetype = "solid")
  )
dev.off()
```

Figure 2: MEP's Nominate Coordinates



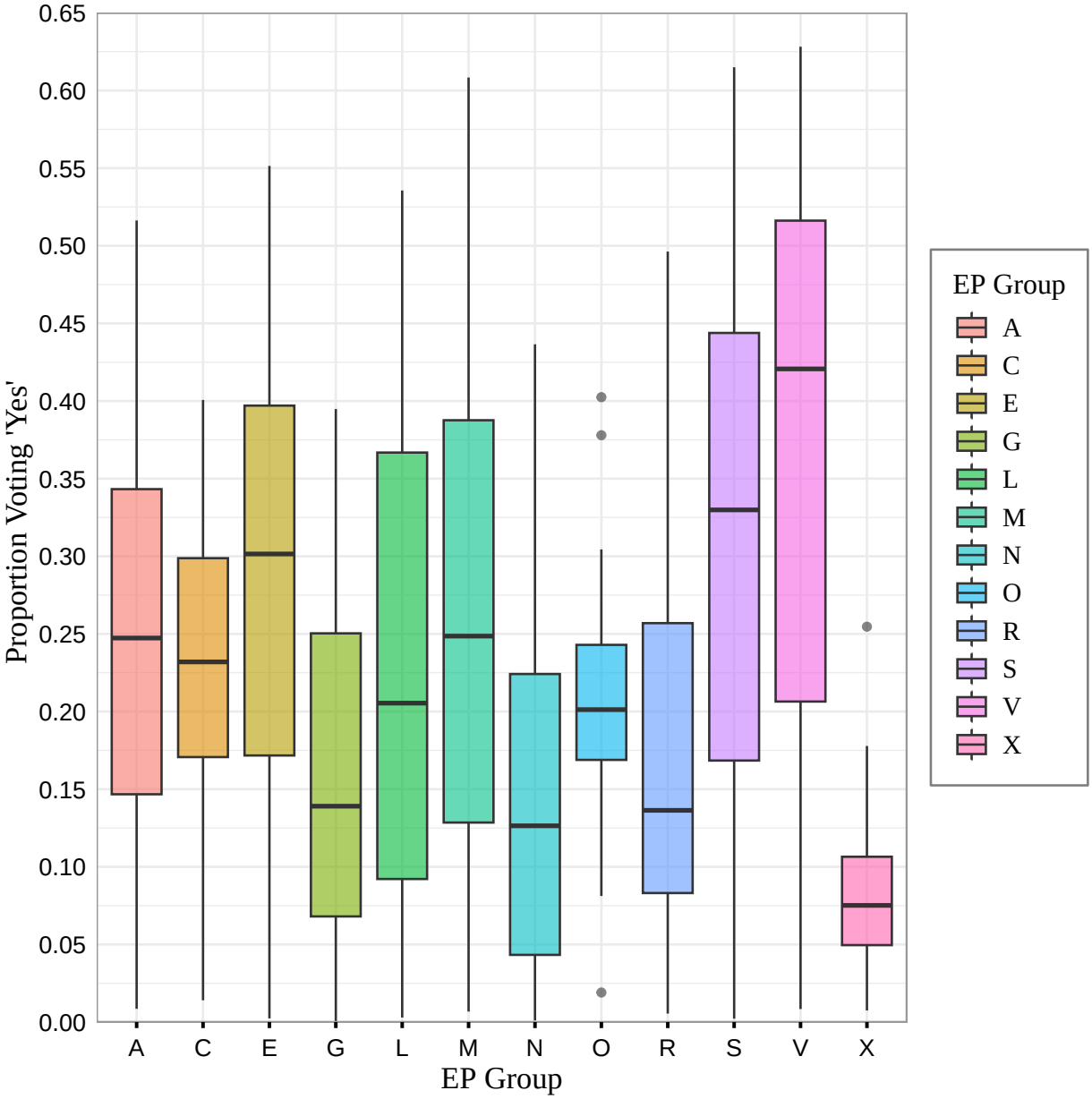
Notes: 3099 MEP's from first five European Parliaments, for MEP's elected for multiple Parliments, their average dimension scores were plotted.

3. Produce a boxplot of the proportion voting *Yes* by EP group to visualize cohesion. Calculating a Yes rate (of yes outcomes divided by total votes) and plotting the pro-

portion voting yes by group in a boxplot clearly displays the differences in yes rates across groups. It also shows the spread difference ()e.g. V vs X) as to how similar MEP's yes rates are within the EP group.

```
##---------- Plot 3 ----------## 260
table(RCV_com$NP)

Yes_rate <- RCV_com %>% #make a Yes rate
    group_by(MEPID, EPG) %>% #grouping by id and EP group
    summarise(total_votes = n(),#list of group as votes in long form
              yes_votes = sum(`Outcome_Category` == "Yes"), #sum within
   them of yes
              yes_rate = yes_votes / total_votes #yes rate
              )

pdf("PS01_Fig3.pdf")
ggplot(Yes_rate, aes(x=EPG, y = yes_rate, fill = EPG)) +#boxplot by group
    , yes rate on y
  geom_boxplot(alpha = 0.6) +
  labs(x = "EP Group",y = "Proportion Voting 'Yes'", fill = "EP Group" )
   +
  scale_y_continuous(limits = c(0,0.65), breaks = seq(0, 0.65, 0.05),
   expand = c(0,0)) + #fitting boundaries
  theme_minimal() +
  theme(
    panel.border = element_rect(colour = "grey60", fill = NA, size = 0.5)
    ,#box round
    axis.title = element_text(family = "serif", size = 14), #title font
    axis.text = element_text(colour = "black", size = 11), #axis text
   change
    axis.ticks.x = element_line(colour = "black", size = 0.75),
    legend.title = element_text(family = 'serif', size = 13, colour = '
   black'), #legend
    legend.text =  element_text(family = 'serif', size = 12, colour = '
   black'),# font change
    legend.position = "right",
    legend.margin = margin(10,10,10,10),
    legend.background = element_rect(
      fill = "white",
      colour = "grey50", #make box around legend to align it
      size = 0.5,
      linetype = "solid")
  )
dev.off()
```

Figure 3: Proportion Voting 'Yes' by EP Group



Notes: Yes rate for each individual MEP in the first five European Parliaments. Calculated as how many times they vote 'Yes' by total number of votes opened for them.

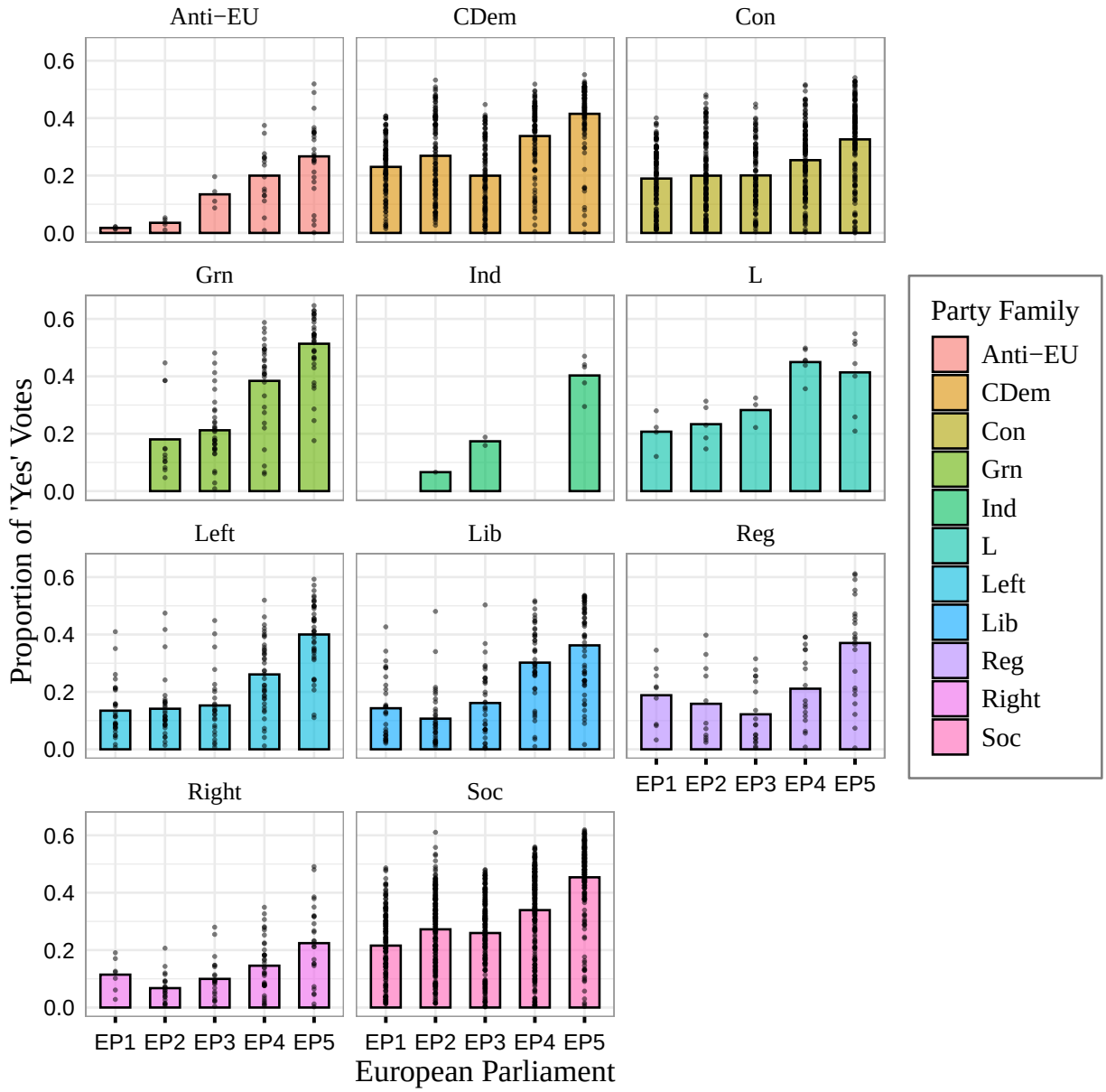
Can see rough grouping of MEP's in groups especially group S, C, G and V. Looking at how group tended to have MEP's with similar coordinate of NOMINATE factor or spread may give insights into groups cohesion or spread in parliament.

4. Display the proportion voting *Yes* per year by national party using a bar plot.
As there are over 100 National parties coded in the dataset, it made sense to read in the more descriptive 'Party Family' with 11 categories corresponding to the NP code to better evaluate using a bar plot. Combining the NP family, and creating a average yes rate for each national party per EP to plot points as well as the average of these averages to create a bar for the 'family' average of all included National parties. I made a mixed type plot with bars as well as the NP average points to clearly visualise the difference over time within each family as well as compare the party family types.

```
  ##---------- Plot 4 ----------## 250
length(unique(RCV_com$NP)) #163 parties

NP_Family <- readxl::read_excel('mep_info_26Jul11.xls', sheet = 'Codes-
    National Parties') %>% #
  select('Code', 'Party Family') %>% #extract info on party family
  rename(NP = `Code`, #code of party
         party_family = `Party Family`)
unique(NP_Family$party_family) #11 'National Party families'

RCV_com <- RCV_com %>%
  left_join(NP_Family, by = "NP") #add back to the dataframe, matching NP
    code

Yes_rate_NP_EP <- RCV_com %>% #yes rate for each party for each year
  group_by(MEPID, NP, party_family, EP) %>% #grouping
  summarise(total_votes = n(),
            yes_votes = sum(`Outcome_Category` == "Yes"),
            yes_rate_NP_EP = yes_votes / total_votes)

Yes_rate_NP_family <- Yes_rate_NP_EP %>% #grouping by family of parties
  group_by(EP, party_family) %>% #get mean rate of national parties
    within
  summarise(mean_yes_rate_NP_EP = mean(yes_rate_NP_EP, .groups = "drop"))

pdf("PS01_Fig4.pdf")
ggplot() + #initialise empty plot (as doing both points and bar)
  geom_col(data = Yes_rate_NP_family ,#use family level data for bar
           aes(x = factor(EP), y = mean_yes_rate_NP_EP, fill = party_
    family), #colour family
           color = "black", width = 0.6, alpha = 0.6) +
  geom_point(data = Yes_rate_NP_EP, #for points use average party rate
    for each year
             aes(x = factor(EP), y = yes_rate_NP_EP), #plug in from
    above
           size = 0.4, alpha = 0.5) +
  facet_wrap(~ party_family, ncol = 3) + #wrap to get subplot for each
```

```
    party_family, w 3 cols
  scale_y_continuous(limits = c(0, 0.65)) + #restrict limits to see
    patterm more clearly
  labs(x = "European Parliament", y = "Proportion of 'Yes' Votes", fill =
    "Party Family") +
  theme_minimal()+
  theme(
    panel.border = element_rect(colour = "grey60", fill = NA, size = 0.5)
    ,
    axis.title = element_text(family = "serif", size = 14), #title font
    axis.text = element_text(colour = "black", size = 10), #axis text
    change
    axis.ticks.x = element_line(colour = "black", size = 0.75),
    strip.text = element_text(family = 'serif', size = 11, colour = '
    black'),
    legend.title = element_text(family = 'serif', size = 13, colour = '
    black'), #legend
    legend.text =  element_text(family = 'serif', size = 12, colour = '
    black'),# font change
    legend.position = "right",
    legend.margin = margin(10,10,10,10),
    legend.background = element_rect(
      fill = "white",
      colour = "grey50", #make box around legend to align it
      size = 0.5,
      linetype = "solid")
  )
dev.off()
```

Figure 4: Average Yes Share National Party's MEPs, over the first five European parliments by Party Family Group



Notes: Points represent a National Parties average 'Yes' proportion in that European Parliament. Bar represents the mean of the parties in the Party Family for each European Parliament.

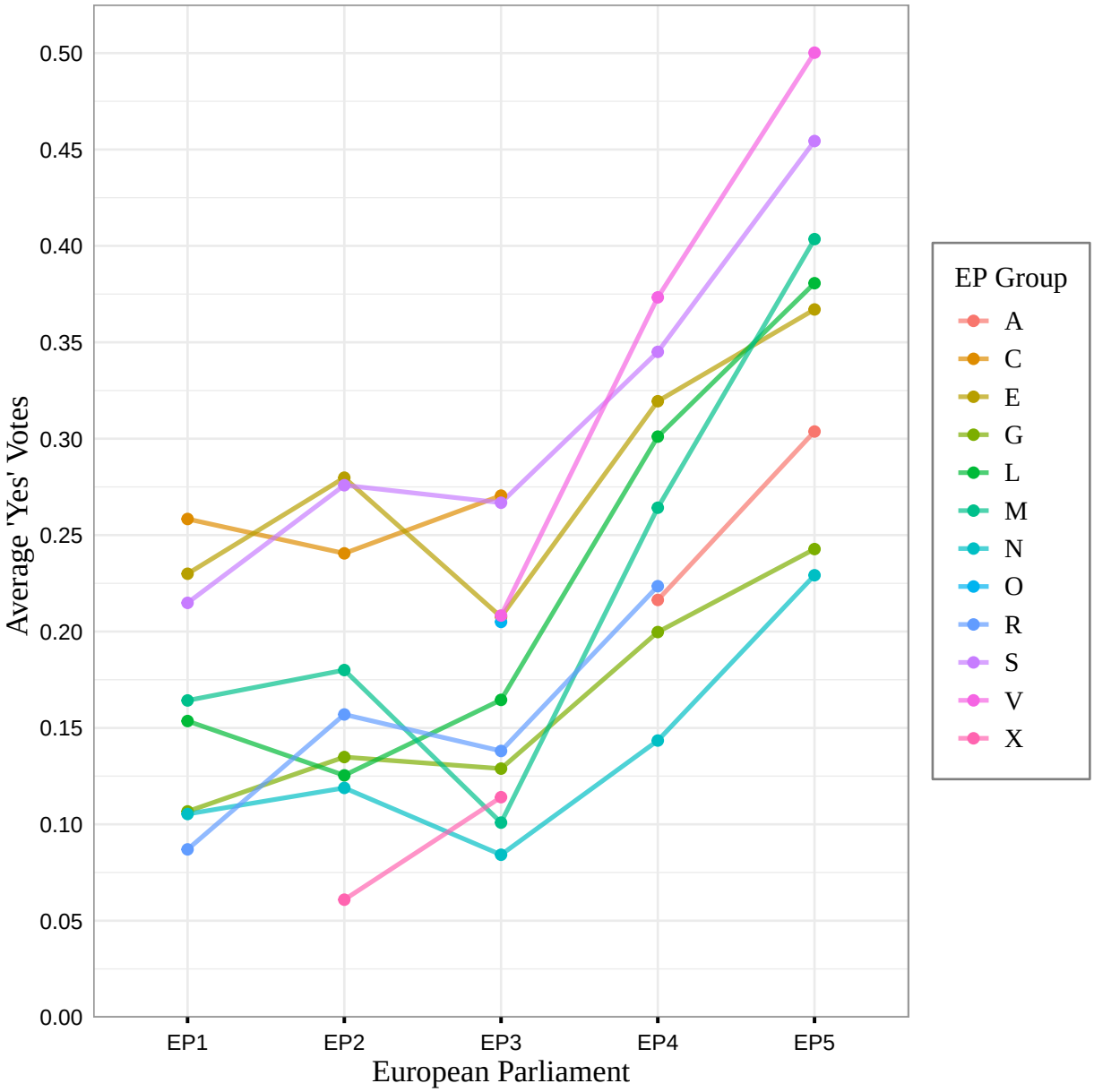
Clearly there is a trend of the over the first five parliments of increasing the average yes rate. By viewing the points the size of groups can be inferred as well as the spread within the family of the national parties.

5. For each EP group, calculate the average *Yes* share per year and plot a line graph. Similar to before creating a yes rate by EP group, to create an average for each group in each of the five EP's. Plotting as a line graph enables comparison of the groups change over time as well as between the groups.

```
##———————— Plot 5 ————————## 340
#Data separate
Yes_rate_EPG <- RCV_com %>% #now grouping by group per EP to see time
    prog
  group_by(EP, EPG) %>% #splitting data
  summarise( #same again
    total_votes = n(),
    yes_votes = sum(`Outcome_Category` == "Yes"),
    yes_rate = yes_votes / total_votes
  )

pdf("PS01_Fig5.pdf")
ggplot(Yes_rate_EPG, aes(x=EP, y = yes_rate, colour = EPG, group = EPG))+
  geom_line(size = 1, alpha = 0.7) + #line mapping each group through EP'
    s
  geom_point(size = 2) + #adding points on mean yes rate
  scale_y_continuous(limits = c(0, 0.525), breaks = seq(0, 0.65, 0.05),
    expand = c(0,0)) +
  labs(x = "European Parliament", y = "Average 'Yes' Votes", colour = "EP
     Group") +
  theme_minimal()+
  theme(
    panel.border = element_rect(colour = "grey60", fill = NA, size = 0.5)
    ,
    axis.title = element_text(family = "serif", size = 14), #title font
    axis.text = element_text(colour = "black", size = 10), #axis text
    change
    axis.ticks.x = element_line(colour = "black", size = 0.75),
    strip.text = element_text(family = 'serif', size = 11, colour = '
    black'),
    legend.title = element_text(family = 'serif', size = 13, colour = '
    black'), #legend
    legend.text =  element_text(family = 'serif', size = 12, colour = '
    black'),# font change
    legend.position = "right",
    legend.margin = margin(10,10,10,10),
    legend.background = element_rect(
      fill = "white",
      colour = "grey50", #make box aorund legend to align it
      size = 0.5,
      linetype = "solid")
  )
```

```
34 dev.off()
```

Figure 5: EP Group average 'Yes' share over the first five European Parliaments



The clear rise of Average 'Yes' votes in the later EP's can be seen here.