Lab Rotation 2 - Portfolio

Mijin Gwak

Preliminaries

```
library(wordbankr) # Wordbank data
library(tidyverse) # general use
library(readxl) # importing xlsx
library(knitr) # formatting, tables, etc
library(kableExtra)
library(psych) # dimensionality assessment
library(Gifi)
library(corrplot) # correlation plots
```

Exploring Data with Descriptive Statistics

Wordbank Data

1. Korean

```
Inst_KO <- get_instrument_data(language = "Korean", form = "WS")
Admin_KO <- get_administration_data(language = "Korean", form = "WS")
Item_KO <- get_item_data(language = "Korean", form = "WS")
```

1.1. Complexity Scale

There are two sources for the Korean CDI dataset: *Pae* and *Yim*. Only *Yim*'s dataset in Wordbank have complexity data in which *NA* denotes that the child isn't producing the relevant construction, *1* denotes they are producing the simpler construction, and *2* denotes they are producing the more complicated counterpart.

Here are the translations (joined by *space+line+space*) to be appended to the item data:

```
translation_KO <- strsplit("big thing flower / big flower \n sister's thing snack / s
ister's snack \n dad's thing watch / dad's watch \n eat not / not eat \n not you ca
n't / you can \n ate it all not / didn't eat it all \n read the book not / not read t
he book \n pour the cup / pour it into the cup \n make it in home / make it at home
 \n mom dad likes / mom likes dad \n sick go to the hospital / go to the hospital bec
ause  sick \n eat should not / should not eat it \n let's puts (it) in here * wrong i
nflection / let's put (it) in here \n (You) have to pours it though * wrong inflectio
n / (You) have to pour it though \n (You) eats a lot * wrong inflection / (You) eat a
lot \n Baby is cry * wrong inflection / Baby is crying \n It's the thing dad drives t
he car / It's the car dad drives \n The thing does it there is a door / There is a do
or that does this \n It is room, right? / It is in the room, right? \n The wind blew
 away like swish / The air was leaked, so it blew away like swish \n Mom flower(+ obj
ect marker) because broken / Because the window was broken while mom was touching the
flowers. \n One video like that / (I) have seen a video like that before \n I helicop
ter sound (+ subject marker) heard high up in the sky / I heard a helicopter making s
ound in the sky \n Because sick because baby I love / (It) hurts, but I love (the sub
ject) because (the subject) is baby \n Say sorry (no case marker) / said sorry (+ cas
e marker indicating quotation) \n bleeding and crying / crying because bleeding \n Pu
t it on, please, Put it on (me), please \n Please, ride it, Please, let (me) ride it
 \n (It) was hurt / (It) hurt me \n (It) won't opens  * wrong inflection, what do (I)
do? / (It) won't open, what do (I) do? \n was * wrong inflection sick   / was sick \n
This thing opened / This (you) have to open", ' \n ') %>%
  as.data.frame() %>%
  setNames('translation')
```

Append the translation to the item data:

```
KO_in_EN <- Item_KO %>%
  filter(type == 'complexity') %>%
  add_column(translation_KO)
```

Combine the data grouped by complexity scores, choose *Yim*'s dataset, and convert the categorical values to corresponding numerical values:

```
Complex_KO <- Admin_KO %>%
  full_join(., Inst_KO, by = "data_id") %>%
  full_join(., KO_in_EN, by = "num_item_id") %>%
  filter(source_name == 'Yim' & type == 'complexity') %>% # filter by the source name
and type
  mutate(
    out = case_when(
      is.na(value) ~ 0, # replace NAs with 0
      value == 1 ~ 1,
      value == 2 ~ 2
    )
  )
```

```
## # A tibble: 6 × 29
##   data_id   age comprehension production language.x form.x birth_order ethnicity
##     <dbl> <int>         <int>      <int> <chr>      <chr>  <fct>       <fct>
## 1  166783    20           176         64 Korean     WS     First       <NA>
## 2  166783    20           176         64 Korean     WS     First       <NA>
## 3  166783    20           176         64 Korean     WS     First       <NA>
## 4  166783    20           176         64 Korean     WS     First       <NA>
## 5  166783    20           176         64 Korean     WS     First       <NA>
## 6  166783    20           176         64 Korean     WS     First       <NA>
## # … with 21 more variables: sex <fct>, zygosity <chr>, norming <lgl>,
## #   mom_ed <fct>, longitudinal <lgl>, source_name <chr>, license <chr>,
## #   value <chr>, num_item_id <dbl>, item_id <chr>, definition <chr>,
## #   language.y <chr>, form.y <chr>, type <chr>, category <chr>,
## #   lexical_category <chr>, lexical_class <chr>, uni_lemma <chr>,
## #   complexity_category <chr>, translation <chr>, out <dbl>
```

1.1.1. Gender and Age of Children Who Have Complexity Scores

Mean and standard deviations of age:

```
Complex_KO_unique <- Complex_KO[!duplicated(Complex_KO$data_id),] # extract unique ids for counting

Complex_KO_unique %>%
  summarise(
    Mean = mean(age),
    SD = sd(age)
  )
```

```
## # A tibble: 1 × 2
##    Mean    SD
##   <dbl> <dbl>
## 1  27.3  4.89
```

Numbers of females and males:

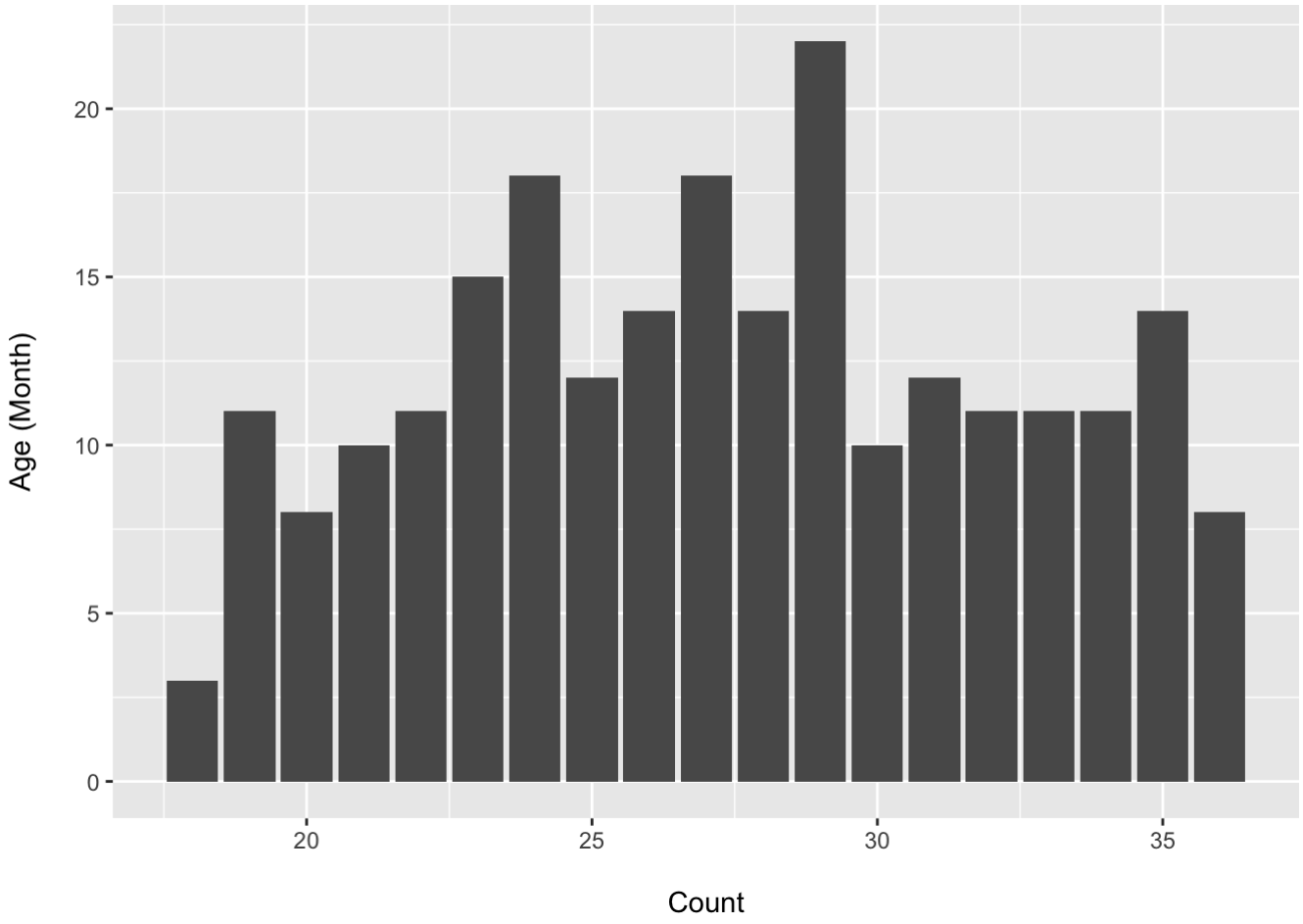
```
Complex_KO_gender <- as.data.frame(table(Complex_KO_unique$sex))
names(Complex_KO_gender)[1] <- 'Gender'
```

```
##   Gender Freq
## 1 Female  102
## 2   Male  129
## 3  Other    0
```

Plots

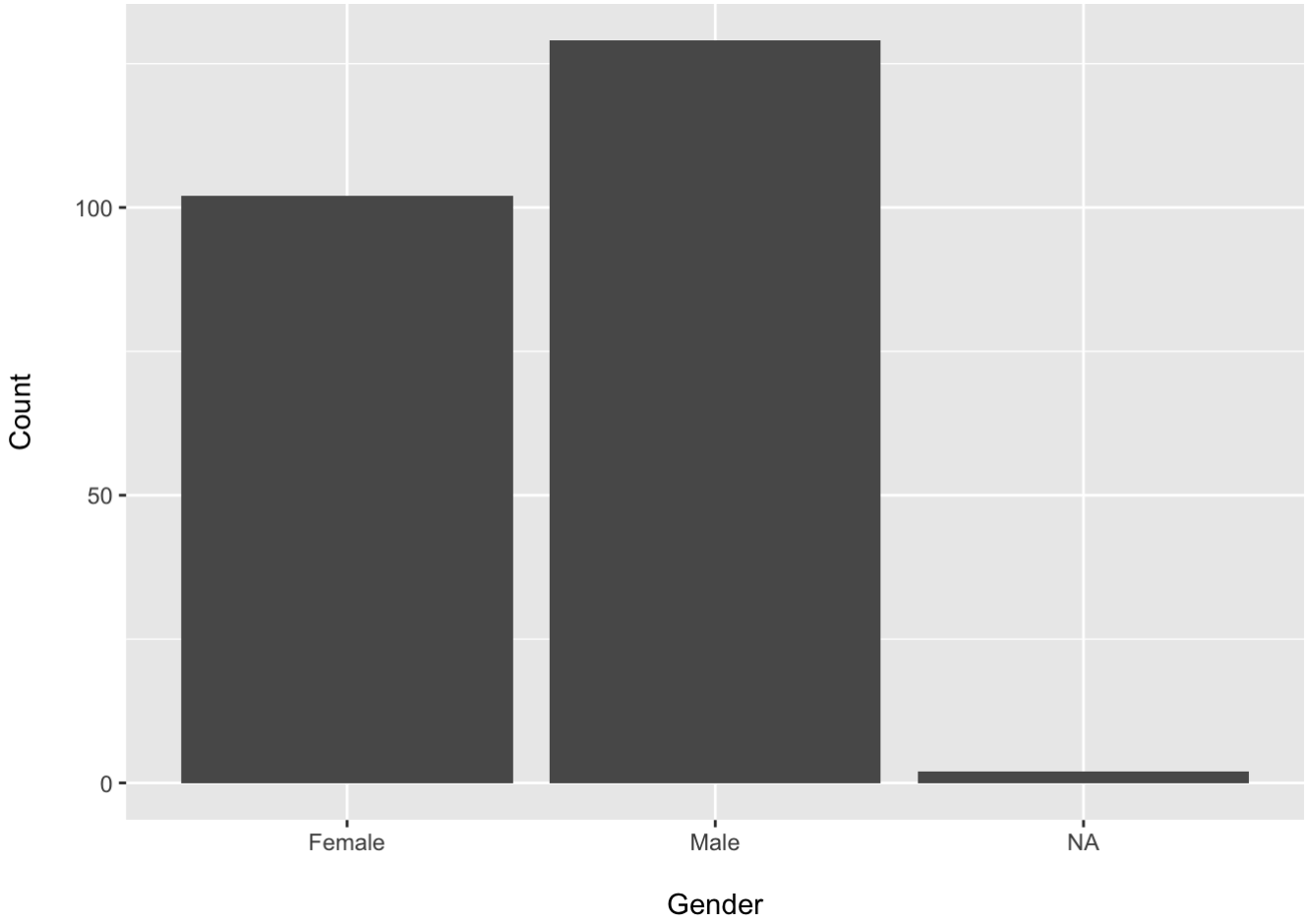
Age distribution:

```
ggplot(Complex_KO_unique, aes(x = age)) +
  geom_bar() +
  labs(x = paste('', 'Count', sep = '\n'), y = paste('Age (Month)', '', sep = '\n'))
```



Gender distribution:

```
ggplot(Complex_KO_unique, aes(x = sex)) +
  geom_bar() +
  labs(x = paste('', 'Gender', sep = '\n'), y = paste('Count', '', sep = '\n'))
```



1.1.2. Means and Standarad Deviations of Complexity Scores per Item

```
Complex_KO_mean <- Complex_KO %>%
  group_by(translation) %>%
  summarise(
    num_item_id = num_item_id,
    mean = mean(out),
    sd = sd(out),
  ) %>%
  distinct(., num_item_id, .keep_all = TRUE)
```

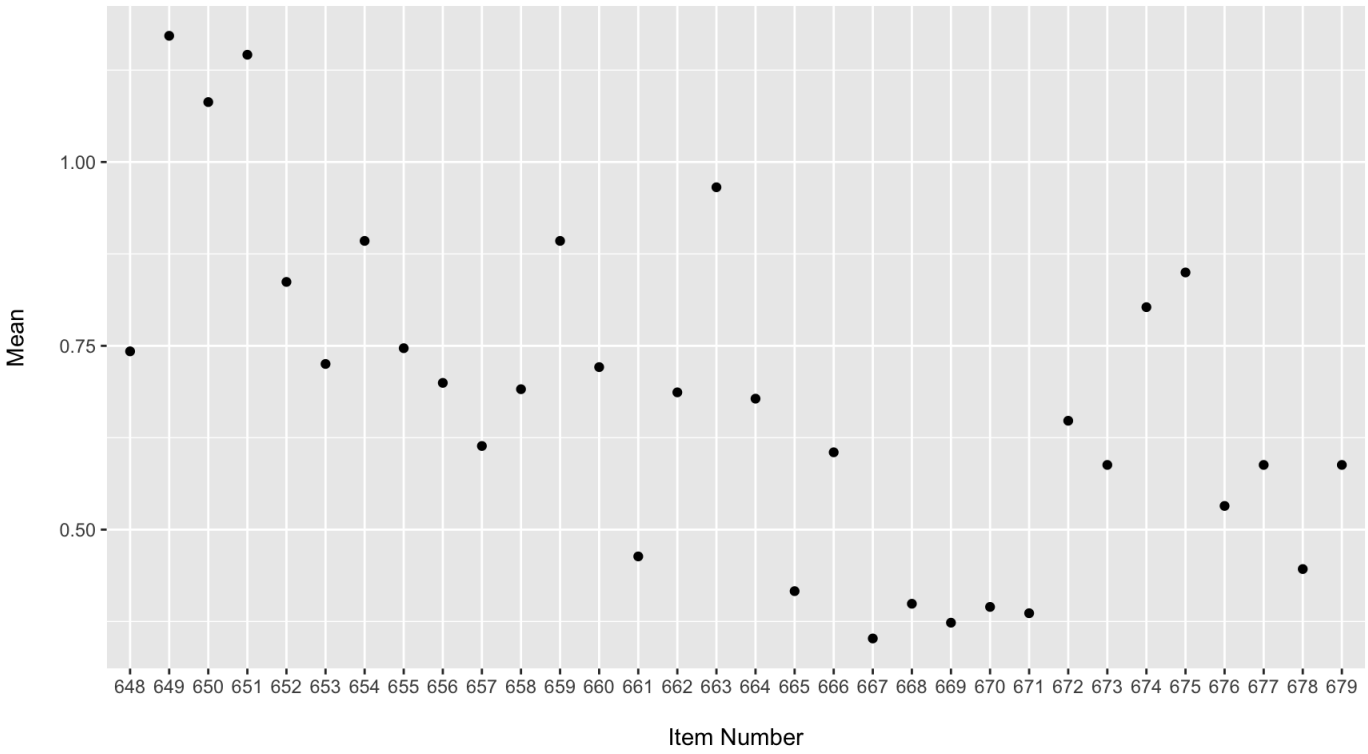
Means and SDs for Each Item (Arranged by Item)

translation	mean	sd
(It) was hurt / (It) hurt me	0.5321888	0.6948350
(It) won't opens * wrong inflection, what do (I) do? / (It) won't open, what do (I) do?	0.5879828	0.8867957
(You) eats a lot * wrong inflection / (You) eat a lot	0.6866953	0.9287755
(You) have to pours it though * wrong inflection / (You) have to pour it though	0.4635193	0.7129949
ate it all not / didn't eat it all	0.7253219	0.9386817
Baby is cry * wrong inflection / Baby is crying	0.9656652	0.9950856
Because sick because baby I love / (It) hurts, but I love (the subject) because (the subject) is baby	0.3862661	0.7691129
big thing flower / big flower	0.7424893	0.9110182
bleeding and crying / crying because bleeding	0.5879828	0.9107745
dad's thing watch / dad's watch	1.0815451	0.9084153
eat not / not eat	1.1459227	0.9446734

translation	mean	sd
eat should not / should not eat it	0.8927039	0.9700640
I helicopter sound (+ subject marker) heard high up in the sky / I heard a helicopter making sound in the sky	0.3948498	0.7703387
It is room, right? / It is in the room, right?	0.6051502	0.8751202
It’s the thing dad drives the car / It’s the car dad drives	0.6781116	0.9258230
let’s puts (it) in here * wrong inflection / let’s put (it) in here	0.7210300	0.9070294
make it in home / make it at home	0.6995708	0.9260827
mom dad likes / mom likes dad	0.6137339	0.8935498
Mom flower(+ object marker) because broken / Because the window was broken while mom was touching the flowers.	0.3991416	0.7709148
not you can’t / you can	0.8369099	0.8142126
One video like that / (I) have seen a video like that before	0.3733906	0.7500432
Please, ride it, Please, let (me) ride it	0.8497854	0.9687854
pour the cup / pour it into the cup	0.7467811	0.9650355
Put it on, please, Put it on (me), please	0.8025751	0.9352264
read the book not / not read the book	0.8927039	0.9876775
Say sorry (no case marker) / said sorry (+ case marker indicating quotation)	0.6480687	0.9029206
sick go to the hospital / go to the hospital because sick	0.6909871	0.8947911
sister’s thing snack / sister’s snack	1.1716738	0.9123980
The thing does it there is a door / There is a door that does this	0.4163090	0.7729757
The wind blew away like swish / The air was leaked, so it blew away like swish	0.3519313	0.6795715
This thing opened / This (you) have to open	0.5879828	0.8770207
was * wrong inflection sick / was sick	0.4463519	0.8083066

Plot

```
Complex_KO_mean$num_item_id <- as.factor(Complex_KO_mean$num_item_id) # convert the i
tem id number as a factor
ggplot(data = Complex_KO_mean, aes(x = num_item_id, y = mean)) +
  geom_point() +
  labs(x = paste('', 'Item Number', sep = '\n'), y = paste('Mean', '', sep = '\n'))
```



1.2. Combine Scale

There is only one item for combine scale: *Is the child combining words?* The scores are coded with *not yet*, *sometimes*, and *often*, which will be re-coded as *0*, *1*, and *2*, respectively.

Combine data sets and re-code:

```
Combine_KO <- Admin_KO %>%
  full_join(.,Inst_KO, by = "data_id") %>%
  full_join(., Item_KO, by = "num_item_id") %>%
  filter(type == 'combine') %>%  # filter by type
  drop_na(value) %>% # remove missing values
  mutate( # re-code
    out = case_when(
      value == 'not yet' ~ 0,
      value == 'sometimes' ~ 1,
      value == 'often' ~ 2))

Combine_KO$translation <- 'Is the child combining words?' # create a translation colu
mn and assign the translation
```

```
## # A tibble: 6 × 29
##   data_id   age comprehension production language.x form.x birth_order ethnicity
##     <dbl> <int>         <int>      <int> <chr>      <chr>  <fct>       <fct>
## 1  166783    20           176         64 Korean     WS     First       <NA>
## 2  166784    20           446        196 Korean     WS     First       <NA>
## 3  166785    34           557        473 Korean     WS     First       <NA>
## 4  166786    24           192        192 Korean     WS     First       <NA>
## 5  166787    29           597        557 Korean     WS     First       <NA>
## 6  166788    31           564        496 Korean     WS     Third       <NA>
## # … with 21 more variables: sex <fct>, zygosity <chr>, norming <lgl>,
## #   mom_ed <fct>, longitudinal <lgl>, source_name <chr>, license <chr>,
## #   value <chr>, num_item_id <dbl>, item_id <chr>, definition <chr>,
## #   language.y <chr>, form.y <chr>, type <chr>, category <chr>,
## #   lexical_category <chr>, lexical_class <chr>, uni_lemma <chr>,
## #   complexity_category <chr>, out <dbl>, translation <chr>
```

1.2.1. Gender and Age of Children Who Have Combine Scores

Mean and standard deviations of age:

```
Combine_KO_unique <- Combine_KO[!duplicated(Combine_KO$data_id),]

Combine_KO_unique %>%
  summarise(
    Mean = mean(age, na.rm = TRUE), # exclude missing values
    SD = sd(age, na.rm = TRUE)
  )
```

```
## # A tibble: 1 × 2
##    Mean    SD
##   <dbl> <dbl>
## 1  27.3  4.90
```

Numbers of females and males:

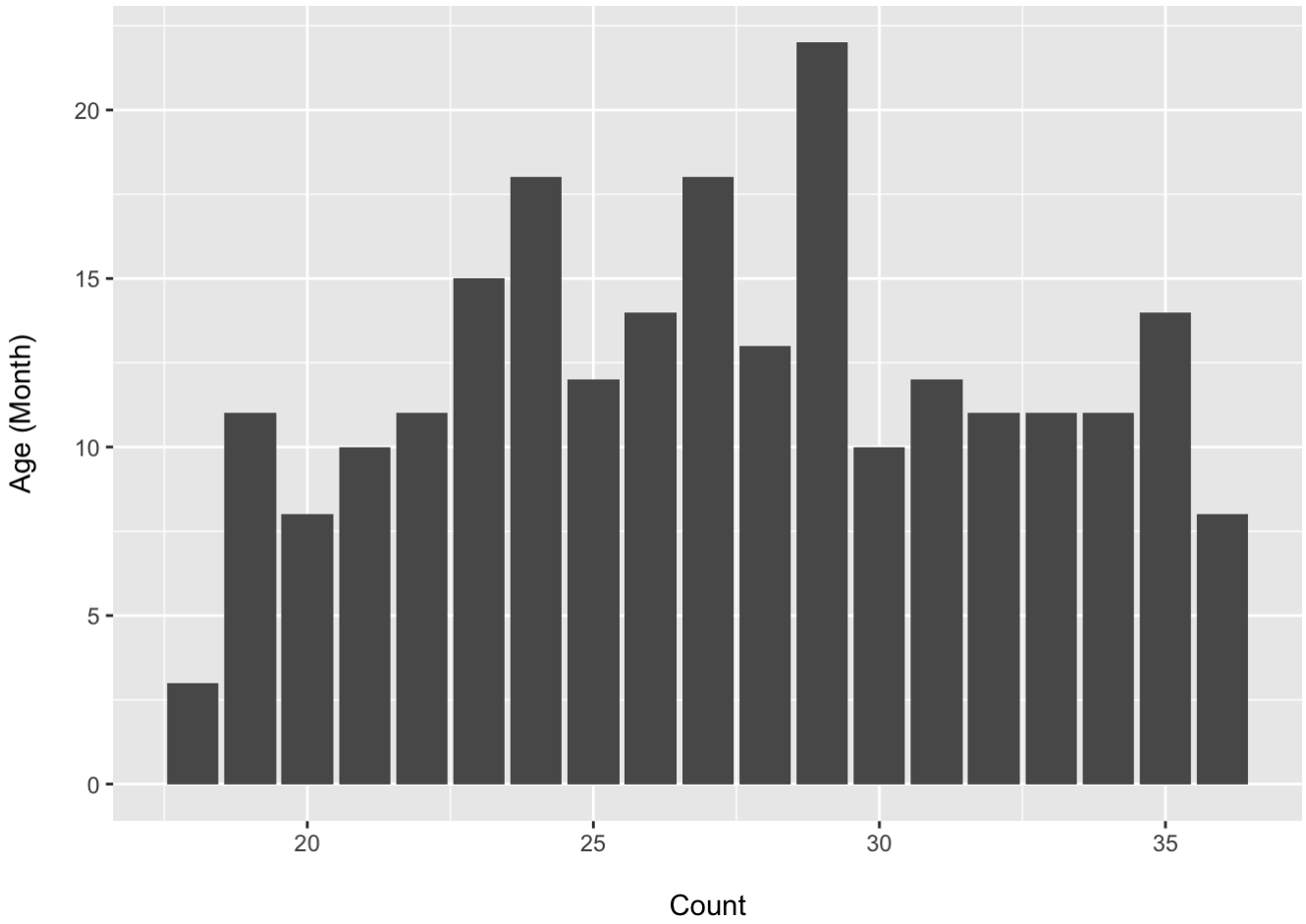
```
Combine_KO_gender <- as.data.frame(table(Combine_KO_unique$sex))
names(Combine_KO_gender)[1] <- 'Gender'
```

```
##   Gender Freq
## 1 Female  102
## 2   Male  128
## 3  Other    0
```

Plots

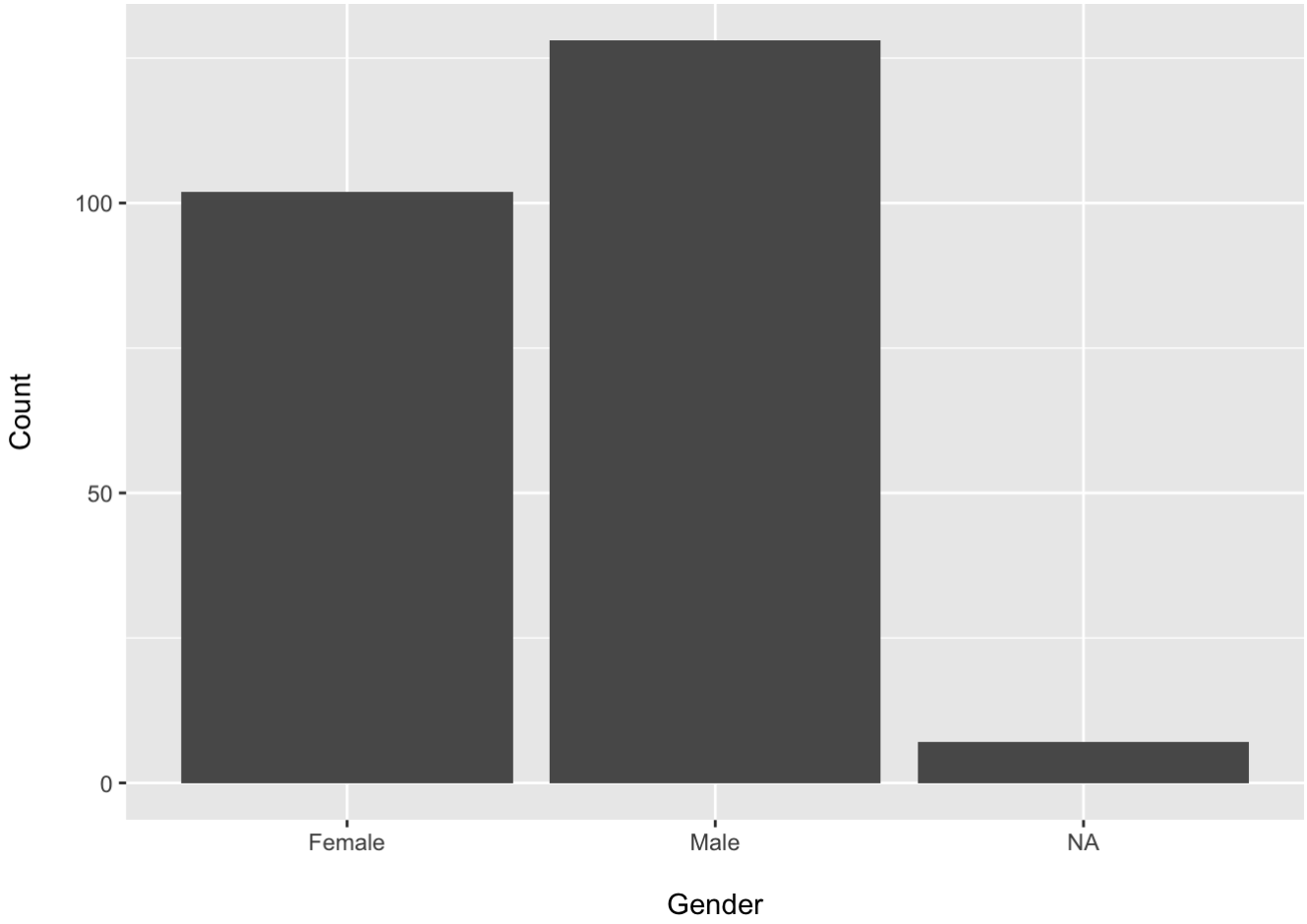
Age distribution:

```
ggplot(Combine_KO_unique, aes(x = age)) +
  geom_bar() +
  labs(x = paste('', 'Count', sep = '\n'), y = paste('Age (Month)', '', sep = '\n'))
```



Gender distribution:

```
ggplot(Combine_KO_unique, aes(x = sex)) +
  geom_bar() +
  labs(x = paste('', 'Gender', sep = '\n'), y = paste('Count', '', sep = '\n'))
```



1.2.2. Mean and Standarad Deviation of the Item

```
Combine_KO %>%
  group_by(translation) %>%
  summarise(
    mean = mean(out),
    sd = sd(out),
  ) %>%
  kable(caption="Means and SDs for Each Item (Arranged by Item)")
```

Means and SDs for Each Item (Arranged by Item)

translation	mean	sd
Is the child combining words?	1.396624	0.7884691

1.3. Sentence Structure Scale

There are five items for sentence structure scores:

item_643 - use of propositional particles
item_644 - inflection
item_645 - connective ending
item_646 - adnominal inflection
item_647 - passive/active affix

The scores for sentence structure are coded with *not yet*, *sometimes*, and *often*, which will be re-coded as *0*, *1*, and *2*, respectively.

Combine data sets and re-code:

```
Structure_KO <- Admin_KO %>%
  full_join(.,Inst_KO, by = "data_id") %>%
  full_join(., Item_KO, by = "num_item_id") %>%
  filter(type == 'sentence_structure') %>%  # filter by type
  drop_na(value) %>% # remove missing values
  mutate( # re-code
    out = case_when(
      value == 'not yet' ~ 0,
      value == 'sometimes' ~ 1,
      value == 'often' ~ 2),
    translation = case_when( # translation of the description of each item
      item_id == 'item_643' ~ 'use of propositional particles',
      item_id == 'item_644' ~ 'inflection',
      item_id == 'item_645' ~ 'connective ending',
      item_id == 'item_646' ~ 'adnominal inflection',
      item_id == 'item_647' ~ 'passive/active affix'))
```

```
## # A tibble: 6 × 29
##   data_id   age comprehension production language.x form.x birth_order ethnicity
##     <dbl> <int>         <int>      <int> <chr>      <chr>  <fct>       <fct>
## 1  166783    20           176         64 Korean     WS     First       <NA>
## 2  166783    20           176         64 Korean     WS     First       <NA>
## 3  166783    20           176         64 Korean     WS     First       <NA>
## 4  166783    20           176         64 Korean     WS     First       <NA>
## 5  166783    20           176         64 Korean     WS     First       <NA>
## 6  166784    20           446        196 Korean     WS     First       <NA>
## # … with 21 more variables: sex <fct>, zygosity <chr>, norming <lgl>,
## #   mom_ed <fct>, longitudinal <lgl>, source_name <chr>, license <chr>,
## #   value <chr>, num_item_id <dbl>, item_id <chr>, definition <chr>,
## #   language.y <chr>, form.y <chr>, type <chr>, category <chr>,
## #   lexical_category <chr>, lexical_class <chr>, uni_lemma <chr>,
## #   complexity_category <chr>, out <dbl>, translation <chr>
```

1.3.1. Gender and Age of Children Who Have Sentence Structure Scores

Mean and standard deviations of age:

```
Structure_KO_unique <- Structure_KO[!duplicated(Structure_KO$data_id),]

Structure_KO_unique %>%
  summarise(
    Mean = mean(age, na.rm = TRUE),
    SD = sd(age, na.rm = TRUE)
  )
```

```
## # A tibble: 1 × 2
##    Mean    SD
##   <dbl> <dbl>
## 1  27.4  4.94
```

Numbers of females and males:

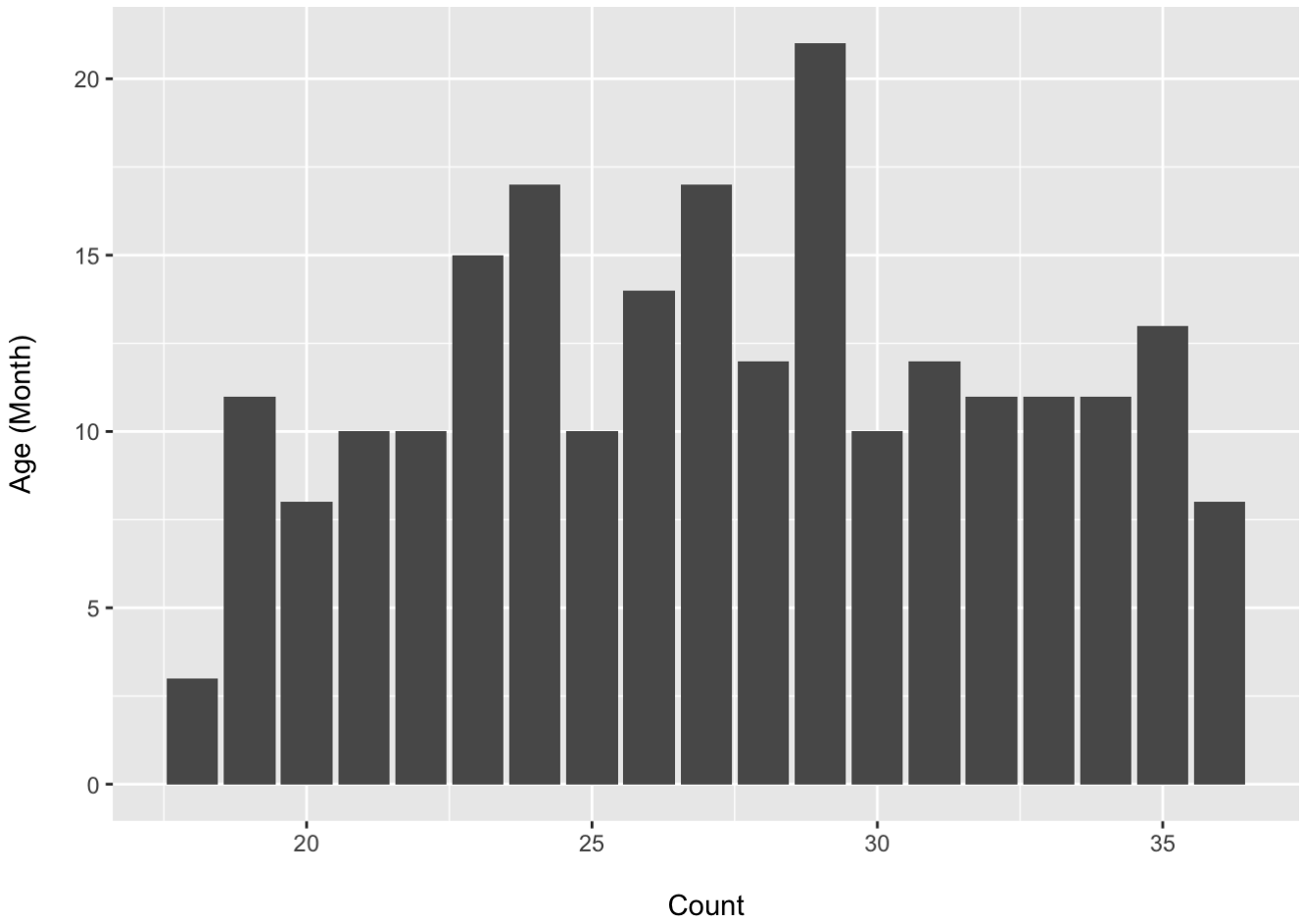
```
Structure_KO_gender <- as.data.frame(table(Combine_KO_unique$sex))
names(Structure_KO_gender)[1] <- 'Gender'
```

```
##   Gender Freq
## 1 Female  102
## 2   Male  128
## 3  Other    0
```

Plots

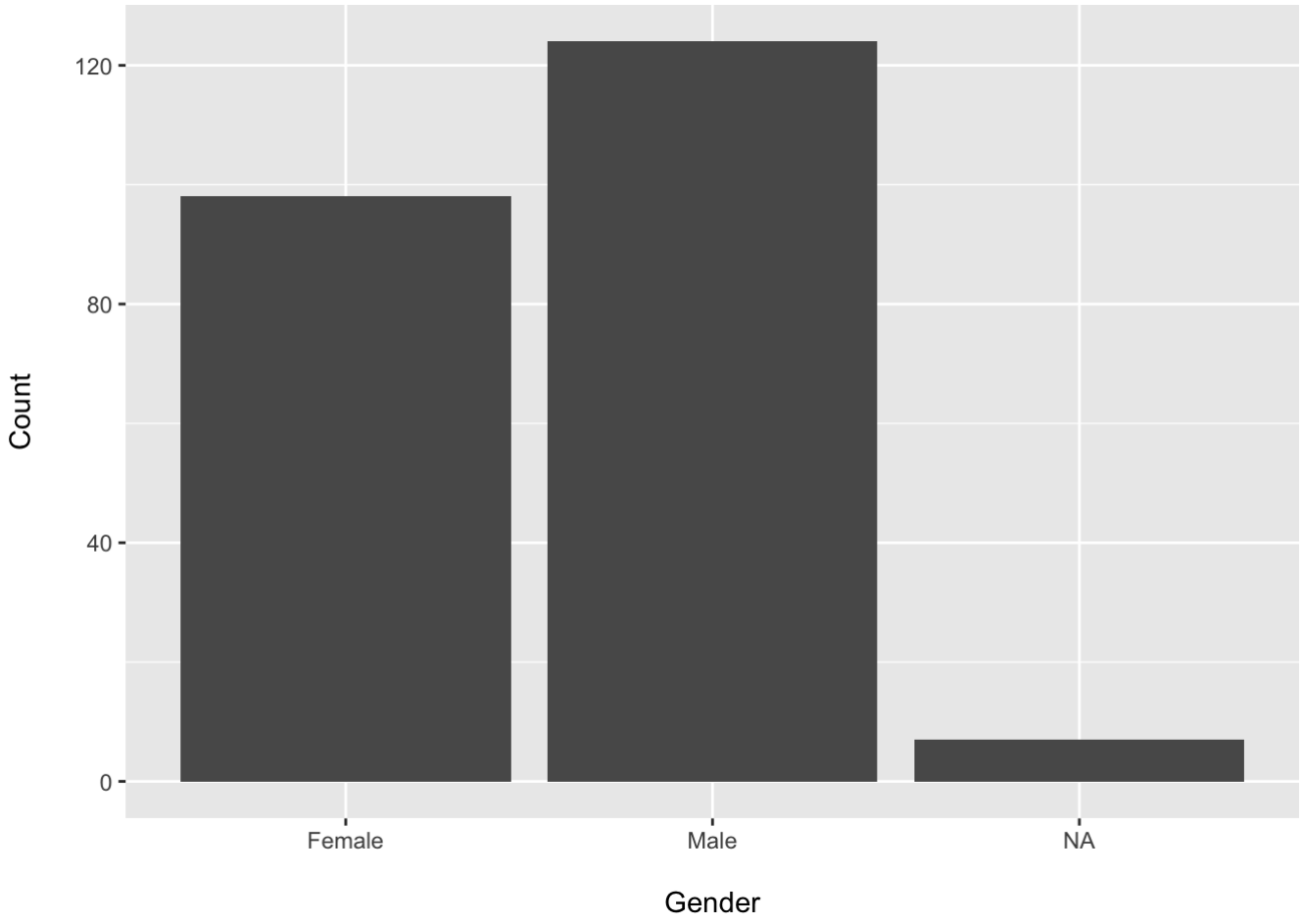
Age distribution:

```
ggplot(Structure_KO_unique, aes(x = age)) +
  geom_bar() +
  labs(x = paste('', 'Count', sep = '\n'), y = paste('Age (Month)', '', sep = '\n'))
```



Gender distribution:

```
ggplot(Structure_KO_unique, aes(x = sex)) +
  geom_bar() +
  labs(x = paste('', 'Gender', sep = '\n'), y = paste('Count', '', sep = '\n'))
```



1.3.2. Means and Standarad Deviations of Sentence Structure Scores per Item

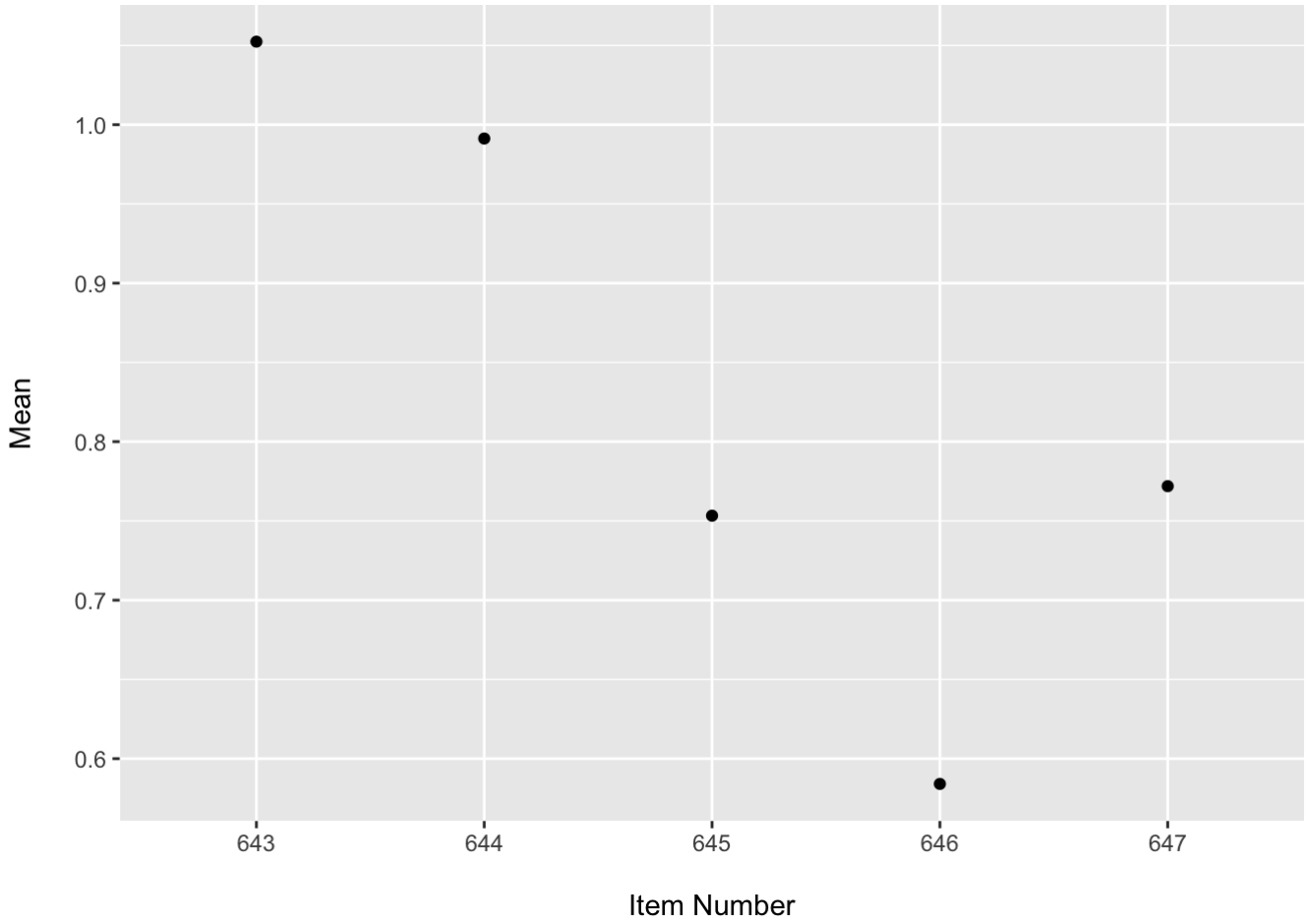
```
Structure_KO_mean <- Structure_KO %>%
  group_by(translation) %>%
  summarise(
    num_item_id = num_item_id,
    mean = mean(out),
    sd = sd(out),
  ) %>%
  distinct(., num_item_id, .keep_all = TRUE)
```

Means and SDs for Each Item (Arranged by Item)

translation	mean	sd
adnominal inflection	0.5840708	0.8025035
connective ending	0.7533040	0.9076495
inflection	0.9912664	0.9031682
passive/active affix	0.7719298	0.8606430
use of propositional particles	1.0524017	0.9304102

Plot

```
Structure_KO_mean$num_item_id <- as.factor(Structure_KO_mean$num_item_id) # convert t
he item id number as a factor
ggplot(data = Structure_KO_mean, aes(x = num_item_id, y = mean)) +
  geom_point() +
  labs(x = paste('', 'Item Number', sep = '\n'), y = paste('Mean', '', sep = '\n'))
```



2. Spanish

Data Screening

```
Inst_ES <- get_instrument_data(language = "Spanish (Mexican)", form = "WS")
Admin_ES <- get_administration_data(language = "Spanish (Mexican)", form = "WS")
Item_ES <- get_item_data(language = "Spanish (Mexican)", form = "WS")
```

Combine data sets and create binary variable:

```
Complex_ES <- Admin_ES %>%
  full_join(.,Inst_ES, by="data_id") %>%
  full_join(., Item_ES, by="num_item_id") %>%
  filter(longitudinal==FALSE) %>%
  filter(type == "complexity") %>%
  mutate(
    out = ifelse(value=="complex", yes=1, no=0)
  )
```

```
head(Complex_ES)
```

```
## # A tibble: 6 × 28
##   data_id   age comprehension production language.x form.x birth_order ethnicity
##     <dbl> <int>         <int>      <int> <chr>      <chr>  <fct>       <fct>
## 1  170402    20            52         52 Spanish (… WS     Second      <NA>
## 2  170402    20            52         52 Spanish (… WS     Second      <NA>
## 3  170402    20            52         52 Spanish (… WS     Second      <NA>
## 4  170402    20            52         52 Spanish (… WS     Second      <NA>
## 5  170402    20            52         52 Spanish (… WS     Second      <NA>
## 6  170402    20            52         52 Spanish (… WS     Second      <NA>
## # … with 20 more variables: sex <fct>, zygosity <chr>, norming <lgl>,
## #   mom_ed <fct>, longitudinal <lgl>, source_name <chr>, license <chr>,
## #   value <chr>, num_item_id <dbl>, item_id <chr>, definition <chr>,
## #   language.y <chr>, form.y <chr>, type <chr>, category <chr>,
## #   lexical_category <chr>, lexical_class <chr>, uni_lemma <chr>,
## #   complexity_category <chr>, out <dbl>
```

Missing scores:

```
Complex_ES %>%
  filter(is.na(out)) %>%
  group_by(definition) %>%
  count()
```

```
## # A tibble: 0 × 2
## # Groups:   definition [0]
## # … with 2 variables: definition <chr>, n <int>
```

-> There are no missing scores.

Gender and Age of Children Who Have Complexity Scores

Mean and standard deviations of age:

```
Complex_ES_unique <- Complex_ES[!duplicated(Complex_ES$data_id),] # extract unique ids for counting

Complex_ES_unique %>%
  summarise(
    Mean = mean(age),
    SD = sd(age)
  )
```

```
## # A tibble: 1 × 2
##    Mean    SD
##   <dbl> <dbl>
## 1  23.1  4.20
```

Numbers of females and males:

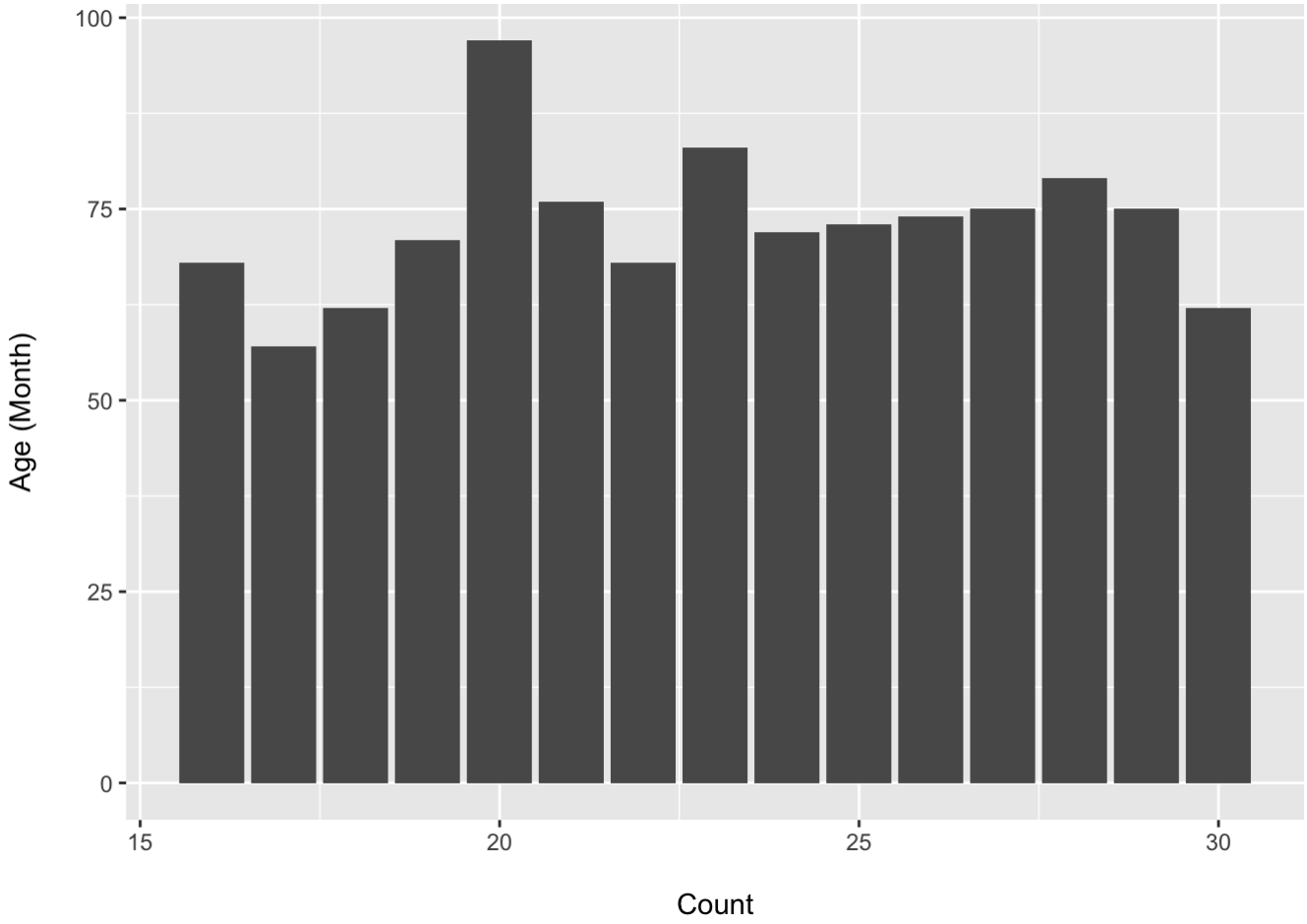
```
Complex_ES_gender <- as.data.frame(table(Complex_ES_unique$sex))
names(Complex_ES_gender)[1] <- 'Gender'
```

```
##   Gender Freq
## 1 Female  541
## 2   Male  551
## 3  Other    0
```

Plots

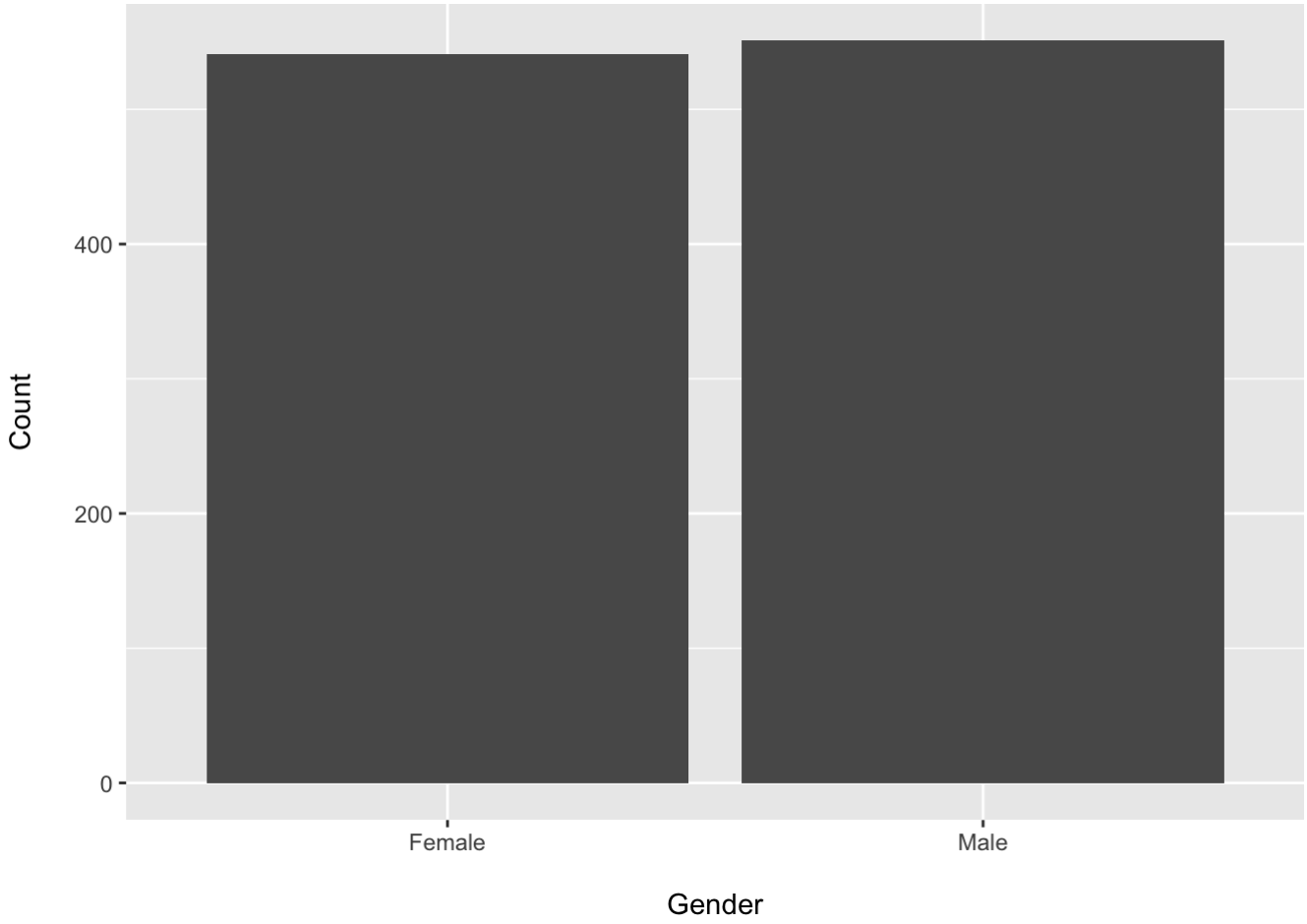
Age distribution:

```
ggplot(Complex_ES_unique, aes(x = age)) +
  geom_bar() +
  labs(x = paste('', 'Count', sep = '\n'), y = paste('Age (Month)', '', sep = '\n'))
```



Gender distribution:

```
ggplot(Complex_ES_unique, aes(x = sex)) +
  geom_bar() +
  labs(x = paste('', 'Gender', sep = '\n'), y = paste('Count', '', sep = '\n'))
```



Means and Standarad Deviations of Complexity Scores per Item

```
Complex_ES_mean <- Complex_ES %>%
  group_by(definition) %>%
  summarise(
    num_item_id = num_item_id,
    mean = mean(out),
    sd = sd(out),
  ) %>%
  distinct(., num_item_id, .keep_all = TRUE)
```

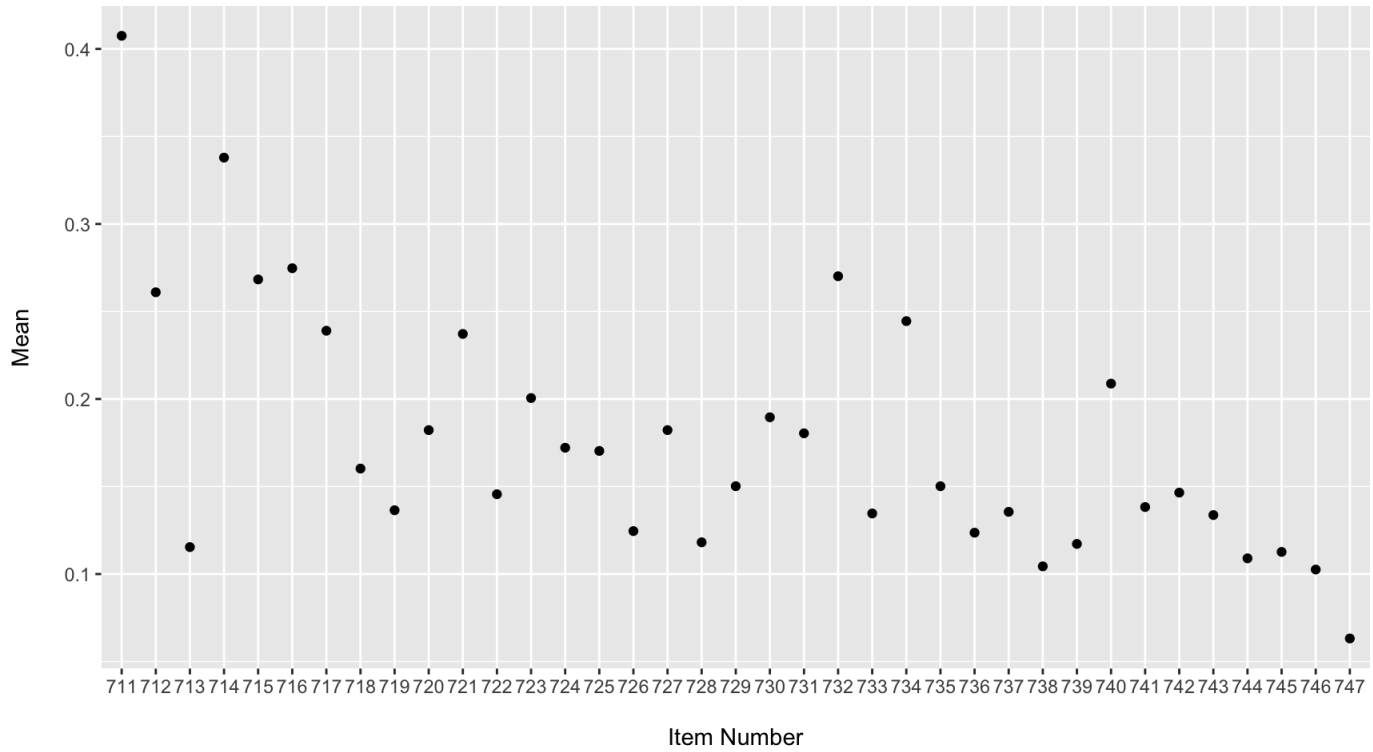
Means and SDs for Each Item (Arranged by Item)

definition	mean	sd
a silla / en la silla	0.2747253	0.4465803
abre dame galleta / abre la caja y dame una galleta	0.1089744	0.3117498
acabó agua / se me acabó el agua	0.1501832	0.3574145
agua vamos / vamos al agua	0.2683150	0.4432855
bravo Tello circo / dije bravo en el circo	0.1346154	0.3414687
calle allá está / allá está la calle	0.1822344	0.3862145
Chalo osito coche / Chalo dejó el osito en el coche	0.1236264	0.3293056
duele panza / me duele la panza	0.1703297	0.3760944
fue casa / se fue a su casa	0.1895604	0.3921327
guaguá grande / tengo un perro grande	0.1245421	0.3303503
lápiz dibujar / dibujo con el lápiz	0.1355311	0.3424468
leche caliente / la leche está caliente	0.1721612	0.3776934

definition	mean	sd
mamá nene compra / mamá y nene fueron a comprar	0.1336996	0.3404852
Marta papá / quiero ir con papá	0.2701465	0.4442388
más leche / dame más leche	0.1822344	0.3862145
mío lápiz / éste es mi lápiz	0.1364469	0.3434198
nene quiere / quiero paleta	0.4075092	0.4915961
nene rompió bici Danny / el niño rompió la bici de Danny	0.1172161	0.3218250
niño llora cayó / el niño llora porque se cayó	0.1465201	0.3537889
no aquí / ése no está aquí	0.1456044	0.3528708
no toca; quemas / no lo toques porque te quemas	0.1126374	0.3162939
Paloma llorando / Paloma está llorando	0.1602564	0.3670118
papá calle / papá se fue a trabajar	0.2445055	0.4299905
papo mami / el zapato es de mami	0.2371795	0.4255481
Pepe uvas / quiero uvas	0.3379121	0.4732151
pollo no / no quiero pollo	0.2390110	0.4266749
pone no / no lo pongas	0.2087912	0.4066311
pongo agua flores / pongo agua para que crezcan las flores	0.0631868	0.2434101
puse a mano / lo puse en mi mano	0.1181319	0.3229121
quiero libro papá / quiero el libra que compró papá	0.1025641	0.3035277
rompió globo / se rompió el globo	0.2005495	0.4005950
silla subir / me quiero subir a la silla	0.1804029	0.3846989
Tito malo / soy malo	0.1153846	0.3196319
tuyo esto / este es tuyo	0.2609890	0.4393752
vamos comer papas carne / vamos a comer papas y carne	0.1382784	0.3453501
ya pinté / ya acabé de pintar	0.1043956	0.3059132
ya puse / ya se lo puse	0.1501832	0.3574145

Plot

```
Complex_ES_mean$num_item_id <- as.factor(Complex_ES_mean$num_item_id)
ggplot(data = Complex_ES_mean, aes(x = num_item_id, y = mean)) +
  geom_point() +
  labs(x = paste('', 'Item Number', sep = '\n'), y = paste('Mean', '', sep = '\n'))
```



3. Chiense

Data Screening

```
Inst_CH <- get_instrument_data(language = "Mandarin (Beijing)", form = "WS")
Admin_CH <- get_administration_data(language = "Mandarin (Beijing)", form = "WS")
Item_CH <- get_item_data(language = "Mandarin (Beijing)", form = "WS")
```

Combine data sets and create binary variable:

```
Complex_CH <- Admin_CH %>%
  full_join(.,Inst_CH, by="data_id") %>%
  full_join(., Item_CH, by="num_item_id") %>%
  filter(type == "complexity")

Complex_CH$value <- as.numeric(Complex_CH$value) # convert the value column to numeri
c
```

```
## # A tibble: 6 × 27
##   data_id   age comprehension production language.x form.x birth_order ethnicity
##     <dbl> <int>         <int>      <int> <chr>      <chr>  <fct>       <fct>
## 1   78965    16            71         71 Mandarin … WS     <NA>        <NA>
## 2   78965    16            71         71 Mandarin … WS     <NA>        <NA>
## 3   78965    16            71         71 Mandarin … WS     <NA>        <NA>
## 4   78965    16            71         71 Mandarin … WS     <NA>        <NA>
## 5   78965    16            71         71 Mandarin … WS     <NA>        <NA>
## 6   78965    16            71         71 Mandarin … WS     <NA>        <NA>
## # … with 19 more variables: sex <fct>, zygosity <chr>, norming <lgl>,
## #   mom_ed <fct>, longitudinal <lgl>, source_name <chr>, license <chr>,
## #   value <dbl>, num_item_id <dbl>, item_id <chr>, definition <chr>,
## #   language.y <chr>, form.y <chr>, type <chr>, category <chr>,
## #   lexical_category <chr>, lexical_class <chr>, uni_lemma <chr>,
## #   complexity_category <chr>
```

Missing scores:

```
Complex_CH %>%
  filter(is.na(value)) %>%
  group_by(definition) %>%
  count()
```

```
## # A tibble: 0 × 2
## # Groups:   definition [0]
## # … with 2 variables: definition <chr>, n <int>
```

-> There are no missing scores.

Since each scale has different choices of values, the choices will also be displayed:

```
choices_CH <- c(
"item_810    3; 2; 1; 0",
"item_811    3; 2; 1; 0",
"item_812    4; 3; 2; 1; 0",
"item_813    3; 2; 1; 0",
"item_814    4; 3; 2; 1; 0",
"item_815    2; 1; 0",
"item_816    2; 1; 0",
"item_817    4; 3; 2; 1; 0",
"item_818    5; 4; 3; 2; 1; 0",
"item_819    3; 2; 1; 0",
"item_820    3; 2; 1; 0",
"item_821    2; 1; 0",
"item_822    4; 3; 2; 1; 0",
"item_823    2; 1; 0",
"item_824    3; 2; 1; 0",
"item_825    3; 2; 1; 0",
"item_826    2; 1; 0",
"item_827    3; 2; 1; 0",
"item_828    2; 1; 0",
"item_829    3; 2; 1; 0",
"item_830    3; 2; 1; 0",
"item_831    4; 3; 2; 1; 0",
"item_832    2; 1; 0",

"item_833    2; 1; 0",
"item_834    2; 1; 0",
"item_835    5; 4; 3; 2; 1; 0",
"item_836    3; 2; 1; 0")

df_choices_CH <- data.frame(x = choices_CH)
df_choices_CH_splited <- as.data.frame(str_split_fixed(df_choices_CH$x, "\t", 2)) %>%
  mutate(max = substr(V2, 1, 2))
colnames(df_choices_CH_splited) <- c("item_id", "choices", "max")
```

```
##     item_id           choices max
## 1  item_810        3; 2; 1; 0   3
## 2  item_811        3; 2; 1; 0   3
## 3  item_812     4; 3; 2; 1; 0   4
## 4  item_813        3; 2; 1; 0   3
## 5  item_814     4; 3; 2; 1; 0   4
## 6  item_815           2; 1; 0   2
## 7  item_816           2; 1; 0   2
## 8  item_817     4; 3; 2; 1; 0   4
## 9  item_818  5; 4; 3; 2; 1; 0   5
## 10 item_819        3; 2; 1; 0   3
## 11 item_820        3; 2; 1; 0   3
## 12 item_821           2; 1; 0   2
## 13 item_822     4; 3; 2; 1; 0   4
## 14 item_823           2; 1; 0   2
## 15 item_824        3; 2; 1; 0   3
## 16 item_825        3; 2; 1; 0   3
## 17 item_826           2; 1; 0   2
## 18 item_827        3; 2; 1; 0   3
## 19 item_828           2; 1; 0   2
## 20 item_829        3; 2; 1; 0   3
## 21 item_830        3; 2; 1; 0   3
## 22 item_831     4; 3; 2; 1; 0   4
## 23 item_832           2; 1; 0   2
## 24 item_833           2; 1; 0   2
## 25 item_834           2; 1; 0   2
## 26 item_835  5; 4; 3; 2; 1; 0   5
## 27 item_836        3; 2; 1; 0   3
```

Gender and Age of Children Who Have Complexity Scores

Mean and standard deviations of age:

```
Complex_CH_unique <- Complex_CH[!duplicated(Complex_CH$data_id),] # extract unique ids for counting

Complex_CH_unique %>%
  summarise(
    Mean = mean(age),
    SD = sd(age)
  )
```

```
## # A tibble: 1 × 2
##    Mean    SD
##   <dbl> <dbl>
## 1  23.0  4.32
```

Numbers of females and males:

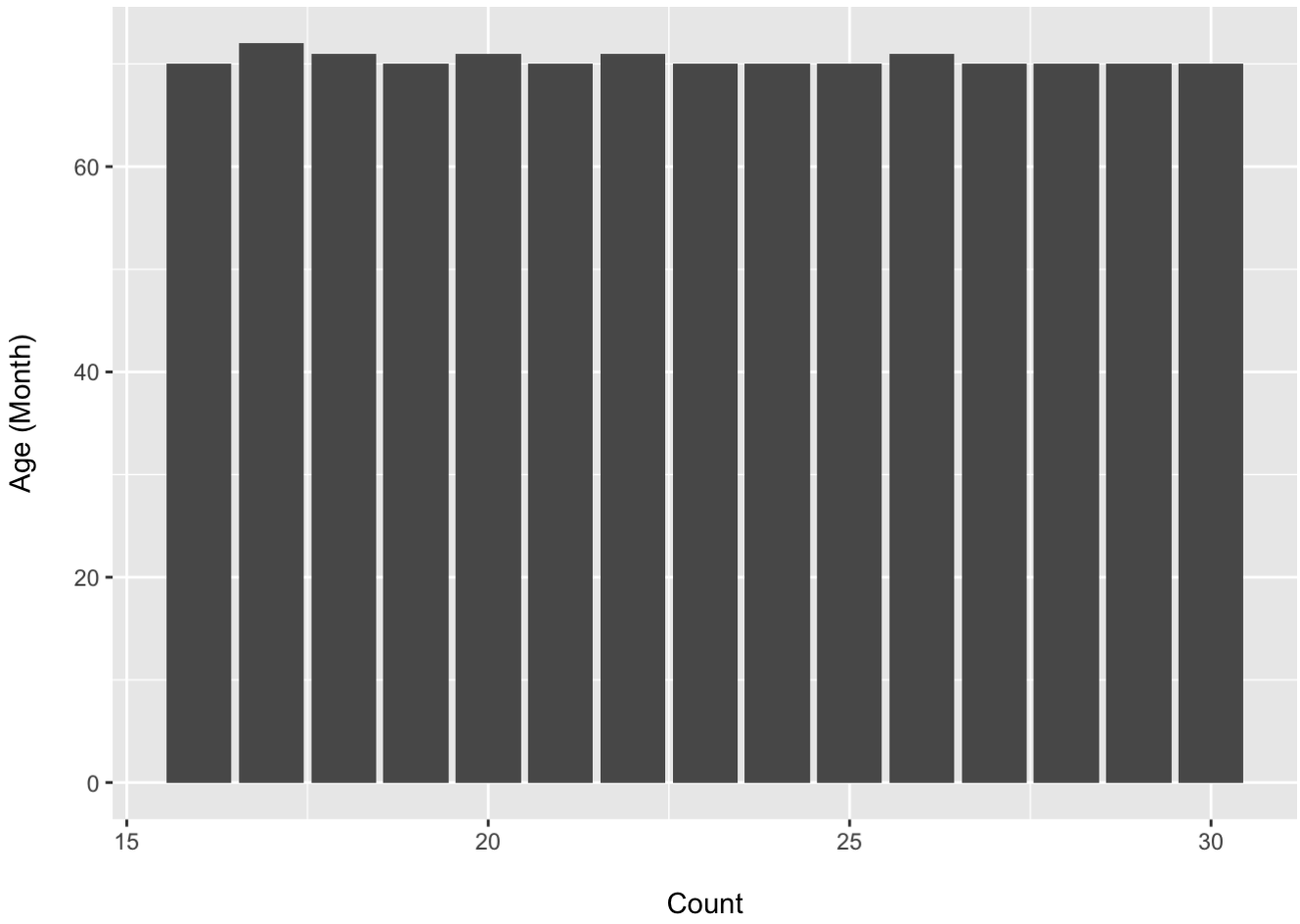
```
Complex_CH_gender <- as.data.frame(table(Complex_CH_unique$sex))
names(Complex_CH_gender)[1] <- 'Gender'
```

```
##   Gender Freq
## 1 Female  526
## 2   Male  530
## 3  Other    0
```

Plots

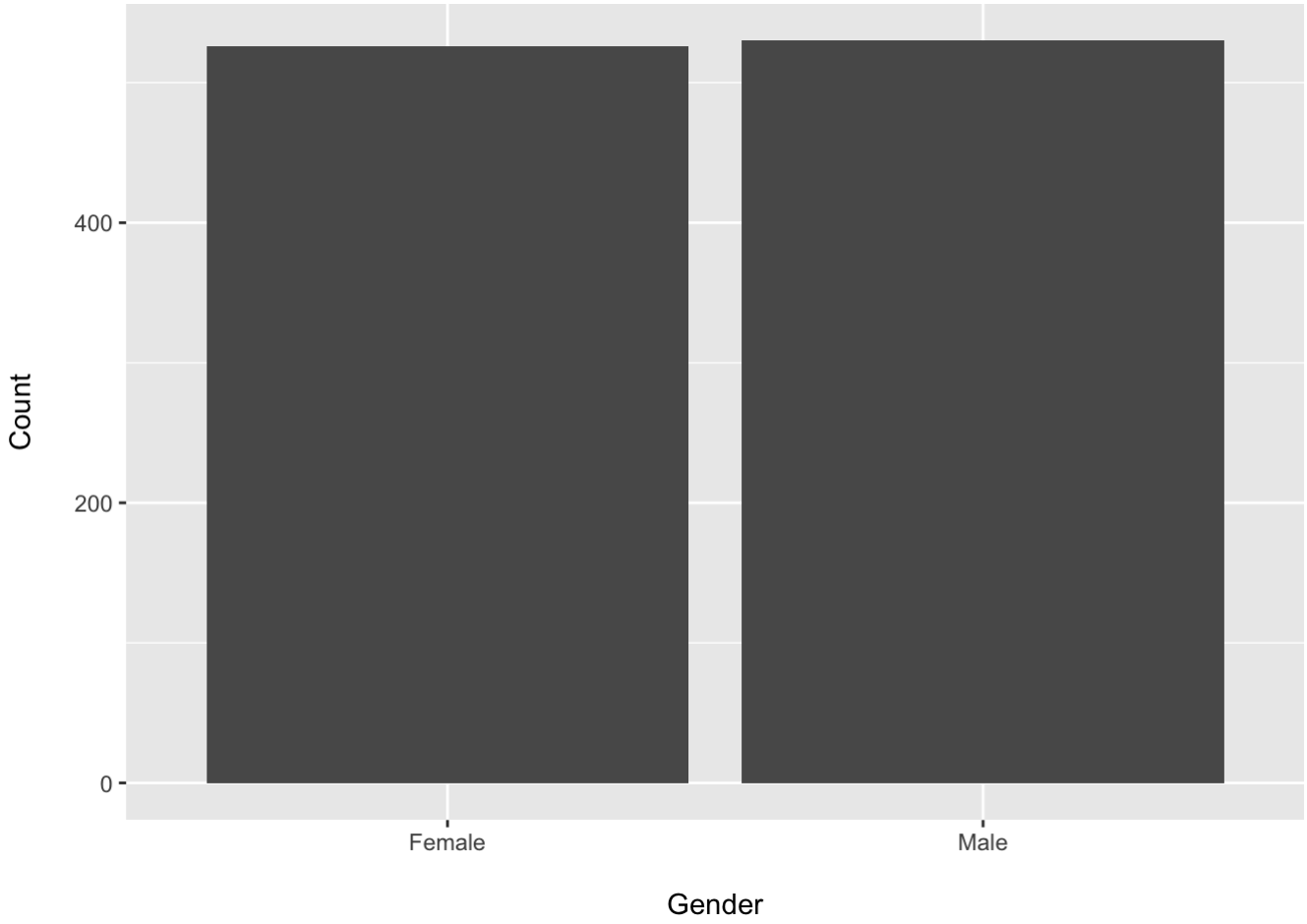
Age distribution:

```
ggplot(Complex_CH_unique, aes(x = age)) +
  geom_bar() +
  labs(x = paste('', 'Count', sep = '\n'), y = paste('Age (Month)', '', sep = '\n'))
```



Gender distribution:

```
ggplot(Complex_CH_unique, aes(x = sex)) +
  geom_bar() +
  labs(x = paste('', 'Gender', sep = '\n'), y = paste('Count', '', sep = '\n'))
```



Means and Standarad Deviations of Complexity Scores per Item

```
Complex_CH %>%
  full_join(., df_choices_CH_splited, by = 'item_id') %>%
  group_by(item_id) %>%
  summarise(
    choices = choices,
    mean = mean(value),
    sd = sd(value)
  ) %>%
  distinct(., item_id, .keep_all = TRUE) %>%
  kable(caption="Means and SDs for Each Item (Arranged by Item ID)") %>%
  kable_styling()
```

Means and SDs for Each Item (Arranged by Item ID)

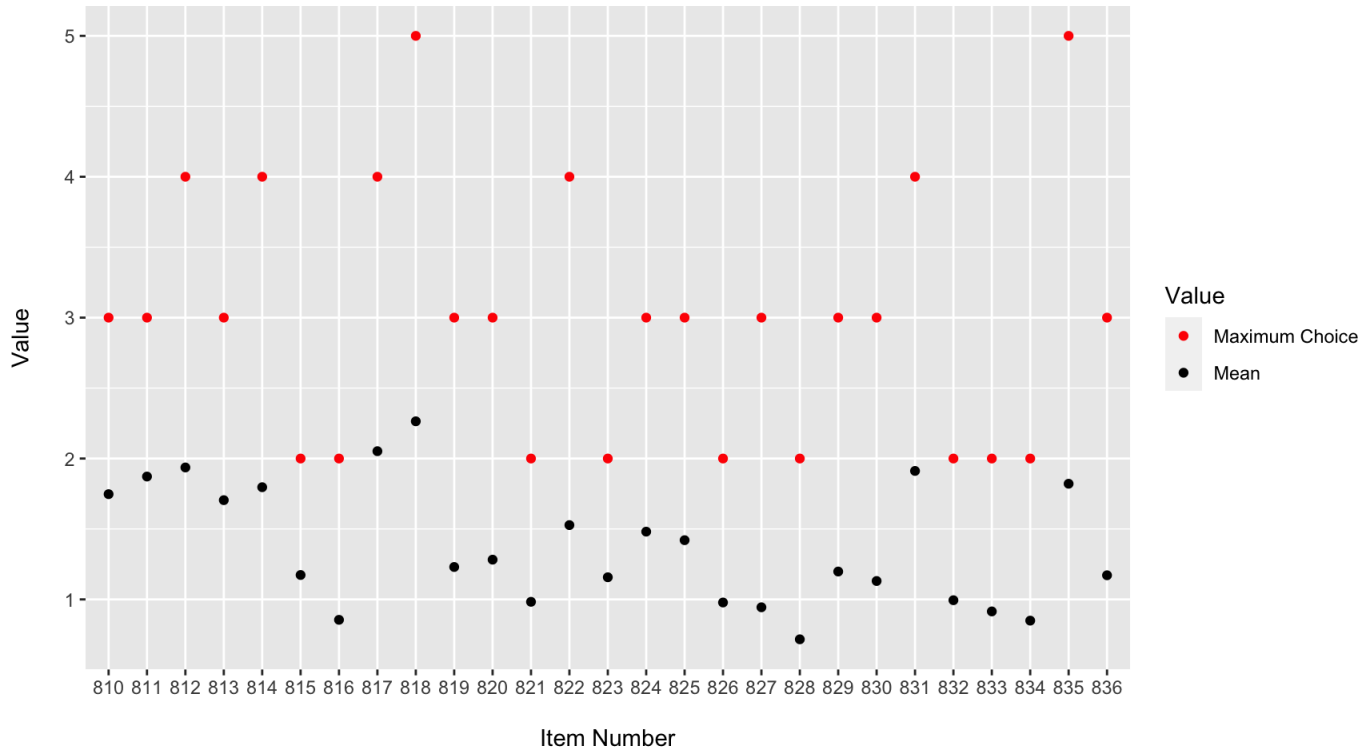
item_id	choices	mean	sd
item_810	3; 2; 1; 0	1.7471591	1.1953668
item_811	3; 2; 1; 0	1.8721591	1.2610616
item_812	4; 3; 2; 1; 0	1.9365530	1.6541629
item_813	3; 2; 1; 0	1.7045455	1.3615638

item_id	choices	mean	sd
item_814	4; 3; 2; 1; 0	1.7964015	1.6288938
item_815	2; 1; 0	1.1732955	0.8790812
item_816	2; 1; 0	0.8551136	0.8127144
item_817	4; 3; 2; 1; 0	2.0520833	1.6807061
item_818	5; 4; 3; 2; 1; 0	2.2642045	1.8603590
item_819	3; 2; 1; 0	1.2301136	1.2187642
item_820	3; 2; 1; 0	1.2821970	1.1706948
item_821	2; 1; 0	0.9829545	0.9058489
item_822	4; 3; 2; 1; 0	1.5274621	1.5989344
item_823	2; 1; 0	1.1571970	0.8975508
item_824	3; 2; 1; 0	1.4810606	1.3255138
item_825	3; 2; 1; 0	1.4204545	1.2631557
item_826	2; 1; 0	0.9782197	0.9289953
item_827	3; 2; 1; 0	0.9441288	1.1107845
item_828	2; 1; 0	0.7168561	0.7618511
item_829	3; 2; 1; 0	1.1979167	1.2833736
item_830	3; 2; 1; 0	1.1306818	1.0779856
item_831	4; 3; 2; 1; 0	1.9119318	1.8464285
item_832	2; 1; 0	0.9943182	0.9499144
item_833	2; 1; 0	0.9147727	0.9248182
item_834	2; 1; 0	0.8494318	0.8896773
item_835	5; 4; 3; 2; 1; 0	1.8210227	1.6218804
item_836	3; 2; 1; 0	1.1714015	1.3032865

Plot

Plot of means with their max choice value:

```
mean_value_CH <-  Complex_CH %>%
  full_join(., df_choices_CH_splited, by = 'item_id') %>%
  group_by(num_item_id) %>%
  summarise(mean = mean(value), max = max) %>%
  distinct(., num_item_id, .keep_all = TRUE)

mean_CH <- mean_value_CH %>%
  select(num_item_id, mean) %>%
  mutate(category = 'Mean')
colnames(mean_CH)[2] <- 'value'

max_CH <- mean_value_CH %>%
  select(num_item_id, max) %>%
  mutate(category = 'Maximum Choice')
colnames(max_CH)[2] <- 'value'

max_CH$value <- as.numeric(max_CH$value)

plot_CH <- rbind(mean_CH, max_CH)

plot_CH$num_item_id <- as.factor(plot_CH$num_item_id)

ggplot(plot_CH, aes(x = num_item_id, y = value, group = category, color = category))
 +
  geom_point() +
  scale_color_manual(values = c('Maximum Choice'='red', "Mean"="black")) +
  labs(x = paste('', 'Item Number', sep = '\n'), y = paste('Value', '', sep = '\n'),
 color = 'Value')
```



Penn Interactive Peer Play Scale (PIPPS) Data

Importing Data

```
PIPPS <- read_excel("CLCL_60mth_questionnaires_itemised_scored_blind.xlsx") %>%
  select(Blind_ID,starts_with("PIPPS"), -contains("_P")) %>% # removes variables othe
r than PIPPS_ + numbers
  pivot_longer(cols = starts_with('PIPPS'),
               names_to = 'item_id',
               values_to = 'value') %>%  # pivot from wide to long
  mutate(item_num = substring(item_id, 7)) # remove the "PIPPS_" to get the numbers o
nly
```

```
## # A tibble: 6 × 4
##   Blind_ID item_id value item_num
##   <chr>    <chr>   <dbl> <chr>
## 1 ScOCBHBM PIPPS_1     3 1
## 2 ScOCBHBM PIPPS_2     2 2
## 3 ScOCBHBM PIPPS_3     1 3
## 4 ScOCBHBM PIPPS_4     3 4
## 5 ScOCBHBM PIPPS_5     2 5
## 6 ScOCBHBM PIPPS_6     4 6
```

Number of Missing Values

Total Number of Missing Values

```
sum(is.na(PIPPS$value))
```

```
## [1] 1158
```

Number of Missing Values per Item

```
PIPPS_num_NAs <- PIPPS %>%
  filter(is.na(value)) %>%
  group_by(item_id) %>%
  count() %>%
  ungroup() %>% # reorder
  slice(match(PIPPS$item_id[1:32], item_id))
```

```
##     item_id  n
## 1   PIPPS_1 37
## 2   PIPPS_2 36
## 3   PIPPS_3 36
## 4   PIPPS_4 36
## 5   PIPPS_5 36
## 6   PIPPS_6 36
## 7   PIPPS_7 36
## 8   PIPPS_8 36
## 9   PIPPS_9 36
## 10 PIPPS_10 38
## 11 PIPPS_11 37
## 12 PIPPS_12 36
## 13 PIPPS_13 36
## 14 PIPPS_14 36
## 15 PIPPS_15 36
## 16 PIPPS_16 36
## 17 PIPPS_17 36
## 18 PIPPS_18 36
## 19 PIPPS_19 37
## 20 PIPPS_20 36
## 21 PIPPS_21 36
## 22 PIPPS_22 37
## 23 PIPPS_23 36
## 24 PIPPS_24 36
## 25 PIPPS_25 36
## 26 PIPPS_26 36
## 27 PIPPS_27 36
## 28 PIPPS_28 36
## 29 PIPPS_29 36
## 30 PIPPS_30 36
## 31 PIPPS_31 36
## 32 PIPPS_32 36
```

Means and Standard Deviations

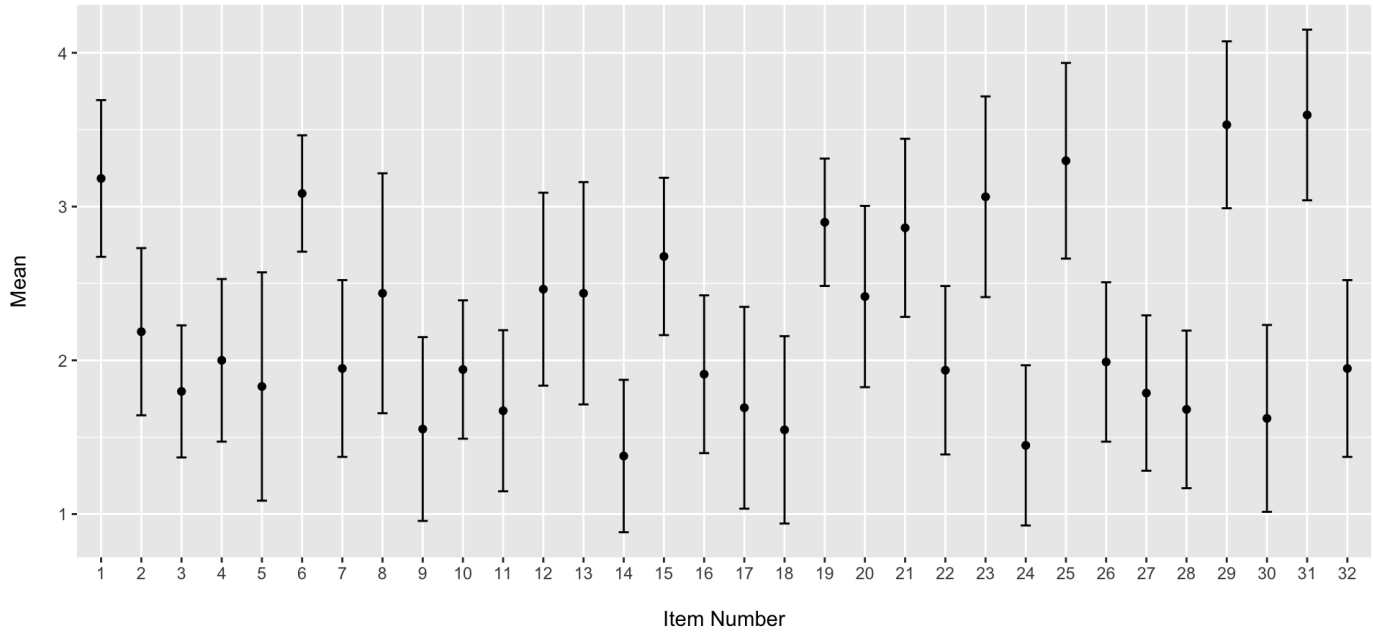
Means and SDs by Item Number

```
PIPPS_mean_sd <- PIPPS %>%
  group_by(item_num) %>%
  summarise(
    mean = mean(value, na.rm = TRUE),
    sd = sd(value, na.rm = TRUE)) %>%
  ungroup() %>% # reorder
  slice(match(PIPPS$item_num[1:32], item_num))
```

```
##    item_num     mean        sd
## 1         1 3.182796 0.5096085
## 2         2 2.186170 0.5437032
## 3         3 1.797872 0.4295481
## 4         4 2.000000 0.5287437
## 5         5 1.829787 0.7425370
## 6         6 3.085106 0.3784398
## 7         7 1.946809 0.5748683
## 8         8 2.436170 0.7802469
## 9         9 1.553191 0.5977920
## 10       10 1.940217 0.4499230
## 11       11 1.672043 0.5238581
## 12       12 2.462766 0.6275030
## 13       13 2.436170 0.7230240
## 14       14 1.377660 0.4956053
## 15       15 2.675532 0.5117304
## 16       16 1.909574 0.5130699
## 17       17 1.691489 0.6562617
## 18       18 1.547872 0.6093763
## 19       19 2.897849 0.4141462
## 20       20 2.414894 0.5895063
## 21       21 2.861702 0.5792294
## 22       22 1.935484 0.5478506
## 23       23 3.063830 0.6526787
## 24       24 1.446809 0.5208971
## 25       25 3.297872 0.6363503
## 26       26 1.989362 0.5183655
## 27       27 1.787234 0.5052912
## 28       28 1.680851 0.5124843
## 29       29 3.531915 0.5428346
## 30       30 1.622340 0.6076845
## 31       31 3.595745 0.5549222
## 32       32 1.946809 0.5748683
```

Plot

```
PIPPS_mean_sd$item_num <- factor(PIPPS_mean_sd$item_num, levels = unique(PIPPS_mean_s
d$item_num)) # keep the order

ggplot(data = PIPPS_mean_sd, aes(x = item_num, y = mean)) +
  geom_point() +
  geom_errorbar(aes(x=item_num, ymin=mean-sd, ymax=mean+sd), width=0.25) +
  labs(x = paste('', 'Item Number', sep = '\n'), y = paste('Mean', '', sep = '\n'),
       caption = paste(" ", "*Error bar: Standard Deviation", sep = "\n"))
```



*Error bar: Standard Deviation

Dimensionality Assessment for Complexity Scale

Wordbank Data

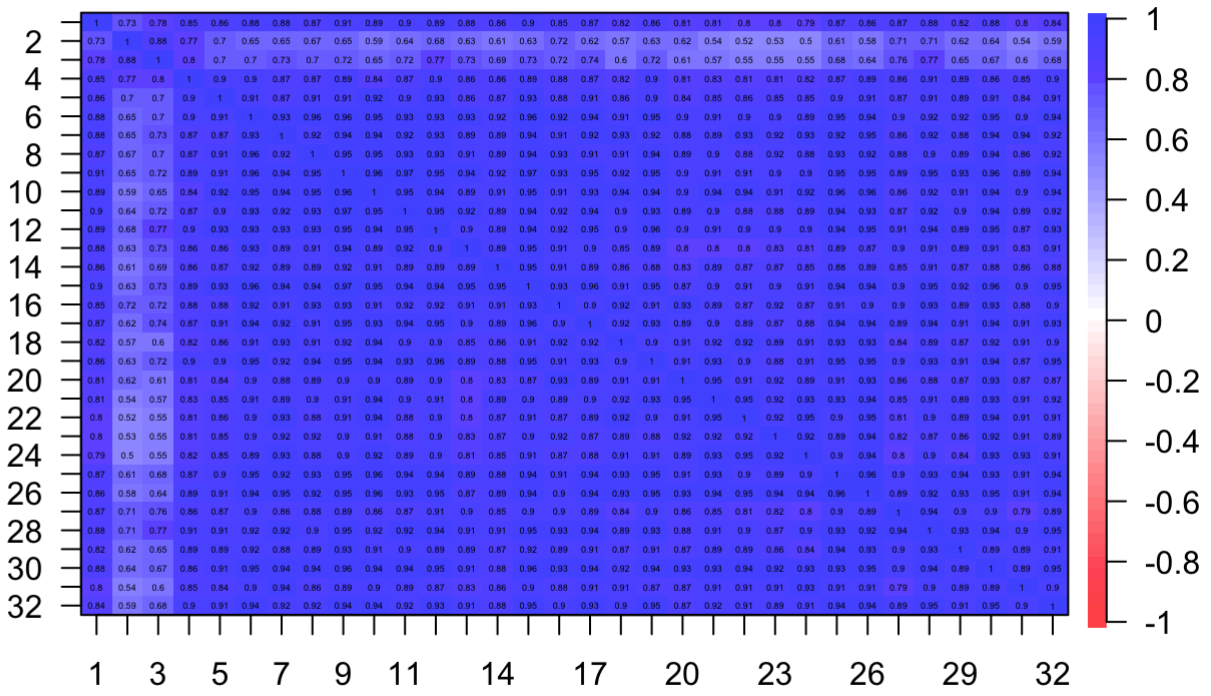
1. Korean

Data preparation:

```
Complex_KO_short_with_ids <- Complex_KO %>%
  dplyr::select(data_id, out, num_item_id) %>%
  pivot_wider(id_cols=data_id, names_from = "num_item_id", values_from="out")

Complex_KO_short <- Complex_KO_short_with_ids %>%
  dplyr::select(-data_id)
```

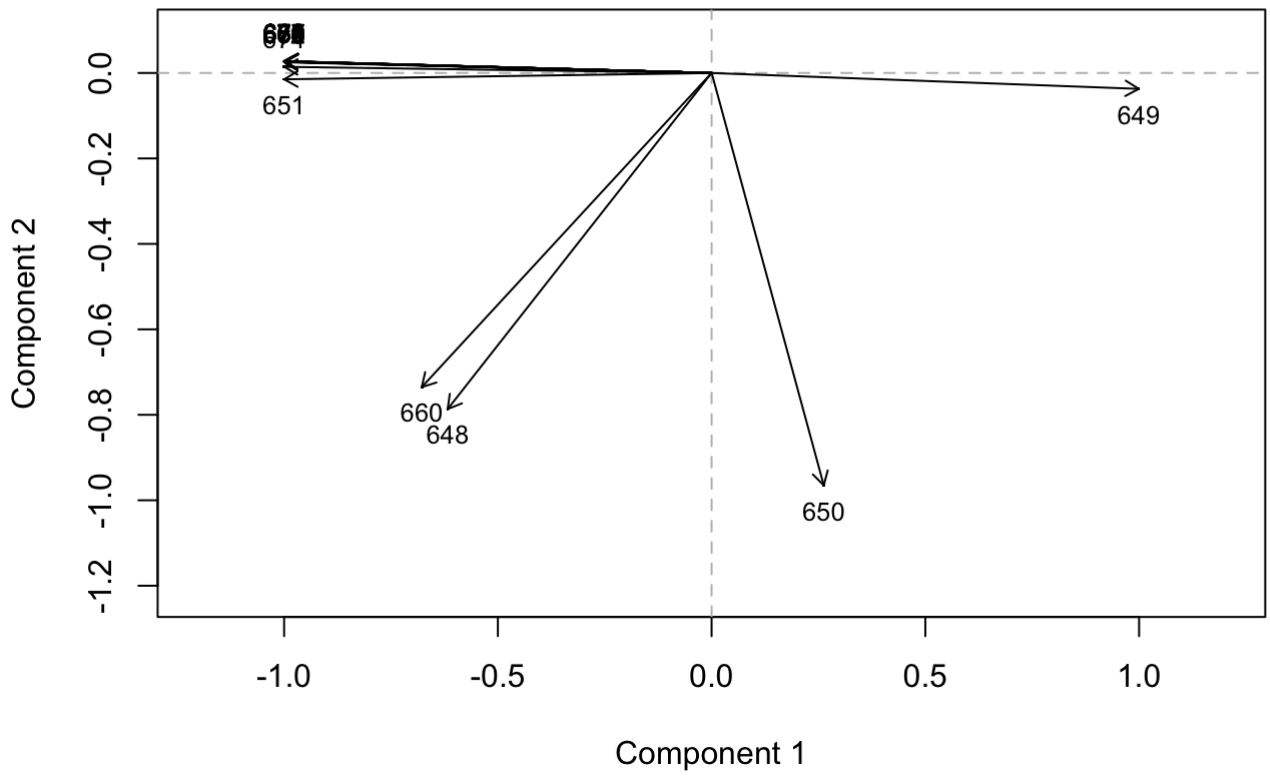
1.1. Polychoric Correlation Matrix

```
Complex_KO_poly <- polychoric(Complex_KO_short)
rho_KO <- Complex_KO_poly$rho

lab_KO = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12", "13", "14",
"15", "16", "17", "18", "19", "20", "21", "22", "23", "24", "25", "26", "27", "28",
"29", "30", "31", "32") # create labels for corPlot

corPlot(rho_KO, labels=lab_KO)
```



1.2. Principal Component Analysis

```
pc_KO <- princals(rho_KO)
plot(pc_KO)
```



Item 660, 648, 650, and 649 seem to be behaving differently from the rest, and therefore should be examined.

Extract items to be examined:

```
extract <- c(648, 649, 650, 660) # items to be examined
Complex_KO_extracted <- subset(Complex_KO, num_item_id %in% extract)
```

```
## # A tibble: 6 × 29
##   data_id   age comprehension production language.x form.x birth_order ethnicity
##     <dbl> <int>         <int>      <int> <chr>      <chr>  <fct>       <fct>
## 1  166783    20           176         64 Korean     WS     First       <NA>
## 2  166783    20           176         64 Korean     WS     First       <NA>
## 3  166783    20           176         64 Korean     WS     First       <NA>
## 4  166783    20           176         64 Korean     WS     First       <NA>
## 5  166784    20           446        196 Korean     WS     First       <NA>
## 6  166784    20           446        196 Korean     WS     First       <NA>
## # … with 21 more variables: sex <fct>, zygosity <chr>, norming <lgl>,
## #   mom_ed <fct>, longitudinal <lgl>, source_name <chr>, license <chr>,
## #   value <chr>, num_item_id <dbl>, item_id <chr>, definition <chr>,
## #   language.y <chr>, form.y <chr>, type <chr>, category <chr>,
## #   lexical_category <chr>, lexical_class <chr>, uni_lemma <chr>,
## #   complexity_category <chr>, translation <chr>, out <dbl>
```

Means and SDs of complexity scores per item:

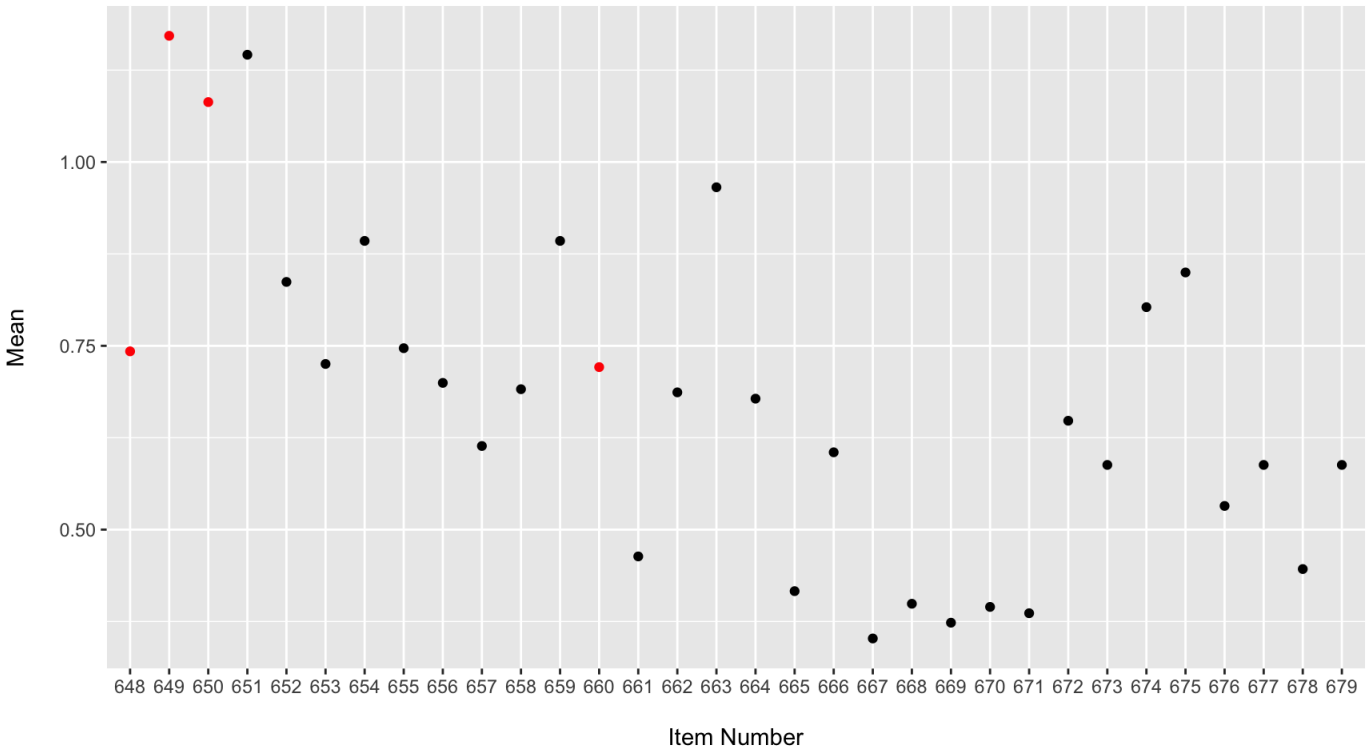
```
Complex_KO_extracted %>%
  group_by(num_item_id, translation) %>%
  summarise(
    mean = mean(out),
    sd = sd(out),
  ) %>%
  kable(caption="Means and SDs for Each Item (Arranged by Item ID)")
```

Means and SDs for Each Item (Arranged by Item ID)

num_item_id	translation	mean	sd
648	big thing flower / big flower	0.7424893	0.9110182
649	sister's thing snack / sister's snack	1.1716738	0.9123980
650	dad's thing watch / dad's watch	1.0815451	0.9084153
660	let's puts (it) in here * wrong inflection / let's put (it) in here	0.7210300	0.9070294

Means in comparison to other items:

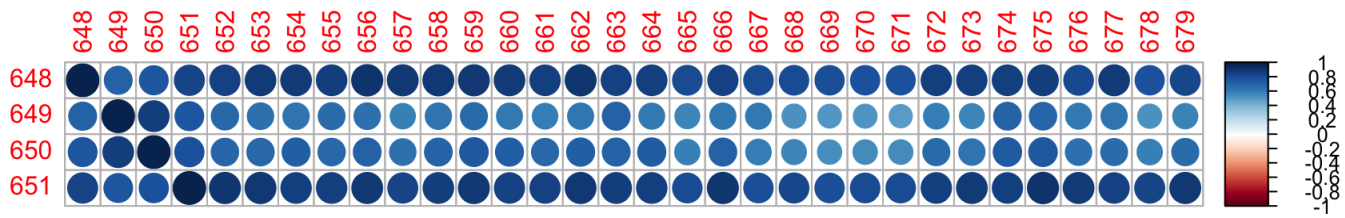
```
complex_scores_KO <- Complex_KO %>%
  group_by(num_item_id) %>%
  summarise(
    mean = mean(out),
    sd = sd(out),
  ) %>%
  mutate(ToHighlight = ifelse(num_item_id %in% extract, 'yes', 'no')) # Assign variab
les to be highlighted

complex_scores_KO$num_item_id <- as.factor(complex_scores_KO$num_item_id) # Assign nu
mber of item as a factor for plotting

ggplot(data = complex_scores_KO, aes(x = num_item_id, y = mean, color = ToHighlight))
+
  geom_point() +
  scale_color_manual(values = c('yes'='red', "no"="black"), guide = 'none') +
  labs(x = paste('', 'Item Number', sep = '\n'), y = paste('Mean', '', sep = '\n')) #
'' to give space between data points and the labels
```




```
  # with error bars:
  # + geom_errorbar(aes(x=num_item_id, ymin=mean-sd, ymax=mean+sd), width=0.25)
```

Correlation with the rest of the items:

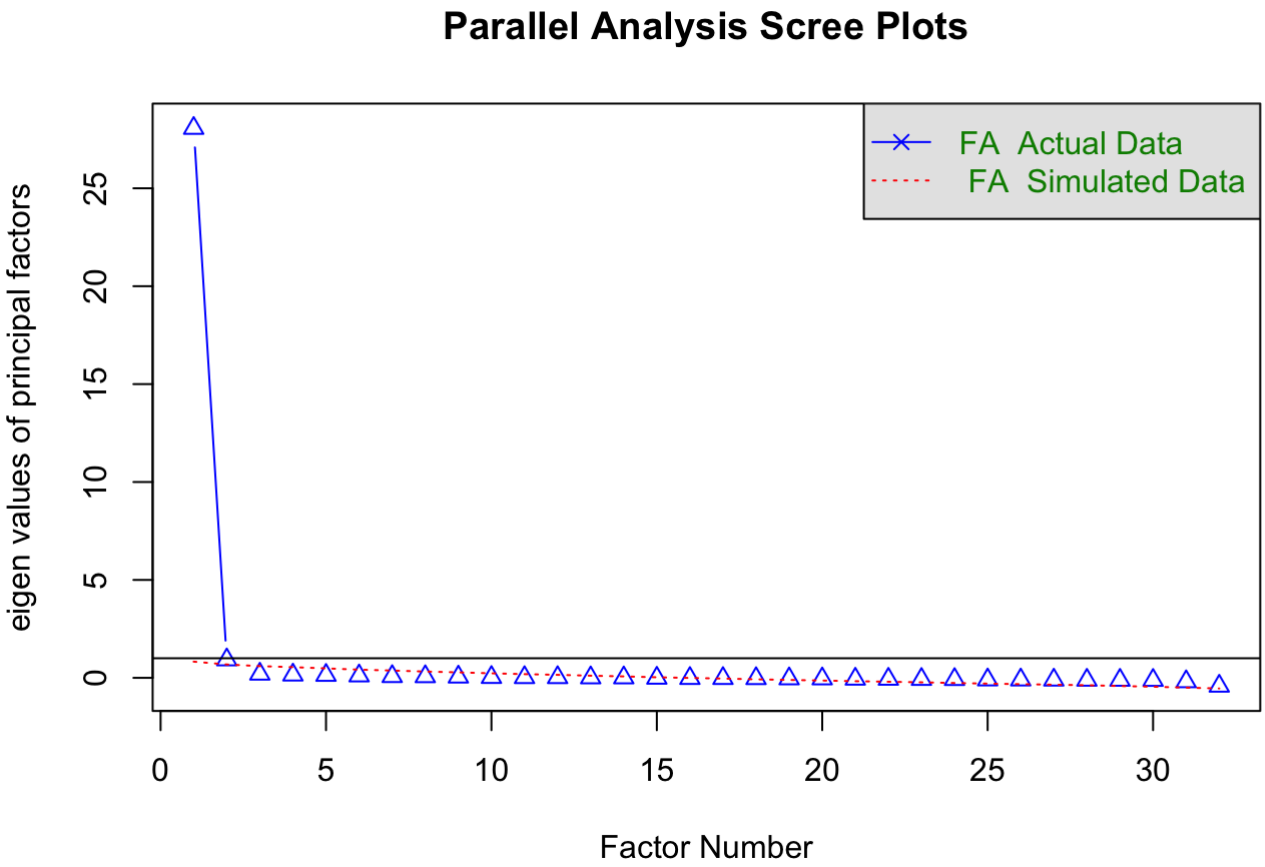
```
rho_KO_extracted <- rho_KO[, as.factor(extract)] # extract items from polychoric corr
elation

rev_rho_KO_extracted <- rho_KO_extracted[nrow(rho_KO_extracted):1,] # for alignment
rotate <- function(x) t(apply(x, 2, rev)) # for alignment

corrplot(rotate(rev_rho_KO_extracted)) # plot
```



1.3. Parallel Analysis

```
fa.parallel(rho_KO, fa="fa", cor="poly", n.obs = 233)
```

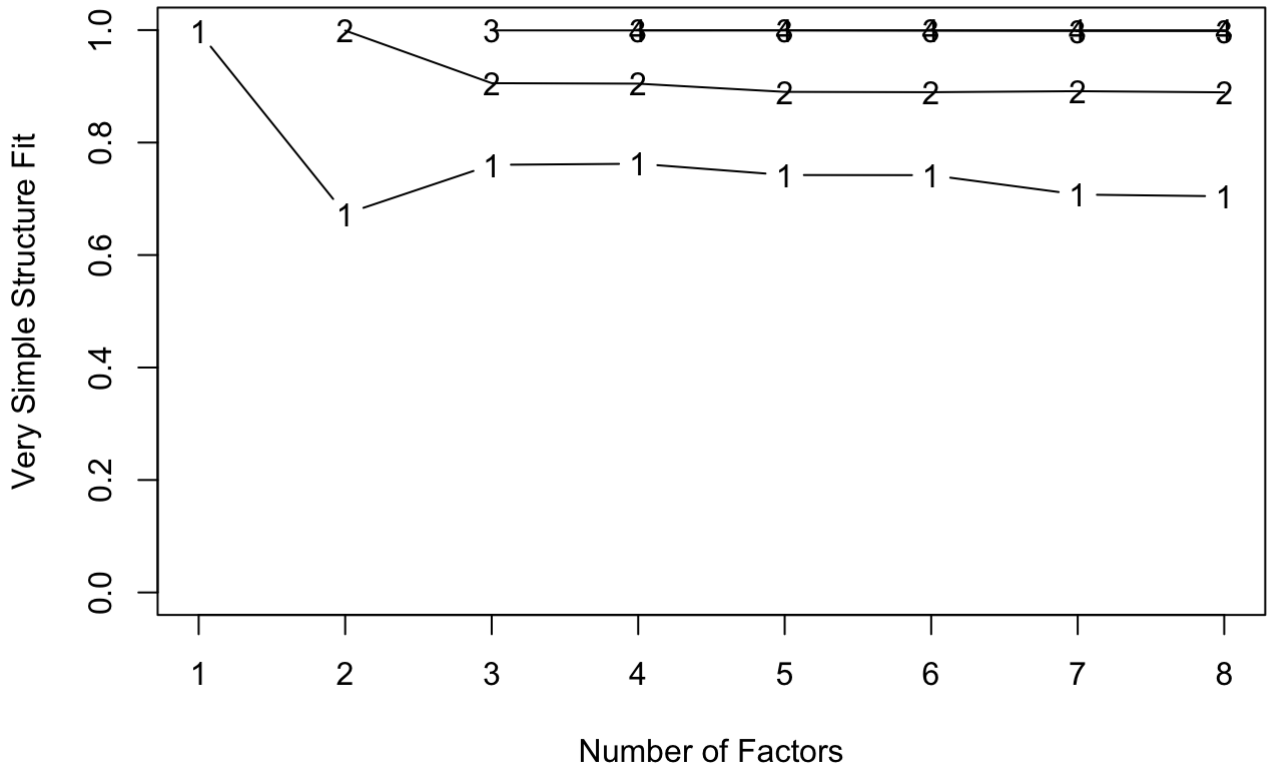
```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect. Try a
## different factor score estimation method.
```




```
## Parallel analysis suggests that the number of factors =  2  and the number of comp
onents =  NA
```

1.4. Very Simple Structure

```
vss(rho_KO, cor="poly", n.obs = 233, fm = "ml")
```


Very Simple Structure
Very Simple Structure Fit
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
7
8
Number of Factors

```
##
## Very Simple Structure
## Call: vss(x = rho_KO, fm = "ml", n.obs = 233, cor = "poly")
## VSS complexity 1 achieves a maximimum of 1  with  1  factors
## VSS complexity 2 achieves a maximimum of 1  with  2  factors
##
## The Velicer MAP achieves a minimum of 0.04  with  2  factors
## BIC achieves a minimum of  35703.29  with  8  factors
## Sample Size adjusted BIC achieves a minimum of  36552.72  with  8  factors
##
## Statistics by number of factors
##   vss1 vss2   map dof chisq prob sqresid fit RMSEA   BIC SABIC complex eChisq
## 1 1.00 0.00 0.053 464 41206    0    2.19   1  0.61 38677 40148     1.0    303
## 2 0.67 1.00 0.042 433 40108    0    0.67   1  0.63 37748 39120     1.7     71
## 3 0.76 0.91 0.042 403 39514    0    0.39   1  0.65 37317 38595     2.1     41
## 4 0.76 0.90 0.045 374 38992    0    0.34   1  0.67 36953 38139     2.1     35
## 5 0.74 0.89 0.048 346 38471    0    0.31   1  0.69 36585 37681     2.2     31
## 6 0.74 0.89 0.052 319 37993    0    0.27   1  0.71 36254 37265     2.3     27
## 7 0.71 0.89 0.056 293 37598    0    0.20   1  0.74 36000 36929     2.3     20
## 8 0.70 0.89 0.064 268 37164    0    0.18   1  0.77 35703 36553     2.3     18
##     SRMR eCRMS  eBIC
## 1 0.0362 0.037 -2226
## 2 0.0176 0.019 -2289
## 3 0.0133 0.015 -2156
## 4 0.0124 0.014 -2003
## 5 0.0117 0.014 -1855
## 6 0.0107 0.013 -1712
## 7 0.0092 0.012 -1577
## 8 0.0089 0.012 -1443
```

2. Spanish

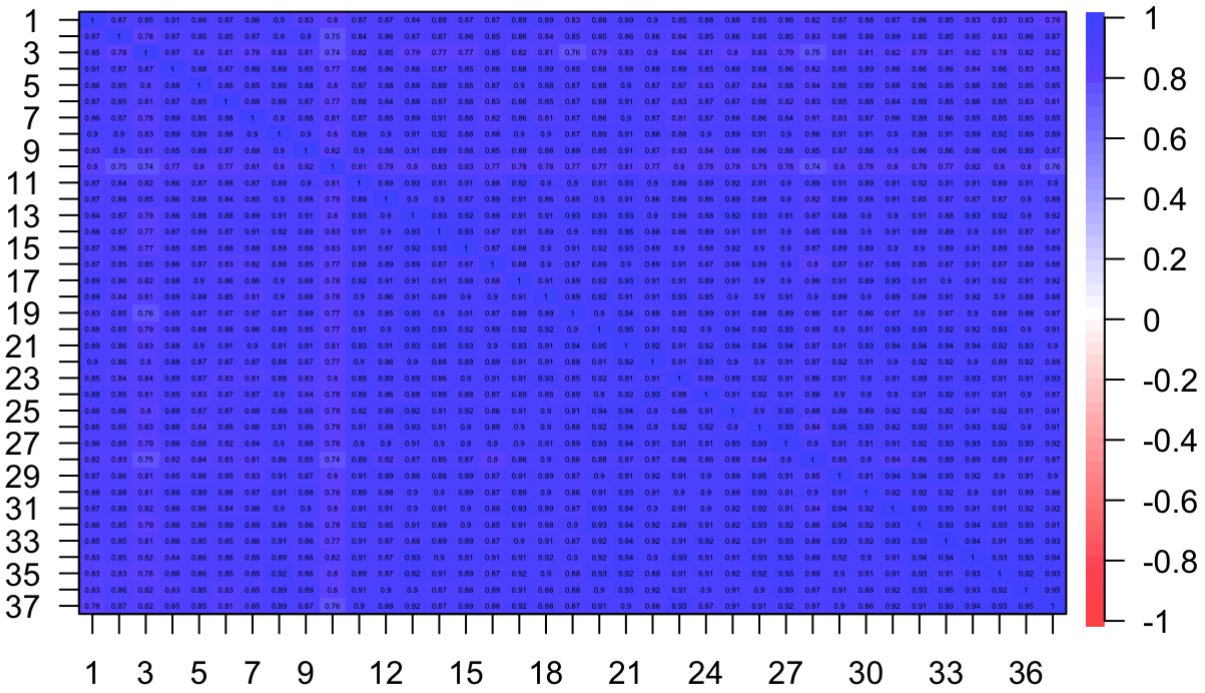
Data preparation:

```
Complex_ES_short_with_ids <- Complex_ES %>%
  dplyr::select(data_id, out, num_item_id) %>%
  pivot_wider(id_cols=data_id, names_from = "num_item_id", values_from="out")

Complex_ES_short <- Complex_ES_short_with_ids %>%
  dplyr::select(-data_id)
```

2.1. Tetrachoric Correlation Matrix

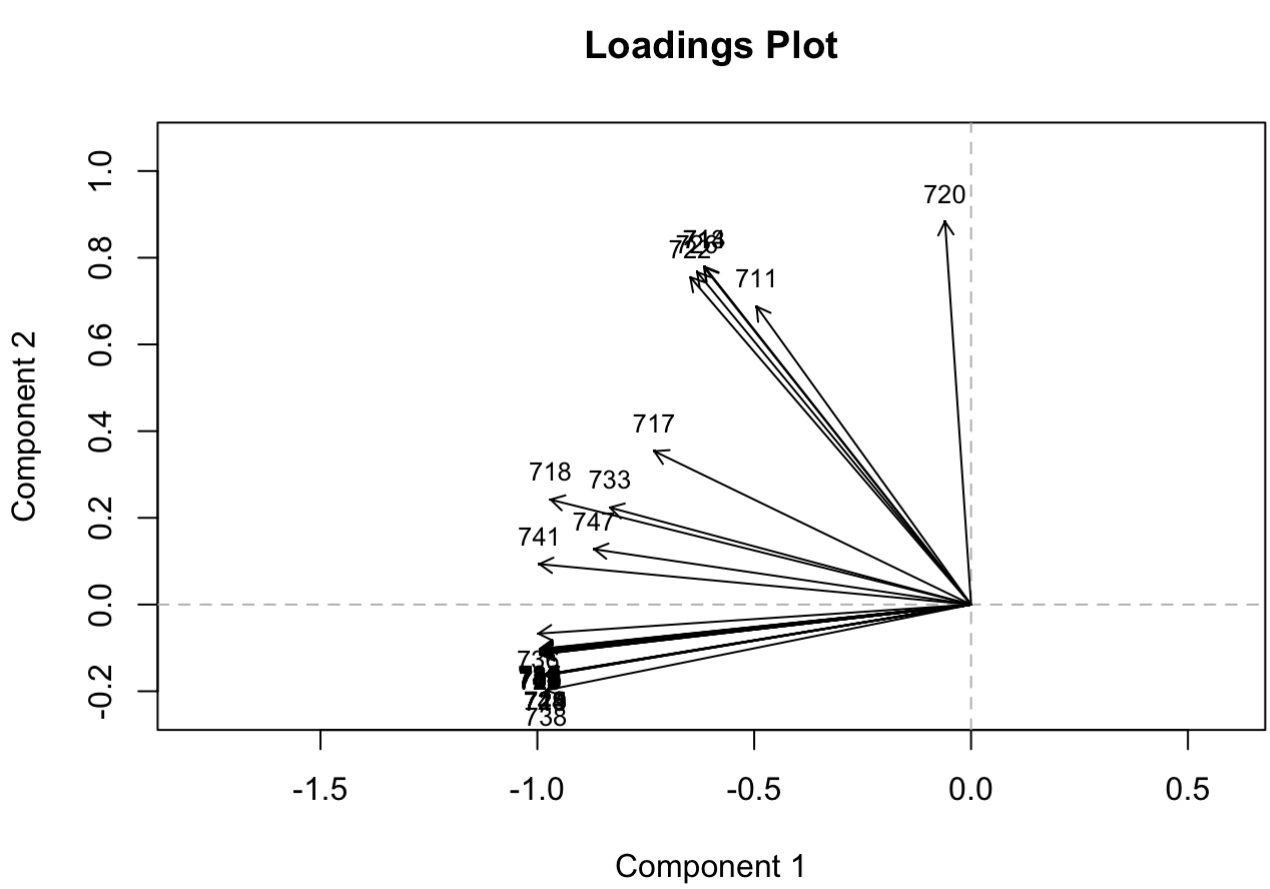
```
Complex_ES_tetra <- tetrachoric(Complex_ES_short)
```

```
rho_ES <- Complex_ES_tetra$rho

lab_ES = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12", "13", "14",
"15", "16", "17", "18", "19", "20", "21", "22", "23", "24", "25", "26", "27", "28",
"29", "30", "31", "32", "33", "34", "35", "36", "37")

corPlot(rho_ES, labels=lab_ES)
```



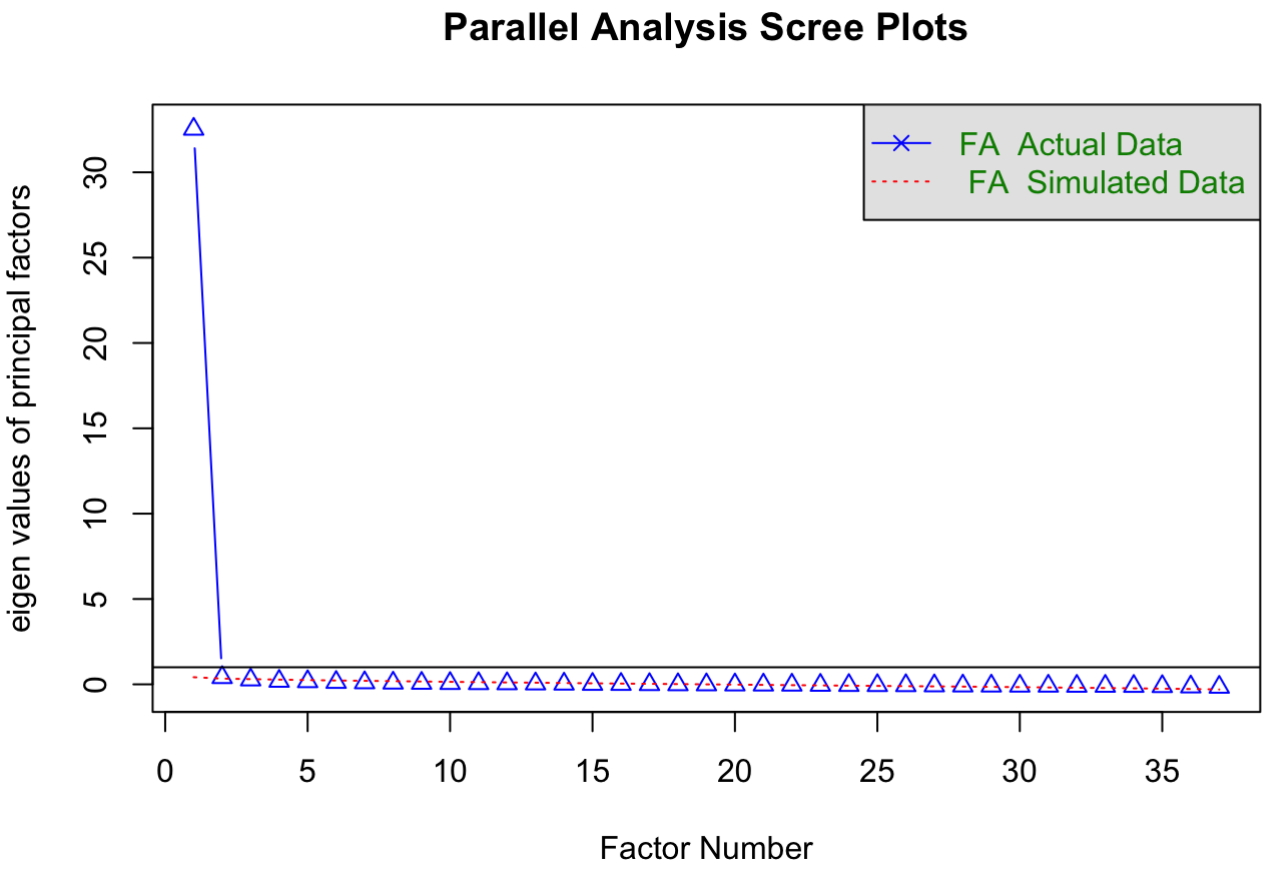
2.2. Principal Component Analysis

```
pc_ES <- princals(rho_ES)
plot(pc_ES)
```



2.3. Parallel Analysis

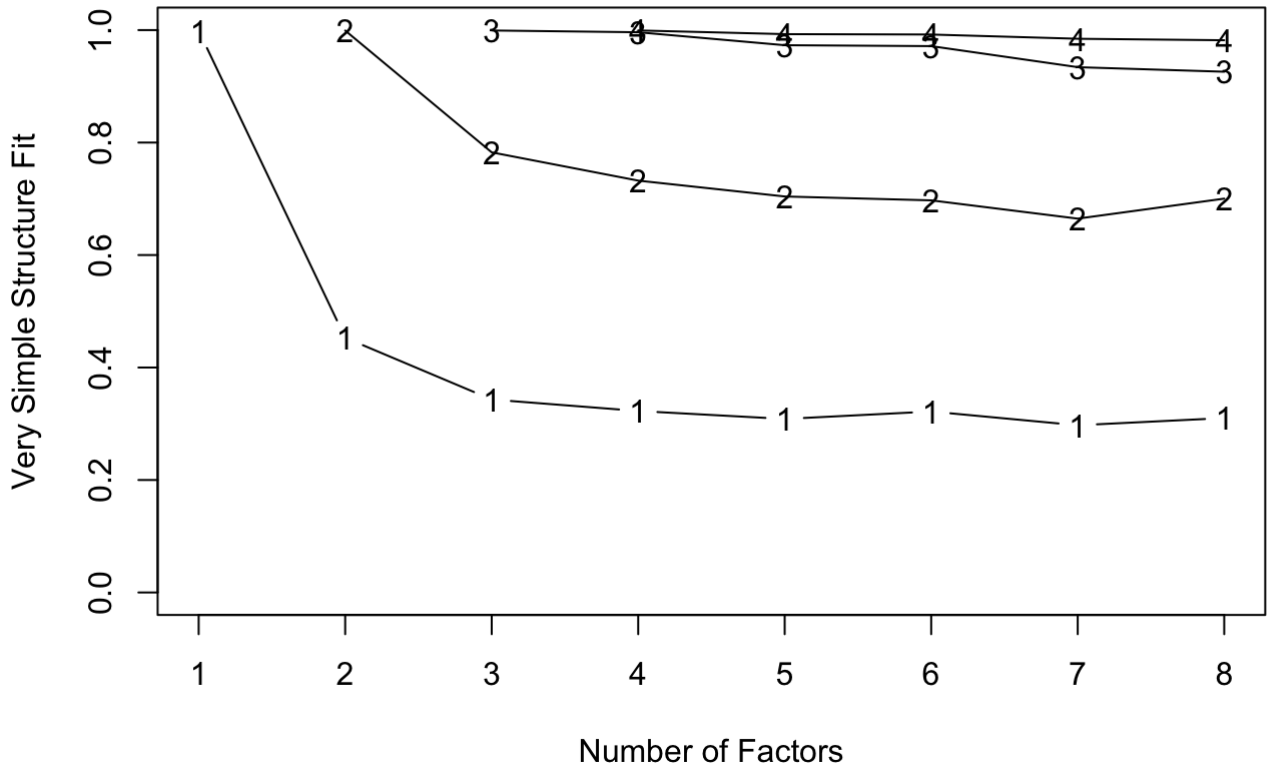
```
fa.parallel(rho_ES, fa="fa", cor="poly", n.obs = 1092, fm = "ml")
```




```
## Parallel analysis suggests that the number of factors =  2  and the number of comp
onents =  NA
```

2.4. Very Simple Structure

```
vss(rho_ES, cor="poly", n.obs = 1092, fm = "ml")
```


Very Simple Structure
Very Simple Structure Fit
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
7
8
Number of Factors

```
##
## Very Simple Structure
## Call: vss(x = rho_ES, fm = "ml", n.obs = 1092, cor = "poly")
## VSS complexity 1 achieves a maximimum of 1  with  1  factors
## VSS complexity 2 achieves a maximimum of 1  with  2  factors
##
## The Velicer MAP achieves a minimum of 0.03  with  3  factors
## BIC achieves a minimum of  125942.8  with  8  factors
## Sample Size adjusted BIC achieves a minimum of  127206.9  with  8  factors
##
## Statistics by number of factors
##   vss1 vss2   map dof  chisq prob sqresid fit RMSEA    BIC  SABIC complex
## 1 1.00 0.00 0.028 629 139519    0    1.16   1  0.45 135119 137116     1.0
## 2 0.45 1.00 0.026 593 137104    0    0.86   1  0.46 132955 134839     1.9
## 3 0.34 0.78 0.026 558 135328    0    0.75   1  0.47 131424 133197     2.7
## 4 0.32 0.73 0.028 524 133572    0    0.62   1  0.48 129906 131571     3.0
## 5 0.31 0.70 0.030 491 132115    0    0.54   1  0.50 128680 130240     3.3
## 6 0.32 0.70 0.031 459 131195    0    0.49   1  0.51 127984 129442     3.4
## 7 0.30 0.66 0.032 428 129797    0    0.48   1  0.53 126803 128162     3.7
## 8 0.31 0.70 0.034 398 128727    0    0.41   1  0.54 125943 127207     3.7
##   eChisq   SRMR eCRMS  eBIC
## 1    561 0.0196 0.020 -3840
## 2    362 0.0158 0.017 -3787
## 3    302 0.0144 0.016 -3601
## 4    232 0.0126 0.014 -3434
## 5    187 0.0113 0.013 -3248
## 6    170 0.0108 0.013 -3041
## 7    174 0.0109 0.014 -2820
## 8    140 0.0098 0.013 -2644
```

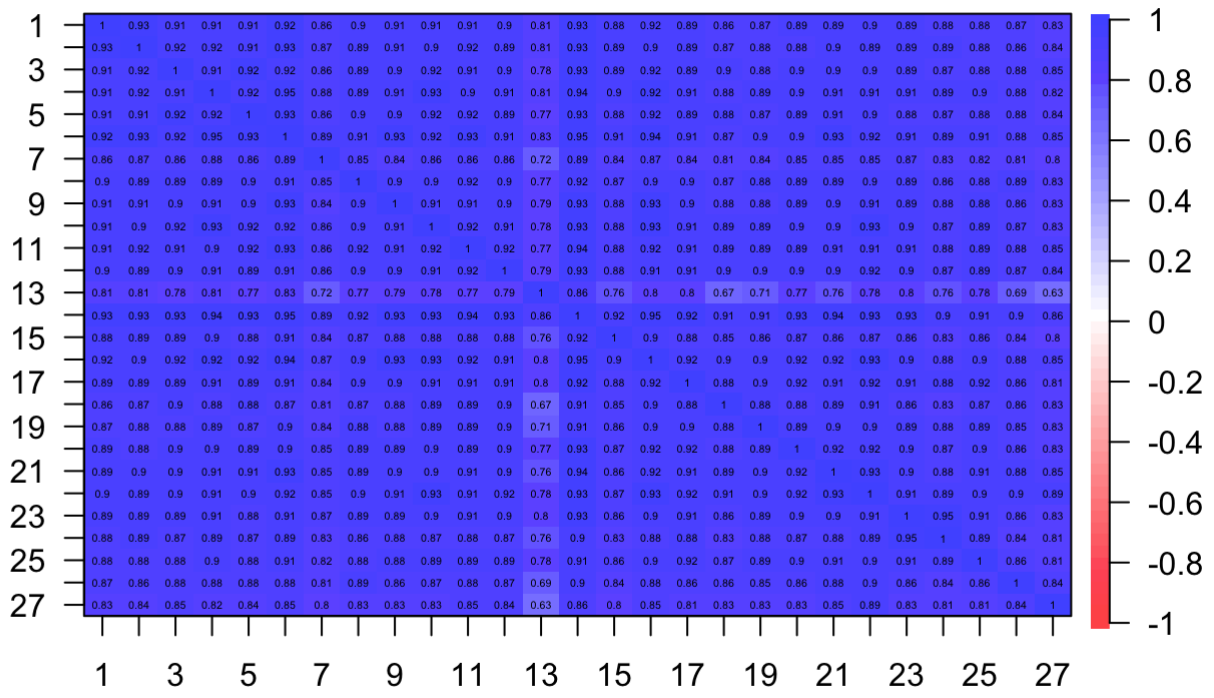
3. Chinese

Data preparation:

```
Complex_CH_short_with_ids <- Complex_CH %>%
  dplyr::select(data_id, value, num_item_id) %>%
  pivot_wider(id_cols=data_id, names_from = "num_item_id", values_from="value")

Complex_ES_short <- Complex_CH_short_with_ids %>%
  dplyr::select(-data_id)
```

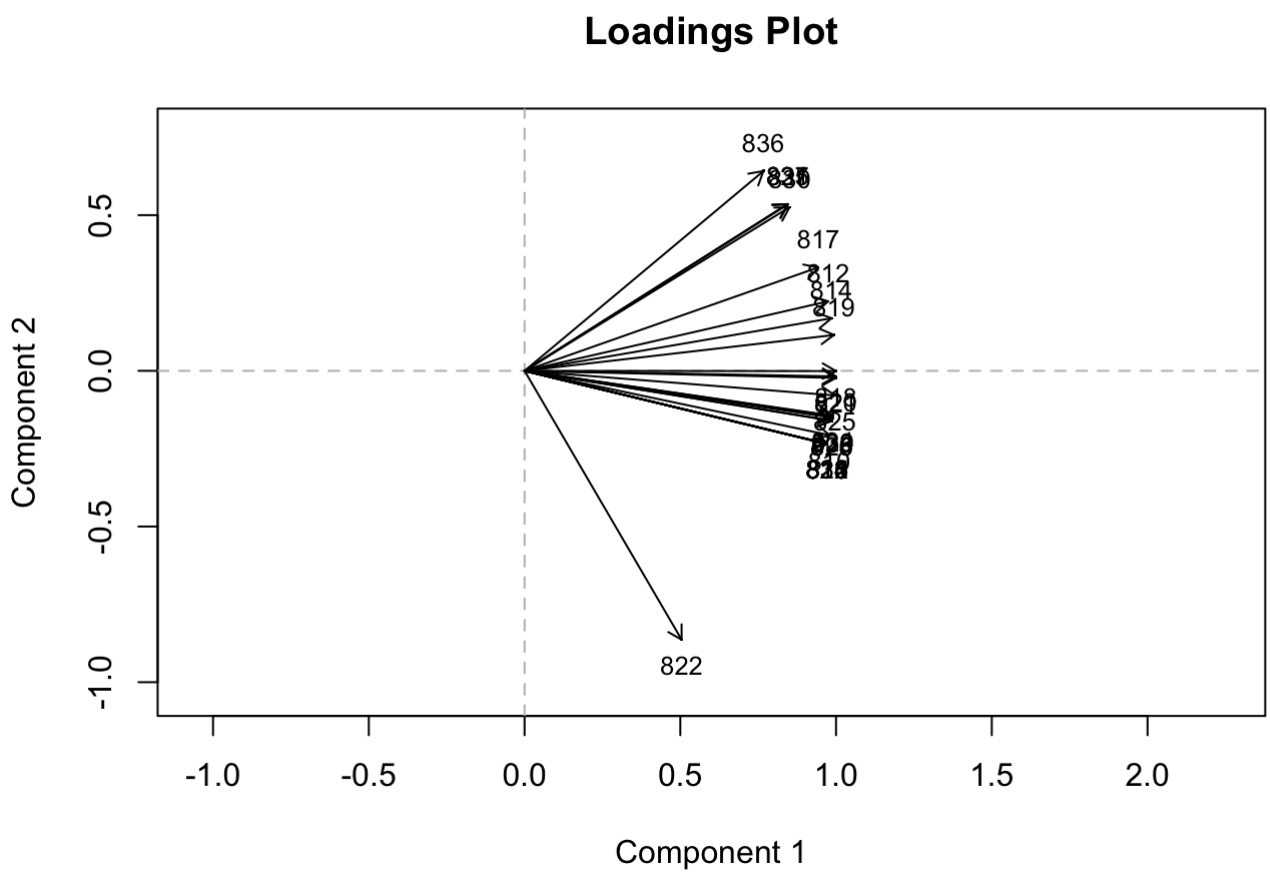
3.1. Polychoric Correlation Matrix

```
Complex_CH_poly <- polychoric(Complex_ES_short)
rho_CH <- Complex_CH_poly$rho

lab_CH = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12", "13", "14",
"15", "16", "17", "18", "19", "20", "21", "22", "23", "24", "25", "26", "27")

corPlot(rho_CH, labels=lab_CH)
```



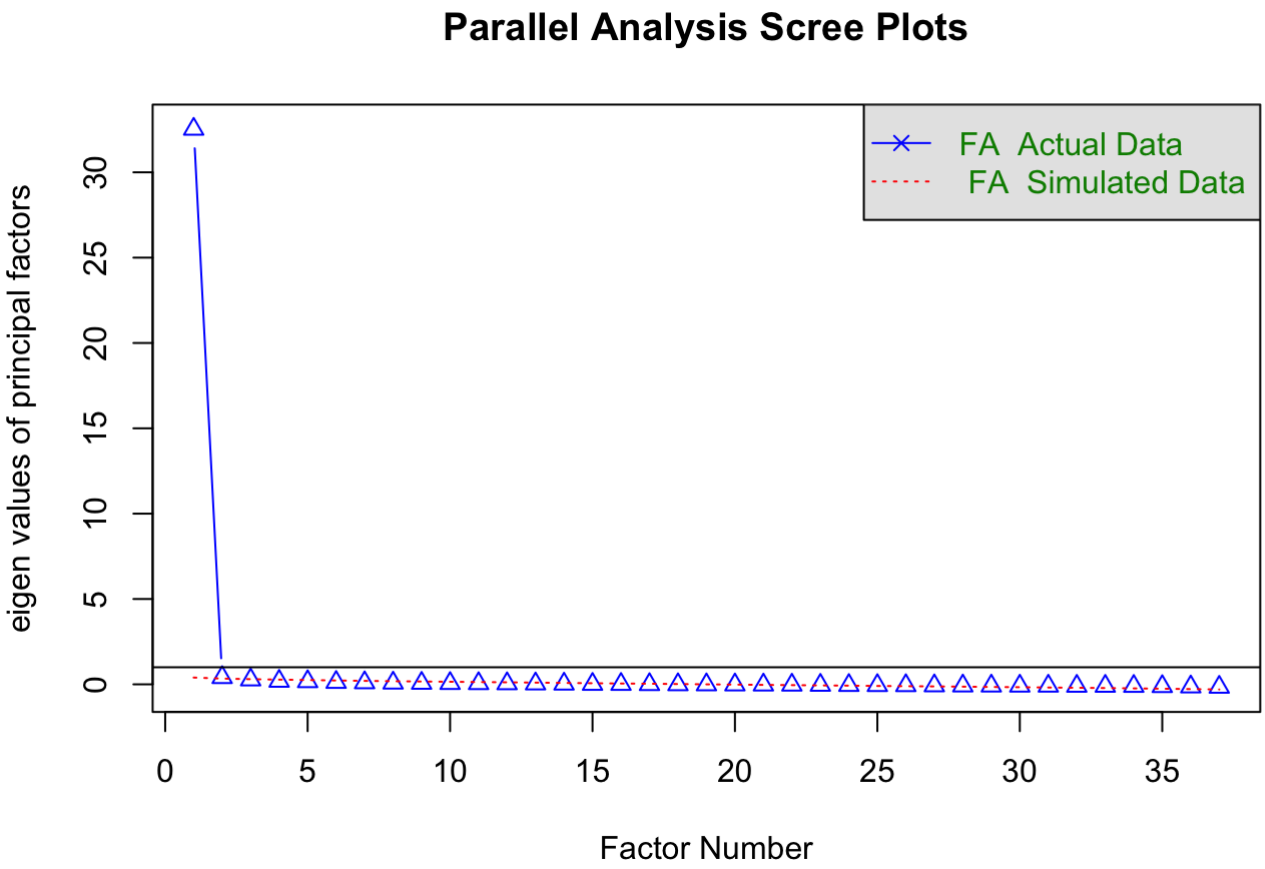
3.2. Principal Component Analysis

```
pc_CH <- princals(rho_CH)
plot(pc_CH)
```



3.3. Parallel Analysis

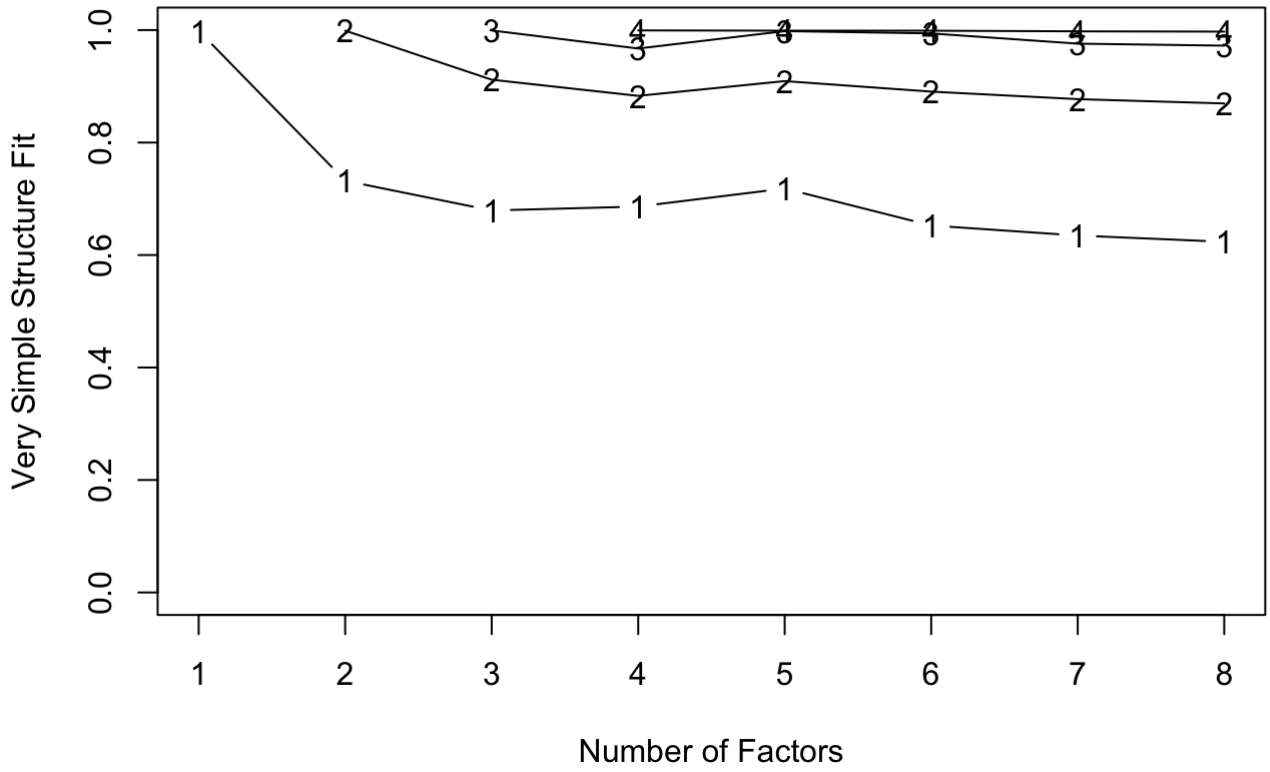
```
fa.parallel(rho_ES, fa="fa", cor="poly", n.obs = 1056, fm = "ml")
```




```
## Parallel analysis suggests that the number of factors =  2  and the number of comp
onents =  NA
```

3.4. Very Simple Structure

```
vss(rho_CH, cor="poly", n.obs = 1056, fm = "ml")
```


Very Simple Structure
Very Simple Structure Fit
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
7
8
Number of Factors

```
##
## Very Simple Structure
## Call: vss(x = rho_CH, fm = "ml", n.obs = 1056, cor = "poly")
## VSS complexity 1 achieves a maximimum of 1  with  1  factors
## VSS complexity 2 achieves a maximimum of 1  with  2  factors
##
## The Velicer MAP achieves a minimum of 0.01  with  1  factors
## BIC achieves a minimum of  19107.28  with  8  factors
## Sample Size adjusted BIC achieves a minimum of  19624.99  with  8  factors
##
## Statistics by number of factors
##   vss1 vss2   map dof chisq prob sqresid fit RMSEA   BIC SABIC complex eChisq
## 1 1.00 0.00 0.015 324 24103    0    0.61   1  0.26 21847 22876     1.0    162
## 2 0.73 1.00 0.016 298 23299    0    0.55   1  0.27 21224 22171     1.9    136
## 3 0.68 0.91 0.017 273 22223    0    0.35   1  0.28 20322 21189     2.3     60
## 4 0.69 0.88 0.018 249 21607    0    0.29   1  0.28 19873 20664     2.7     40
## 5 0.72 0.91 0.023 226 21269    0    0.28   1  0.30 19696 20414     2.4     36
## 6 0.65 0.89 0.025 204 20859    0    0.27   1  0.31 19438 20086     2.6     37
## 7 0.63 0.88 0.027 183 20461    0    0.24   1  0.32 19187 19768     2.8     30
## 8 0.62 0.87 0.032 163 20242    0    0.22   1  0.34 19107 19625     2.9     25
##     SRMR  eCRMS  eBIC
## 1 0.0148 0.0154 -2094
## 2 0.0136 0.0147 -1939
## 3 0.0090 0.0102 -1840
## 4 0.0073 0.0087 -1694
## 5 0.0070 0.0087 -1537
## 6 0.0071 0.0093 -1383
## 7 0.0063 0.0088 -1244
## 8 0.0058 0.0085 -1110
```

Penn Interactive Peer Play Scale (PIPPS) Data

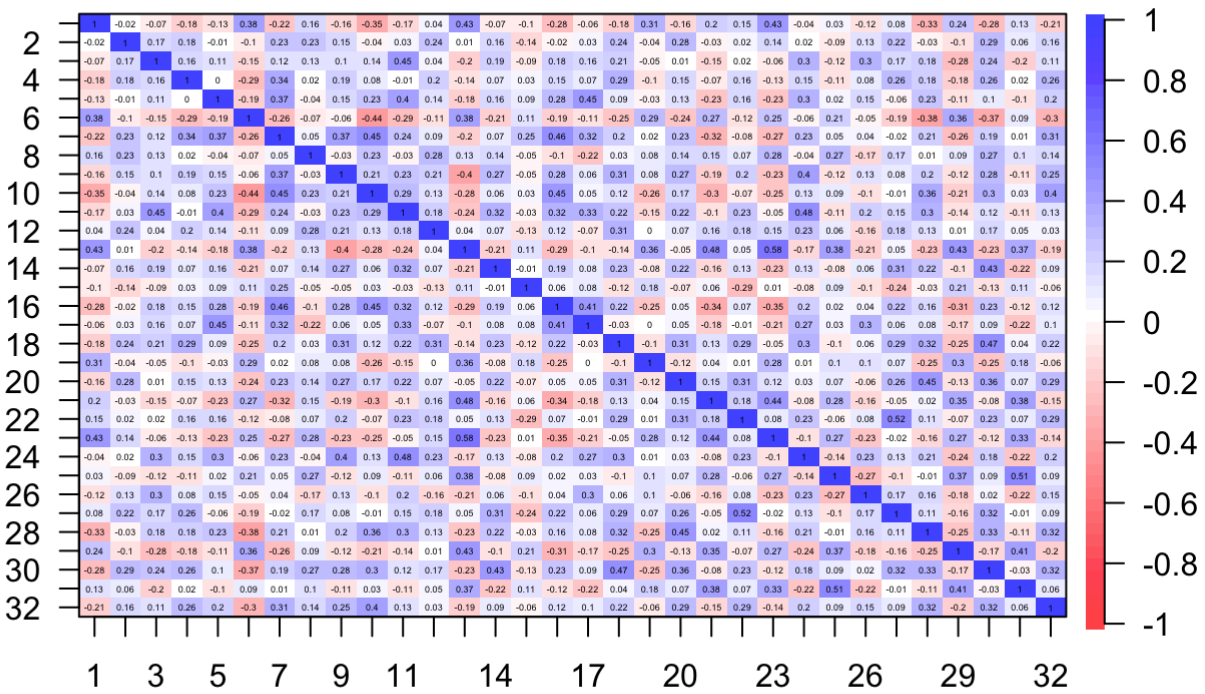
Data preparation

```
PIPPS_wide_with_ids <- PIPPS %>%
  pivot_wider(id_cols=Blind_ID, names_from = "item_num", values_from="value") %>%
  drop_na()

PIPPS_wide <- PIPPS_wide_with_ids %>%
  dplyr::select(-Blind_ID)
```

```
## # A tibble: 88 × 32
##       `1`   `2`   `3`   `4`   `5`   `6`   `7`   `8`   `9`  `10`  `11`  `12`  `13`
##     <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1      3     2     1     3     2     4     2     2     2     1     1         3     3
##  2      3     2     2     1     3     3     2     1     2     2   2.5         3     2
##  3      4     3     2     2     3     3     3     3     1     2     2         2     3
##  4      4     3     2     2     1     4     2     3     2     2     1         3     3
##  5      3     2     1     2     3     3     3     3     1     3     1         3     3
##  6      3     2     2     2     2     3     3     2     2     3     2         2     3
##  7      3     2     2     2     2     3     2     3     2     2     2         3     2
##  8      3     3     2     3     1     3     2     3     1     2     1         3     2
##  9      2     2     2     2     1     3     2     3     1   2.5     2         2     1
## 10      3     3     1     2     2     3     2     2     1     1     1         2     3
## # … with 78 more rows, and 19 more variables: 14 <dbl>, 15 <dbl>, 16 <dbl>,
## #   17 <dbl>, 18 <dbl>, 19 <dbl>, 20 <dbl>, 21 <dbl>, 22 <dbl>, 23 <dbl>,
## #   24 <dbl>, 25 <dbl>, 26 <dbl>, 27 <dbl>, 28 <dbl>, 29 <dbl>, 30 <dbl>,
## #   31 <dbl>, 32 <dbl>
```

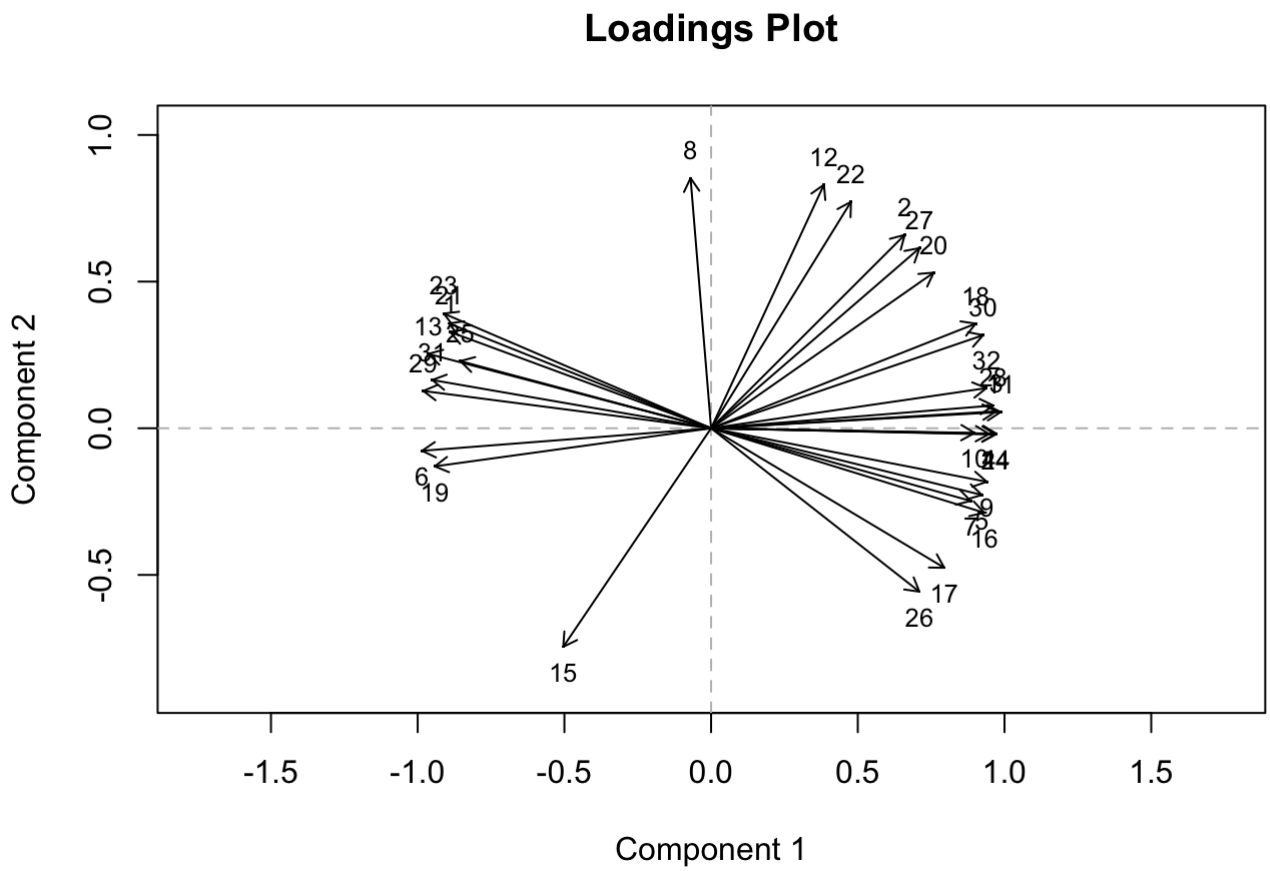
1. Correlation Matrix

```
PIPPS_cor_matrix <- cor(PIPPS_wide)
corPlot(PIPPS_cor_matrix)
```



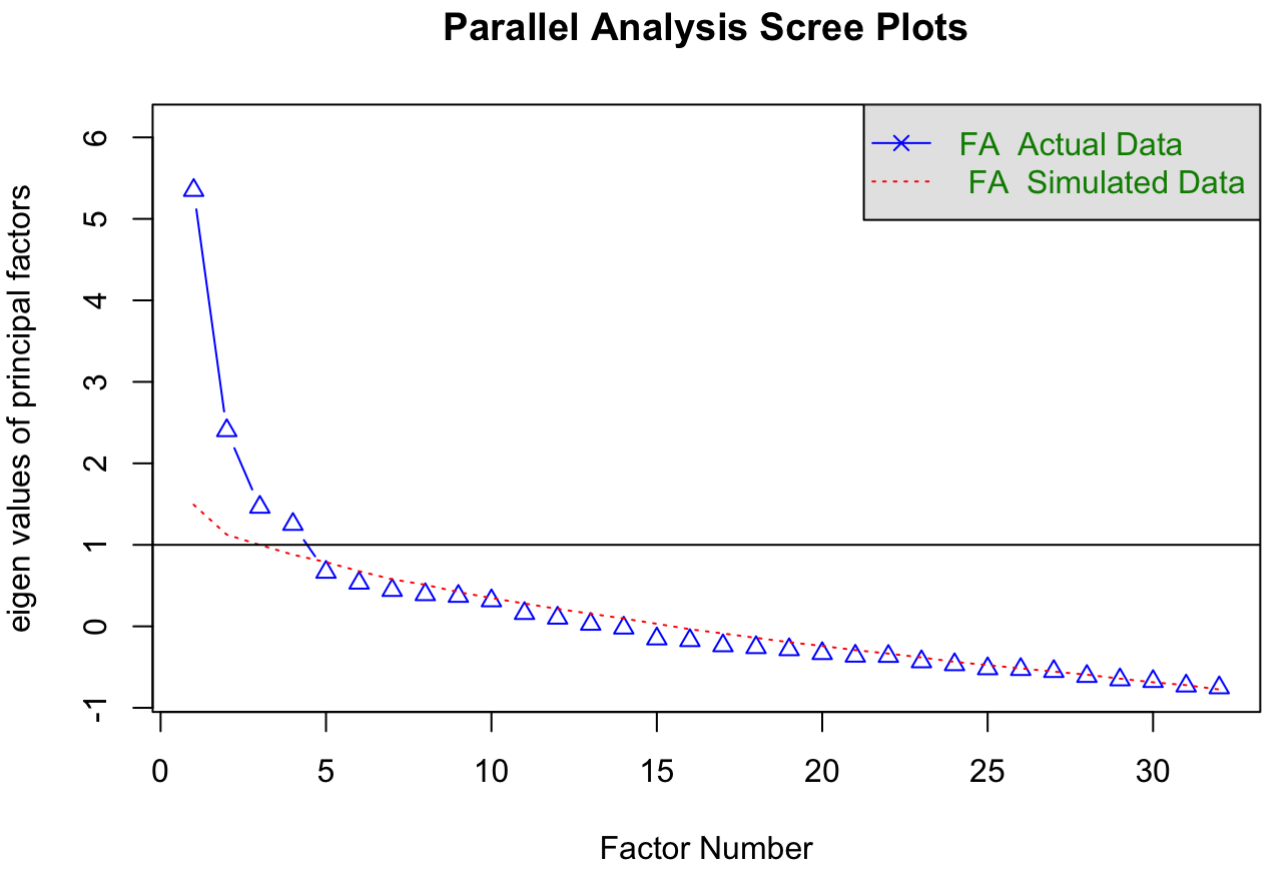
2. Principal Component Analysis

```
pc_PIPPS <- princals(PIPPS_cor_matrix)
plot(pc_PIPPS)
```



3. Parallel Analysis

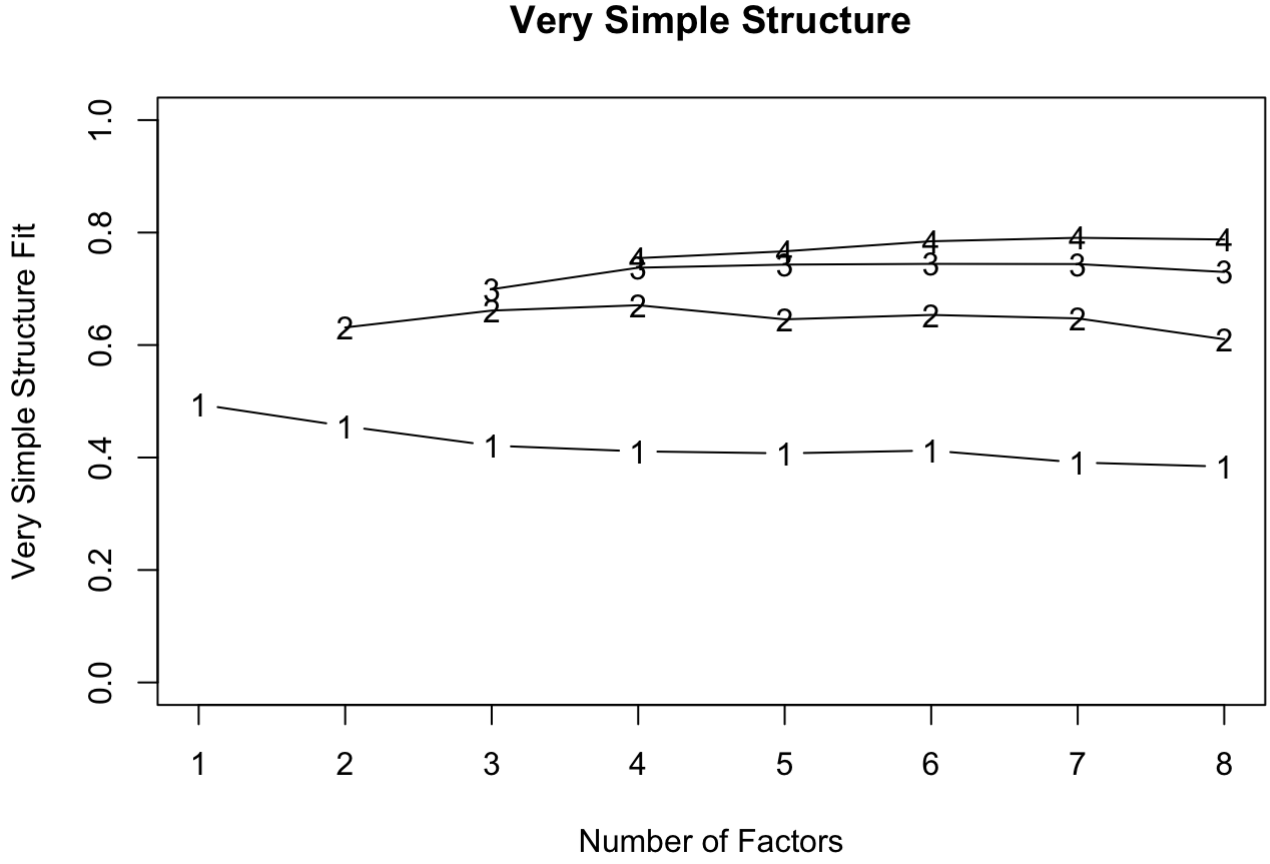
```
fa.parallel(PIPPS_cor_matrix, fa="fa", cor="poly", fm = "ml", n.obs = 88)
```




```
## Parallel analysis suggests that the number of factors =  4  and the number of comp
onents =  NA
```

4. Very Simple Structure

```
vss(PIPPS_cor_matrix, cor="poly", n.obs = 88, fm = "ml")
```


Very Simple Structure
Very Simple Structure Fit
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
7
8
Number of Factors

```
##
## Very Simple Structure
## Call: vss(x = PIPPS_cor_matrix, fm = "ml", n.obs = 88, cor = "poly")
## VSS complexity 1 achieves a maximimum of 0.49  with  1  factors
## VSS complexity 2 achieves a maximimum of 0.67  with  4  factors
##
## The Velicer MAP achieves a minimum of 0.02  with  4  factors
## BIC achieves a minimum of  -1318.76  with  2  factors
## Sample Size adjusted BIC achieves a minimum of  -69.76  with  8  factors
##
## Statistics by number of factors
##   vss1 vss2   map dof chisq    prob sqresid  fit RMSEA   BIC  SABIC complex
## 1 0.49 0.00 0.024 464   764 4.7e-17      38 0.49 0.085 -1313 151.00     1.0
## 2 0.46 0.63 0.020 433   620 8.6e-09      27 0.63 0.069 -1319  47.61     1.4
## 3 0.42 0.66 0.019 403   532 1.6e-05      22 0.70 0.059 -1272  -0.33     1.7
## 4 0.41 0.67 0.019 374   459 1.7e-03      18 0.75 0.050 -1215 -35.15     1.9
## 5 0.41 0.65 0.020 346   406 1.4e-02      17 0.77 0.043 -1143 -51.34     2.0
## 6 0.41 0.65 0.021 319   367 3.3e-02      15 0.80 0.040 -1061 -54.67     2.1
## 7 0.39 0.65 0.023 293   328 7.7e-02      13 0.82 0.035  -984 -59.13     2.3
## 8 0.38 0.61 0.025 268   284 2.3e-01      12 0.84 0.024  -915 -69.76     2.4
##   eChisq  SRMR eCRMS  eBIC
## 1   1330 0.123 0.128  -747
## 2    790 0.095 0.102 -1149
## 3    564 0.080 0.089 -1240
## 4    393 0.067 0.077 -1282
## 5    355 0.064 0.076 -1194
## 6    296 0.058 0.073 -1132
## 7    246 0.053 0.069 -1066
## 8    211 0.049 0.067  -989
```

In conclusion of the dimensionality checks, it seems that there are 4 factors. Therefore, an exploratory factor analysis will be conducted with four factors.

5. Exploratory Factor Analysis

```
motFA <- fa(PIPPS_cor_matrix, nfactors = 4, rotate = "oblimin", fm = "ml")
```

```
## Loading required namespace: GPArotation
```

```
print(motFA$loadings, cutoff = 0.2) # factor loadings above the cutoff size 0.2
```

```
##
## Loadings:
##    ML2    ML3    ML4    ML1
## 1          0.307         -0.503
## 2   0.367
## 3   0.252          0.249
## 4   0.316
## 5                  0.606
## 6  -0.314  0.280         -0.404
## 7                  0.525  0.410
## 8   0.302  0.325
## 9   0.267          0.277
## 10                         0.669
## 11  0.257          0.518
## 12  0.372
## 13         0.691         -0.275
## 14  0.352
## 15 -0.347  0.222  0.216
## 16                 0.451  0.285
## 17                 0.654
## 18  0.603
## 19         0.333  0.210 -0.338
## 20  0.512                 0.212
## 21  0.209  0.500 -0.209
## 22  0.597                 -0.260
## 23         0.581         -0.254
## 24  0.269          0.509 -0.202
## 25         0.687          0.288
## 26        -0.322  0.267 -0.284
## 27  0.586
## 28  0.355                 0.310
## 29 -0.205  0.518
## 30  0.579                 0.328
## 31         0.645          0.203
## 32  0.303                 0.324
##
##                  ML2   ML3   ML4   ML1
## SS loadings    3.114 2.970 2.295 2.235
## Proportion Var 0.097 0.093 0.072 0.070
## Cumulative Var 0.097 0.190 0.262 0.332
```

Summary

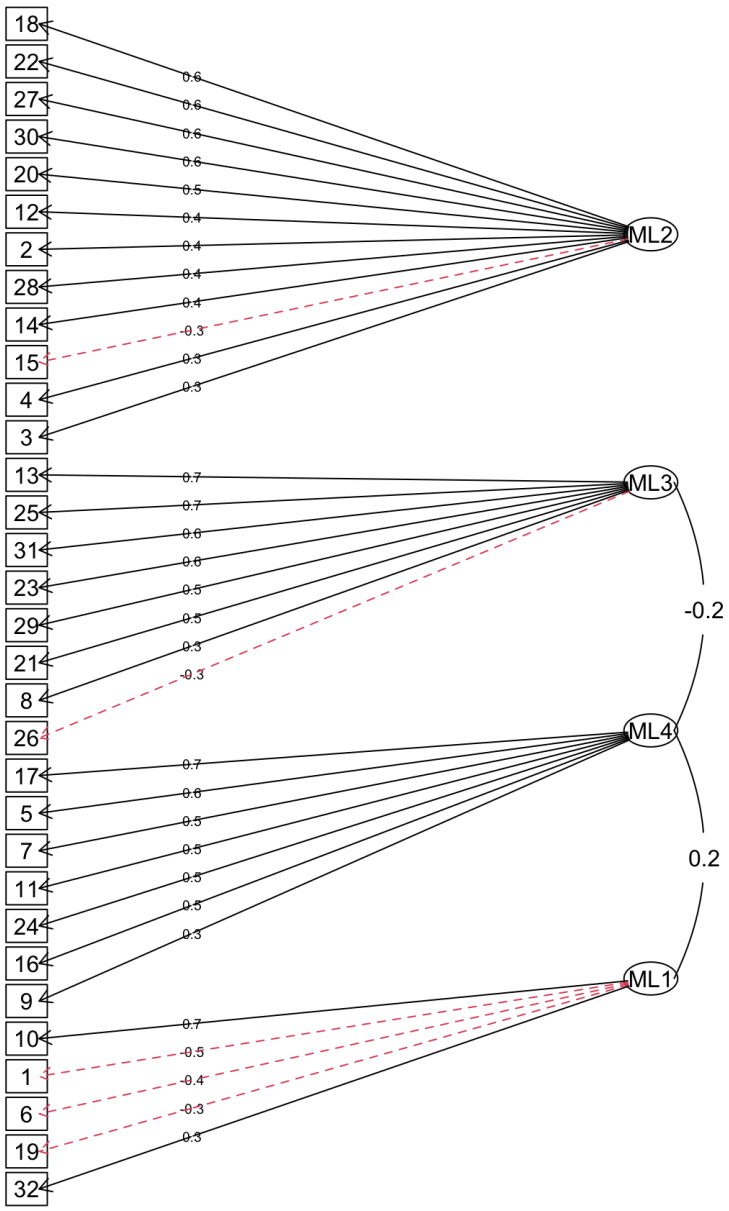
```
summary(motFA, cut=0.2, rotate = 'oblimin')
```

```
##
## Factor analysis with Call: fa(r = PIPPS_cor_matrix, nfactors = 4, rotate = "oblimi
n", fm = "ml")
##
## Test of the hypothesis that 4 factors are sufficient.
## The degrees of freedom for the model is 374  and the objective function was  6.3
##
## The root mean square of the residuals (RMSA) is  0.07
## The df corrected root mean square of the residuals is  0.08
##
##  With factor correlations of
##       ML2   ML3   ML4   ML1
## ML2  1.00 -0.09  0.18  0.13
## ML3 -0.09  1.00 -0.25 -0.15
## ML4  0.18 -0.25  1.00  0.22
## ML1  0.13 -0.15  0.22  1.00
```

Diagram

```
fa.diagram(motFA, cut=.2)
```

Factor Analysis



Comparing the results with the PIPPS instructions

EFA	Item	PIPPS_P	PIPPS_T
ML1	1*	PI	PI
	6*	PI	PI, PDR**
	10	PDR, PDC	PDR, PDC
	19*	PI	PI, PDR**
	32	PDR	PDR
ML2	2	PDR	PDR
	3	PDC	PDR, PDC
	4		PDR
	12	PDR	PDR
	14	PDR	PDR
	15*		PDR**
	18	PDR	PDR
	20	PDR	PDR
	22	PDR	PDR
	27	PDR	PDR
	28	PDC	PDC
	30	PDR	PDR
ML3	8	PDR	PDR
	13	PI	PI
	21	PI	PI, PDC**
	23	PI	PI
	25	PI	PI
	26*		PI*
	29	PI	PI
	31	PI	PI
ML4	5		
	7	PDC	PDC
	9	PDC	PDC
	11	PDC	PDC
	16		PDC
	17	PDC	PDC
	24	PDC	PDC

* Negative correlation with EFA factor

** Reverse scored