Scientific data visualization

Using ggplot2

Sacha Epskamp

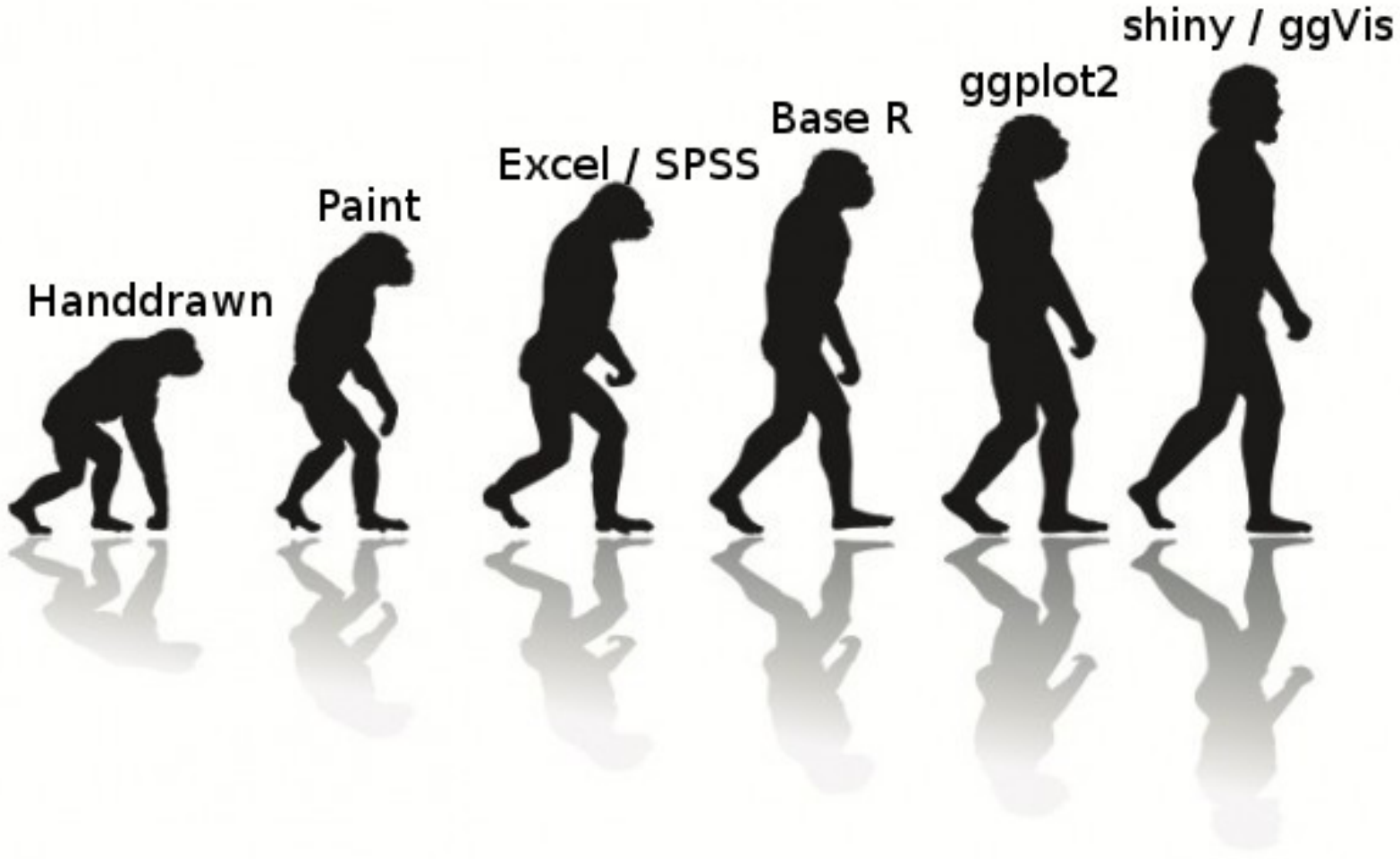
University of Amsterdam
Department of Psychological Methods

11-04-2014

Hadley Wickham

Hadley Wickham

Evolution of data visualization



Scientific data visualization

- Data and analysis results are best communicated through *visualizations*
- The leading software for statistical analyses is the statistical programming language **R**
- The leading R extension for data visualization is **ggplot2**
- This presentation will quickly teach you strong visualization techniques in R

First use of **R**

- We will use the environment **RStudio** for our work in **R**
- **RStudio** has 4 panels:

Console	This is the actual **R** window, you can enter commands here and execute them by pressing enter
Source	This is where we can edit *scripts*. It is where you should always be working. Control-enter sends selected codes to the console
Plots/Help	This is where plots and help pages will be shown
Workspace	Shows which objects you currently have

- Anything following a # symbol is treated as a comment!

R workflow

- `File` → `New File` → `R script`
- Write codes in the `R` script
- Select codes and press `control + enter` to execute them

Import data

```
File <- "http://sachaepskamp.com/files/OPdata.csv"
Data <- read.csv(File)
```

Look at data

```
head(Data)
```

```
##   userID Measurement Gender Age Study      Work Neuroticism
## 1      1           1 female  24   yes part time          low
## 2      1           2 female  24   yes part time          low
## 3      1           3 female  24   yes part time          low
## 4      1           4 female  24   yes part time          low
## 5      1           5 female  24   yes part time          low
## 6      1           6 female  24   yes part time          low
##   Extraversion Openness Conscienciousness Agreeableness
## 1          low     high               high          high
## 2          low     high               high          high
## 3          low     high               high          high
## 4          low     high               high          high
## 5          low     high               high          high
## 6          low     high               high          high
##   Stress
## 1  0.375
## 2  0.875
## 3  1.375
## 4  1.875
## 5  1.875
## 6  0.750
```

Look at data

```
names(Data)
```

```
##  [1] "userID"           "Measurement"
##  [3] "Gender"           "Age"
##  [5] "Study"            "Work"
##  [7] "Neuroticism"      "Extraversion"
##  [9] "Openness"         "Conscienciousness"
## [11] "Agreeableness"    "Stress"
```

Look at data

```
str(Data)


## 'data.frame': 750 obs. of  12 variables:
##  $ userID           : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Measurement      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Gender           : Factor w/ 2 levels "female","male": 1 1 1 1 1
##  $ Age              : int  24 24 24 24 24 24 24 24 24 24 ...
##  $ Study            : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2
##  $ Work             : Factor w/ 3 levels "full time","none",..: 3 3
##  $ Neuroticism      : Factor w/ 2 levels "high","low": 2 2 2 2 2 2
##  $ Extraversion     : Factor w/ 2 levels "high","low": 2 2 2 2 2 2
##  $ Openness         : Factor w/ 2 levels "high","low": 1 1 1 1 1 1
##  $ Conscienciousness: Factor w/ 2 levels "high","low": 1 1 1 1 1 1
##  $ Agreeableness    : Factor w/ 2 levels "high","low": 1 1 1 1 1 1
##  $ Stress           : num  0.375 0.875 1.375 1.875 1.875 ...
```

Look at data

```
View(Data)
```

ggplot2

- **ggplot2** (Wickham, 2009) is an implementation of the Grammer of Graphics (Wilkinson, Wills, Rope, Norton, & Dubbs, 2006)
- Very different from base R plotting but also very flexible and powerfull
- Uses data frames as input
  - Data must be in **long format**
  - This means that each row is an observation and each column a variable
  - Use `reshape2` to get data in long format
  - Also check out `dplyr` (`http://sachaepskamp.com/files/dplyrTutorial.html`)

Basics of a plot

- A plot is a 2D repressentation of data, in which variables can be visualized by, e.g.,:
  - Horizontal placing
  - Vertical placing
  - Color
  - Different Lines
  - Line type
  - Size
  - Shape
  - ...
- These are called **aesthetics**
- In `ggplot2` we first set aesthetic mapping of our data using `aes()` inside `ggplot()`
  - Which variables will be mapped to which aesthetics?

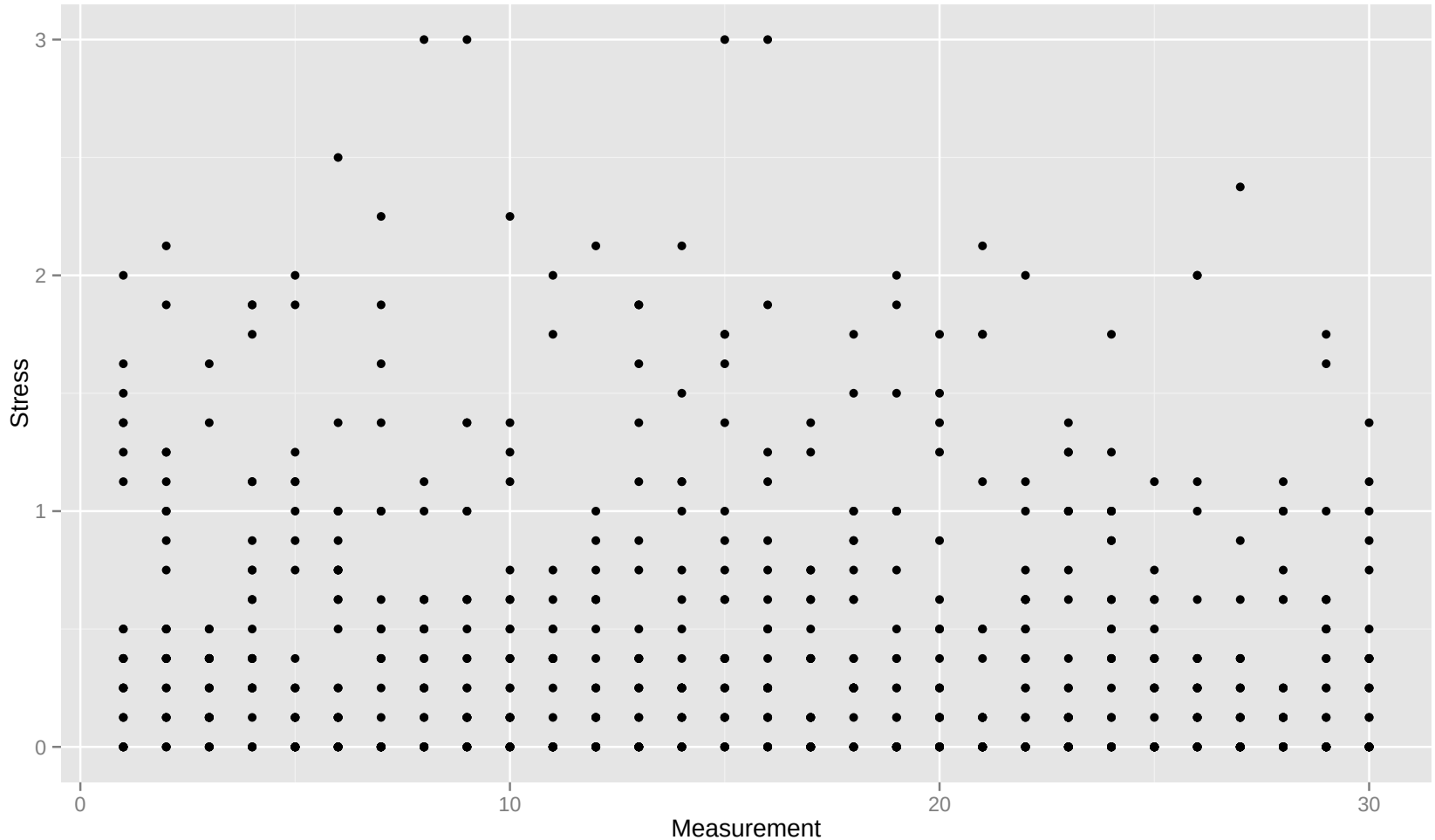
```
install.packages("ggplot2")
library("ggplot2")
ggplot(Data, aes(x = Measurement, y = Stress))
```

```
## Error:  No layers in plot
```

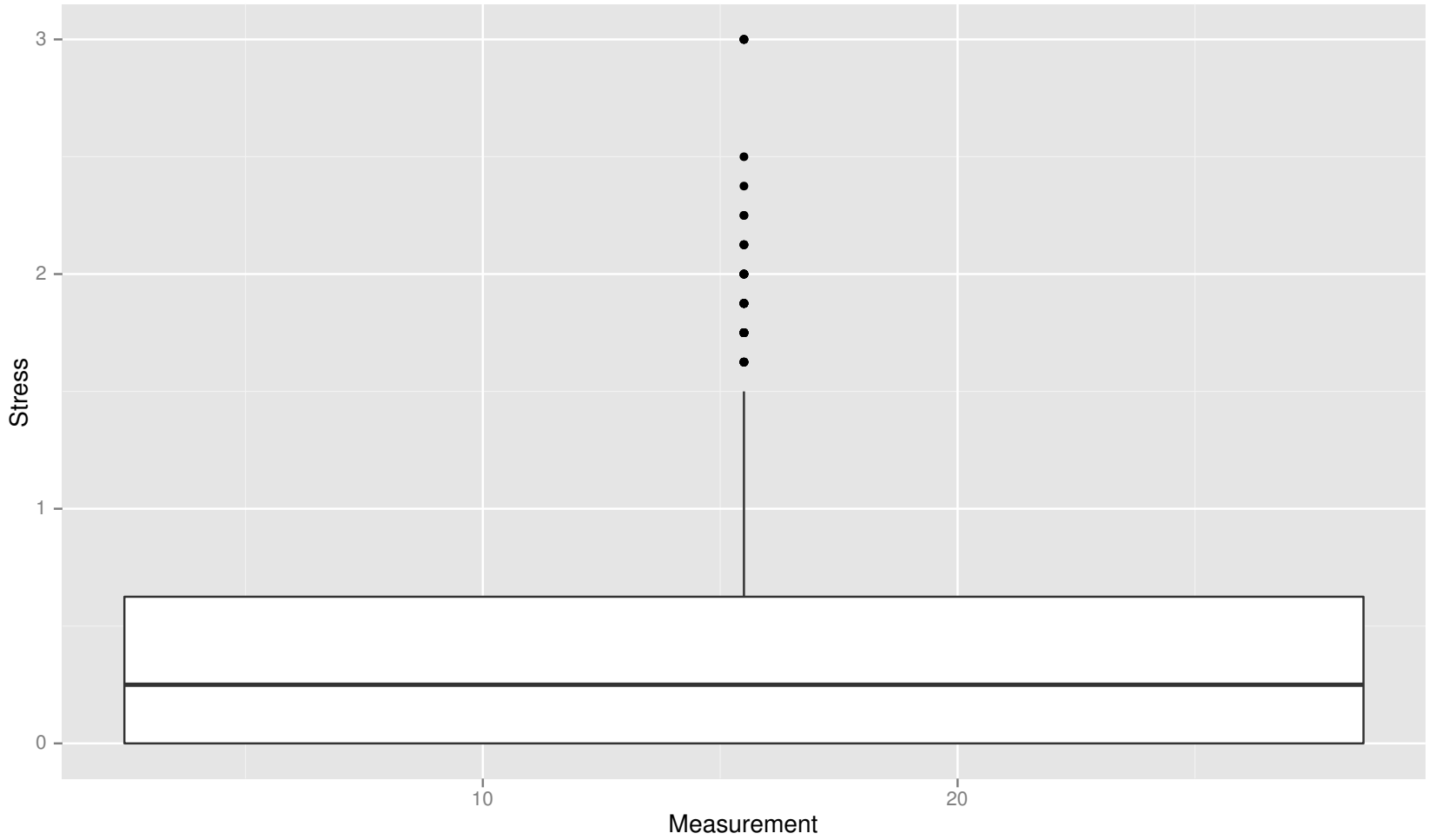
Geometrics

- Next, we define how these aesthetics are used and *what* we are plotting:
  - Lines
  - Points
  - Boxplots
  - Curves
  - ...
- These are called geometrics (geoms)
- We can add these to the plot using +

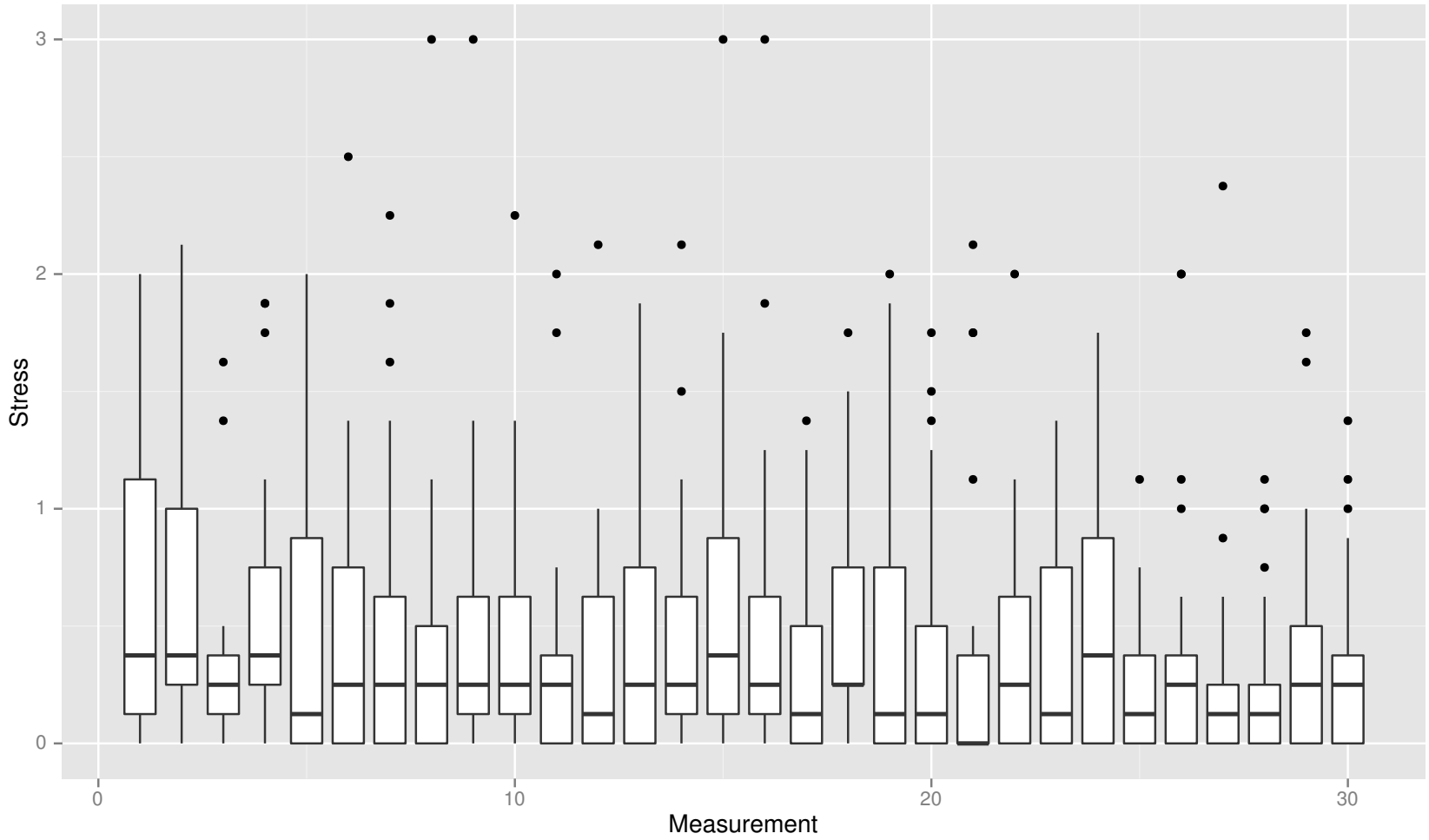
```
ggplot(Data, aes(x = Measurement, y = Stress)) +
  geom_point()
```



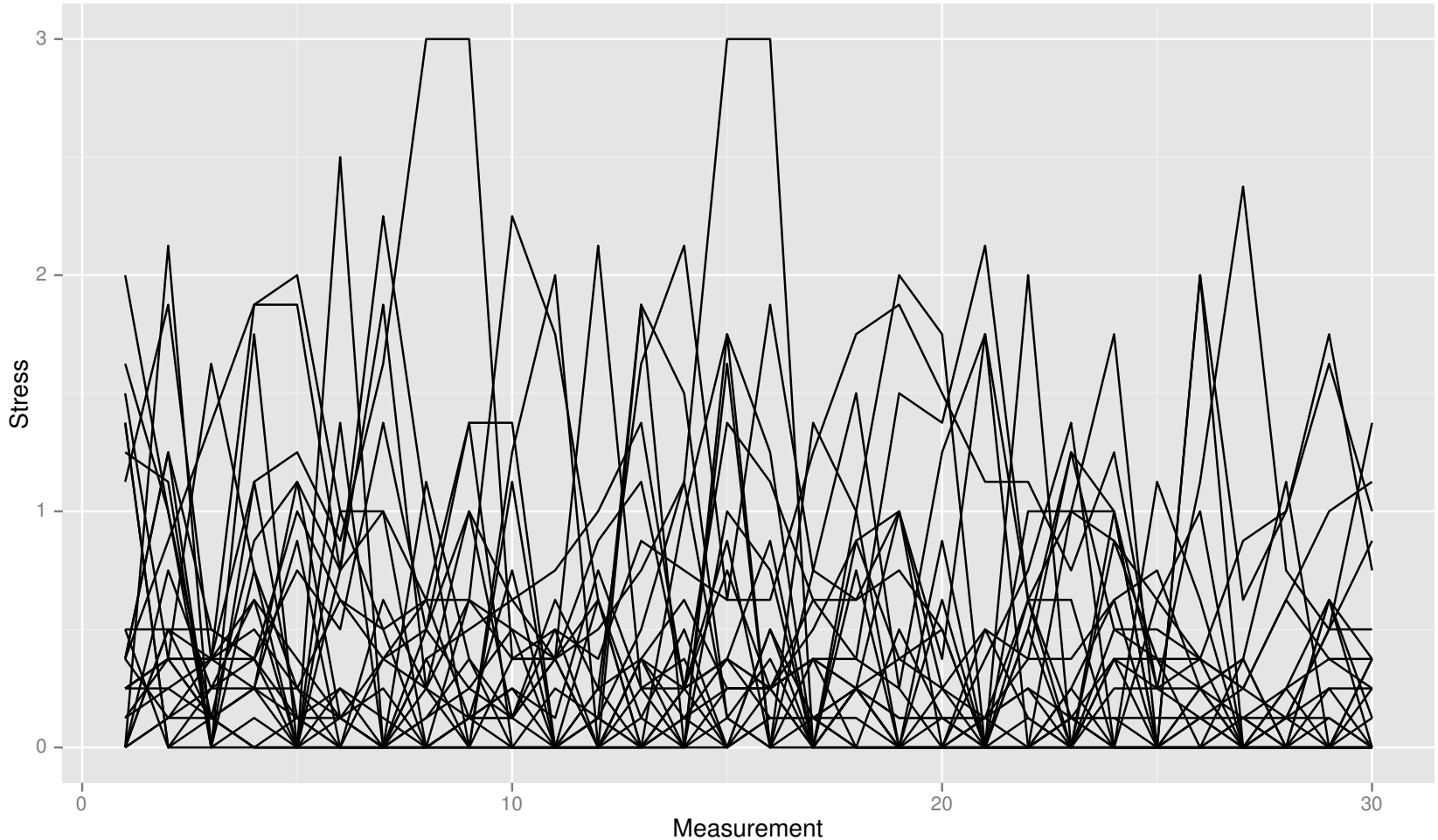
```
ggplot(Data, aes(x = Measurement, y = Stress)) +
  geom_boxplot()
```



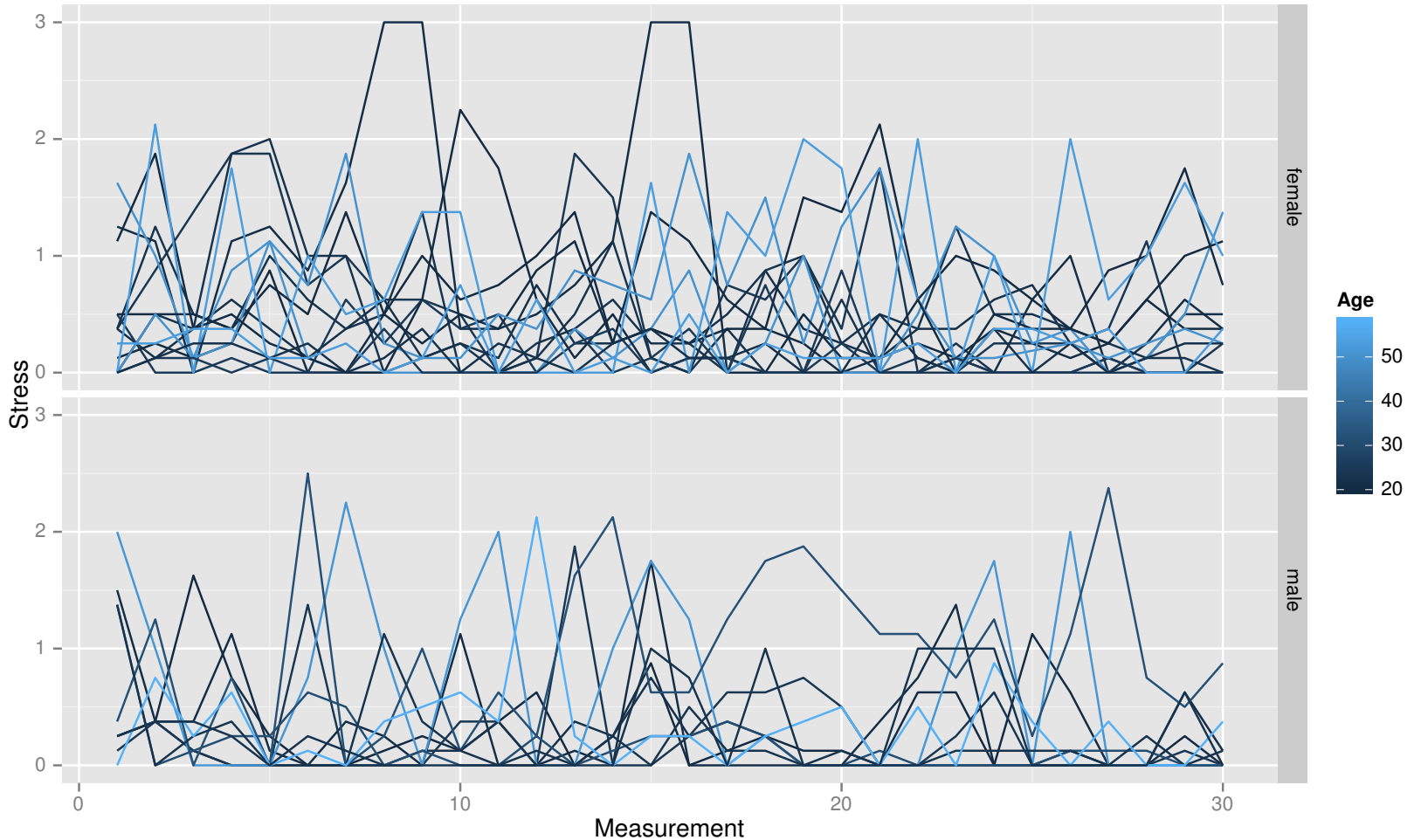
```
ggplot(Data, aes(x = Measurement, y = Stress, group = Measurement)) +
  geom_boxplot()
```



```
ggplot(Data,
 aes(x = Measurement, y = Stress, group = userID)
) + geom_line()
```



```
ggplot(Data,
 aes(x = Measurement, y = Stress, group = userID,
colour = Age)) + geom_line() +
 facet_grid(Gender ~ .)
```

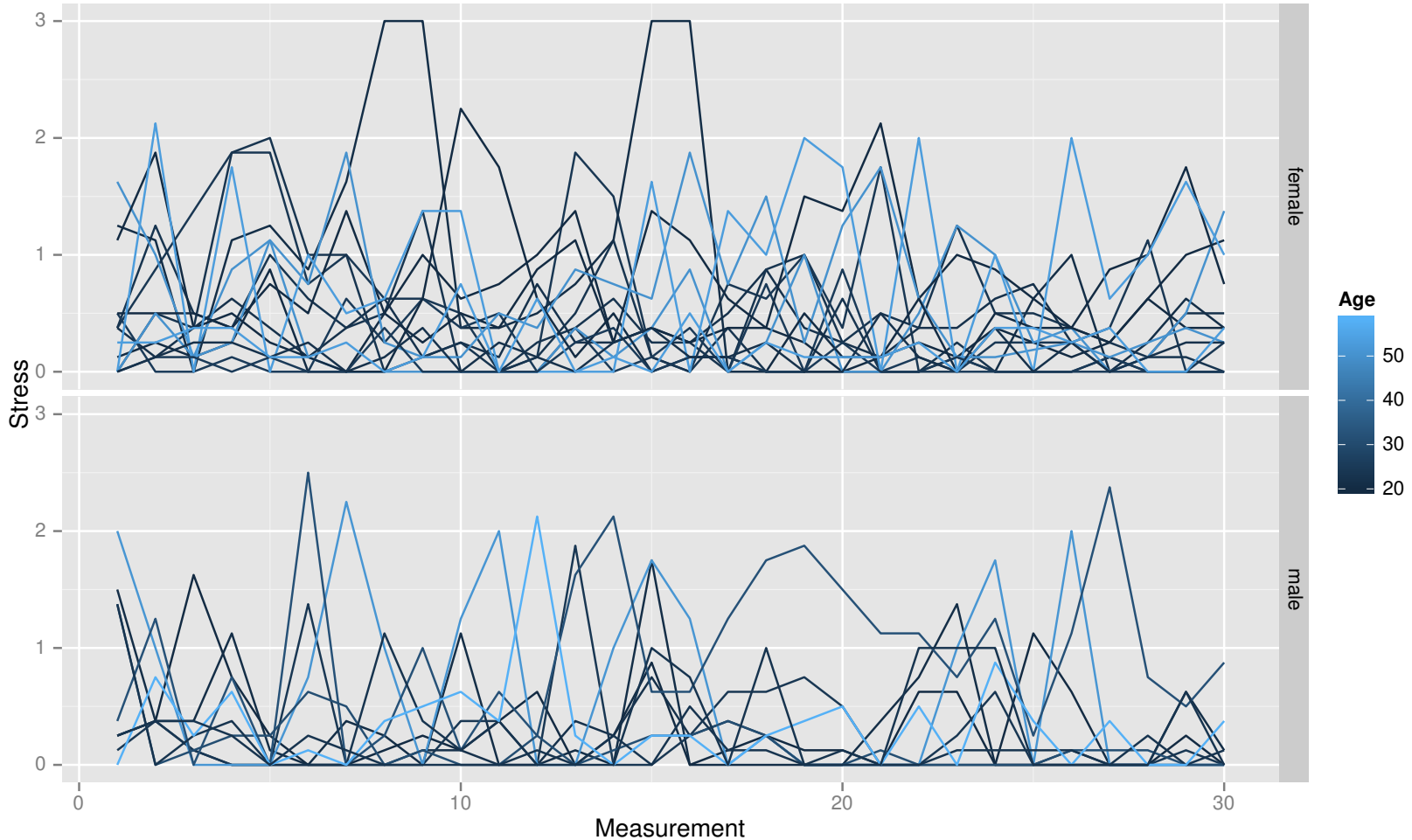


Store elements in an object:

```
g <- ggplot(Data,
 aes(x = Measurement, y = Stress, group = userID,
colour = Age))
g <- g + geom_line()
g <- g +facet_grid(Gender ~ .)
```

Print the object to plot:

```
print(g)
```

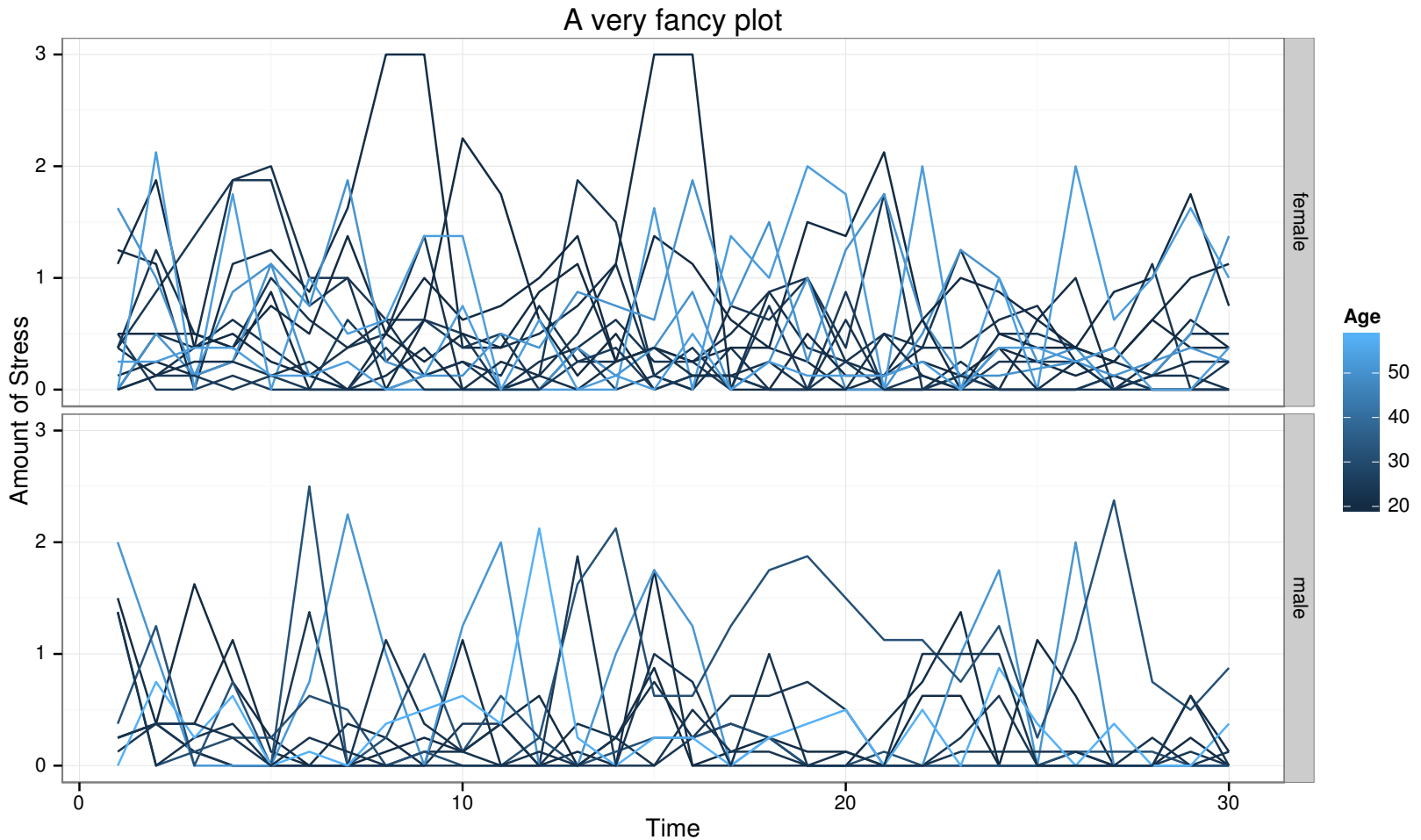


- Many more graphical options can be *added* to ggplot calls

xlab	Label of *x*-axis
ylab	Label of *y*-axis
ggtitle	Title of plot
theme	Many, many graphical settings
theme_bw()	A default black and white theme

- Use Google!

```
g + xlab("Time") + ylab("Amount of Stress") +
  ggtitle("A very fancy plot") + theme_bw()
```

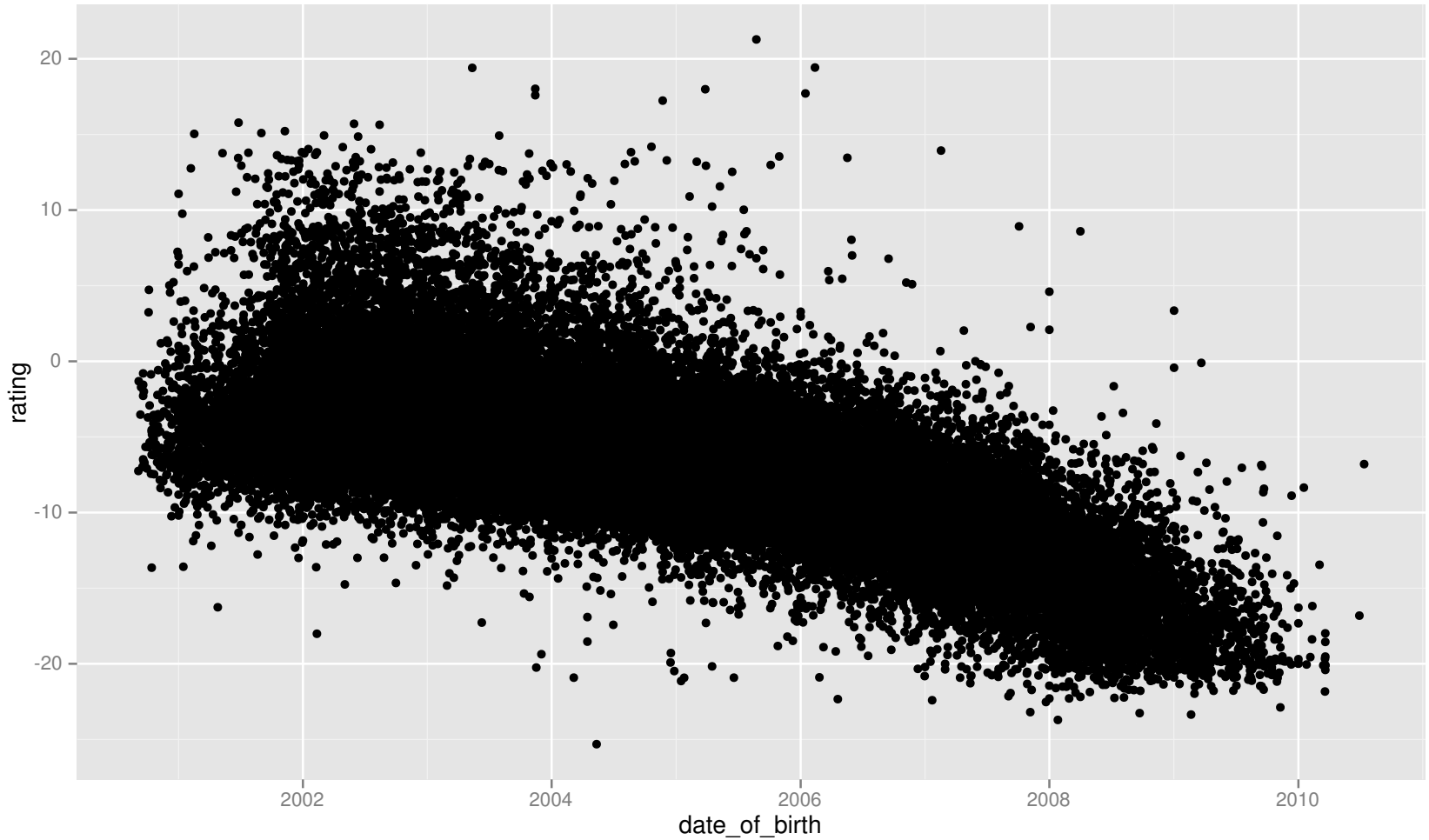



3 + 1
5
6
1
3
2
4
?
0

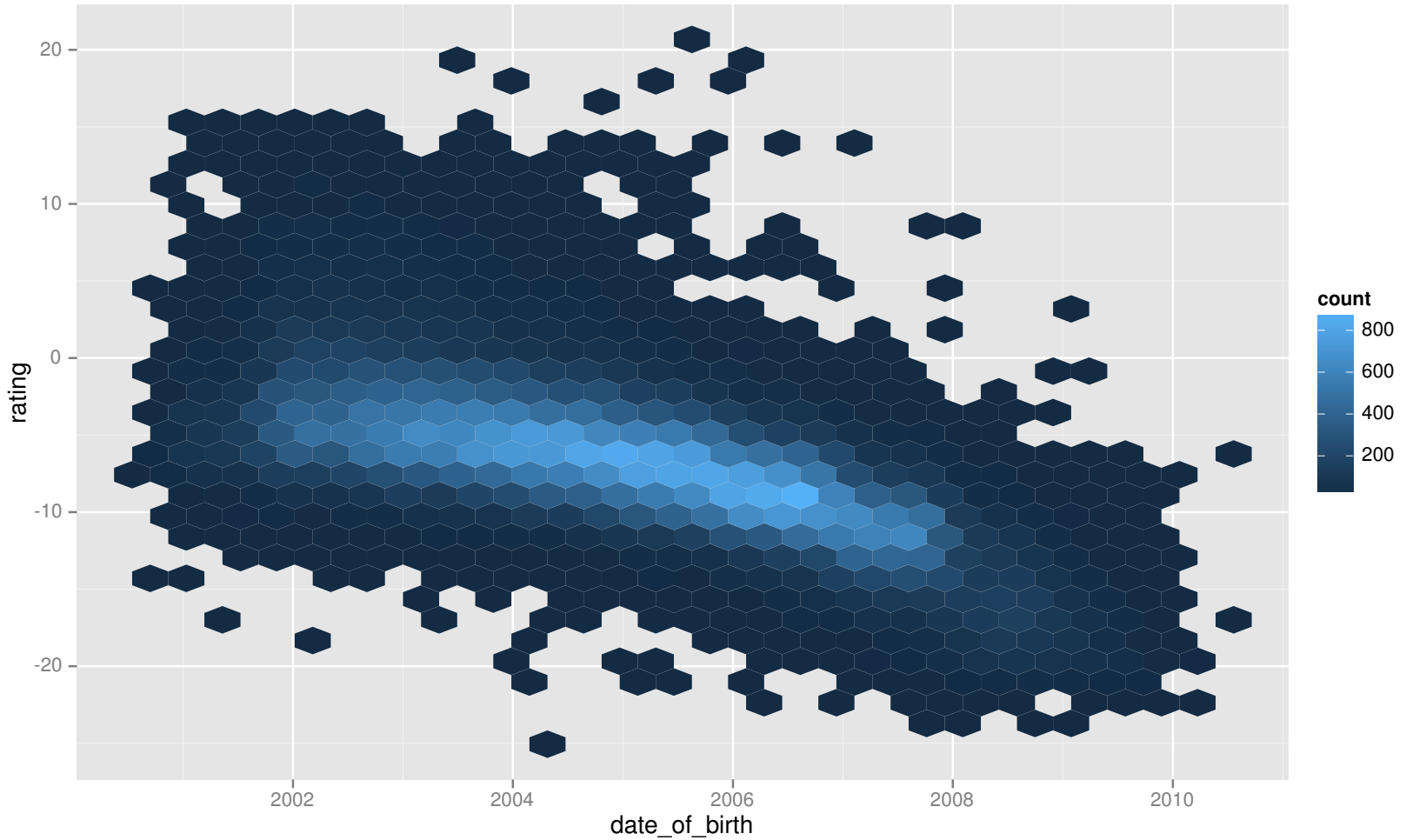
```r
str(sumData)
```

```
## 'data.frame': 66095 obs. of  5 variables:
##  $ user_id      : num  1456 1713 1837 1845 21167 ...
##  $ rating       : num  0.482 -5.225 -6.639 -0.417 -3.008 ...
##  $ date_of_birth: Date, format: "2001-06-03" ...
##  $ grade        : num  8 8 8 8 7 8 7 8 8 8 ...
##  $ gender       : chr  "f" "m" "m" "f" ...
```

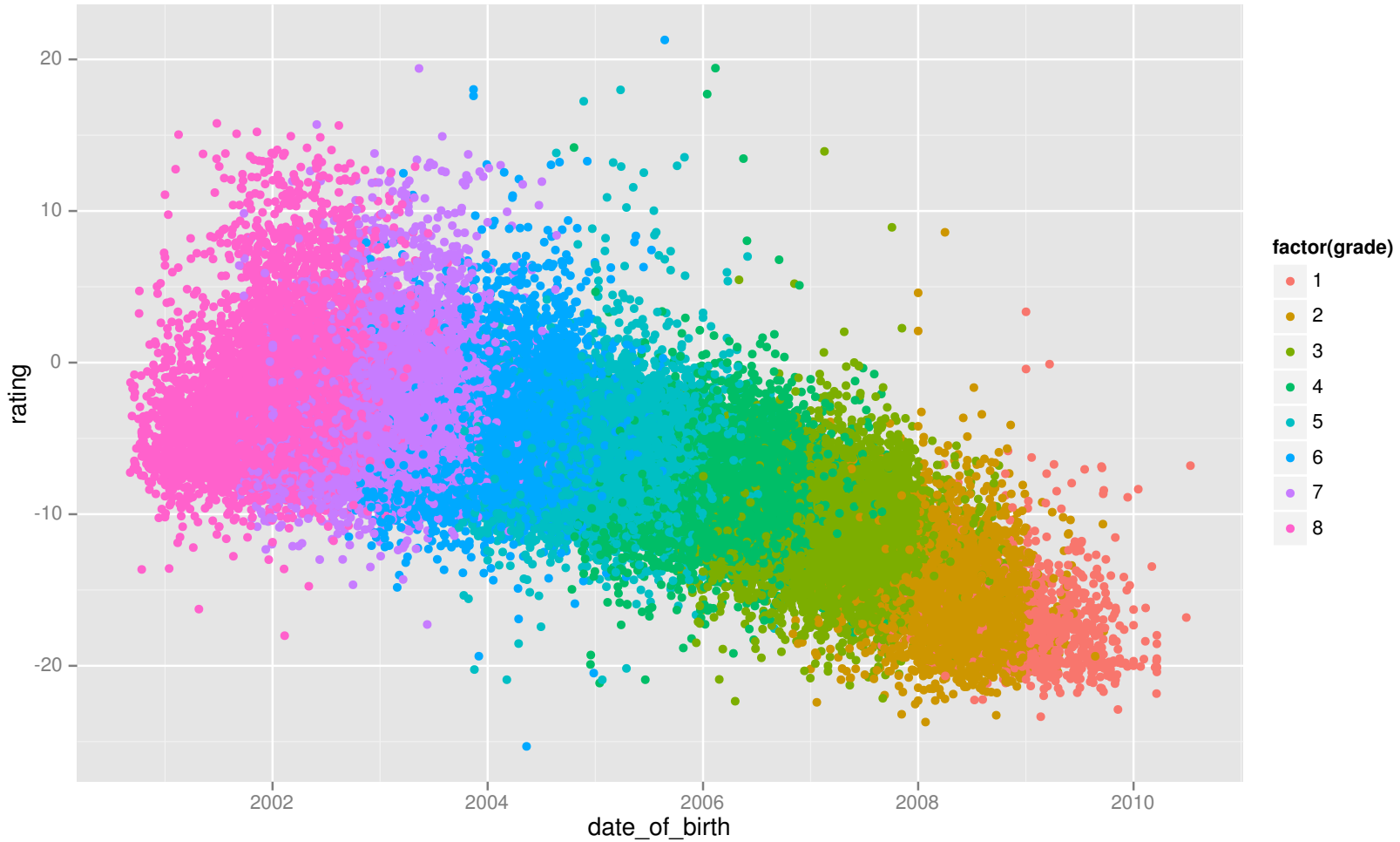
```
ggplot(sumData, aes(x = date_of_birth, y = rating)) +
  geom_point()
```



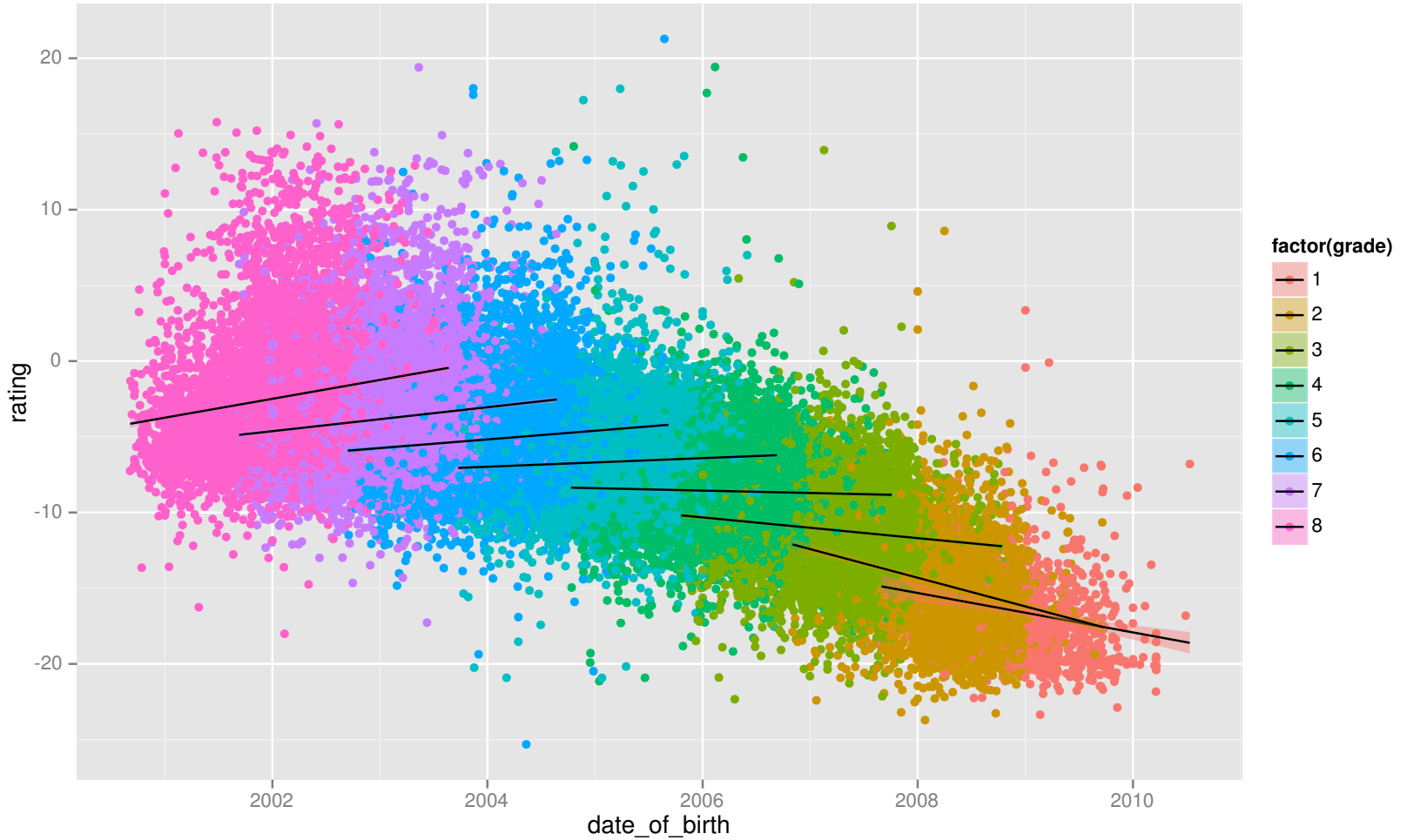
```
ggplot(sumData, aes(x = date_of_birth, y = rating)) +
  stat_binhex()
```



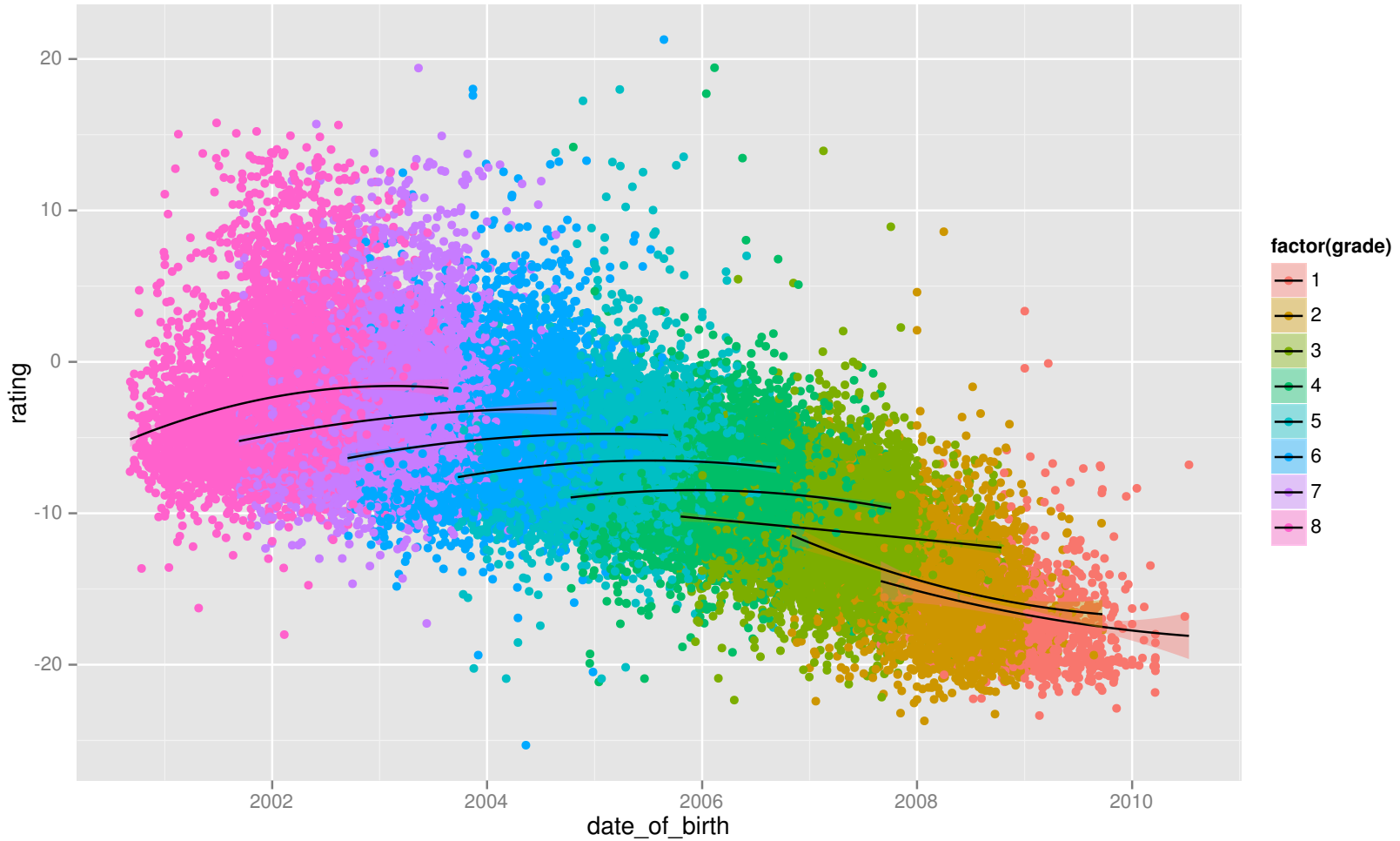
```
ggplot(sumData, aes(x = date_of_birth, y = rating,
    colour = factor(grade))) + geom_point()
```



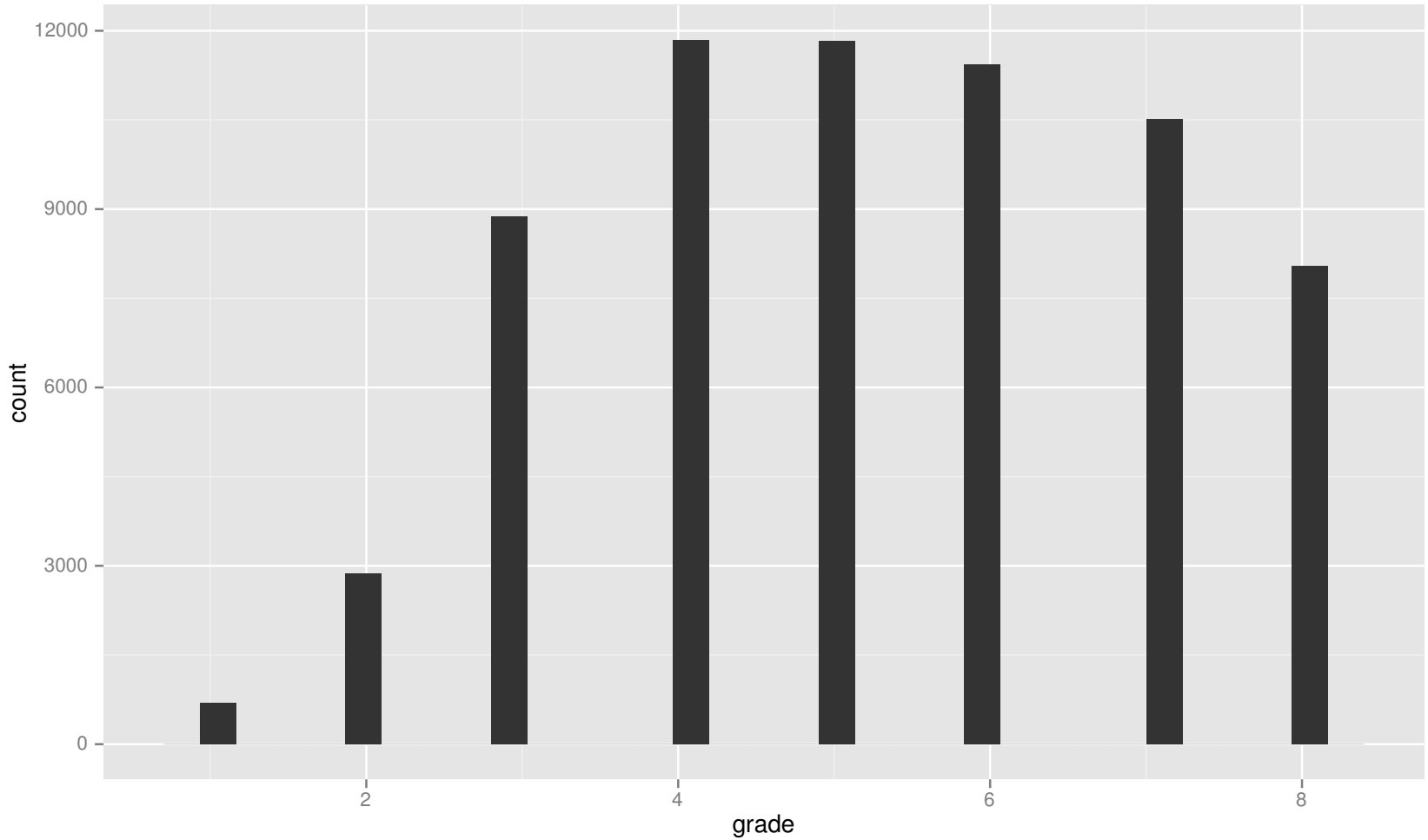
```
ggplot(sumData, aes(x = date_of_birth, y = rating,
  colour = factor(grade), fill = factor(grade))) +
  geom_point() + geom_smooth(col = "black", method = "lm")
```



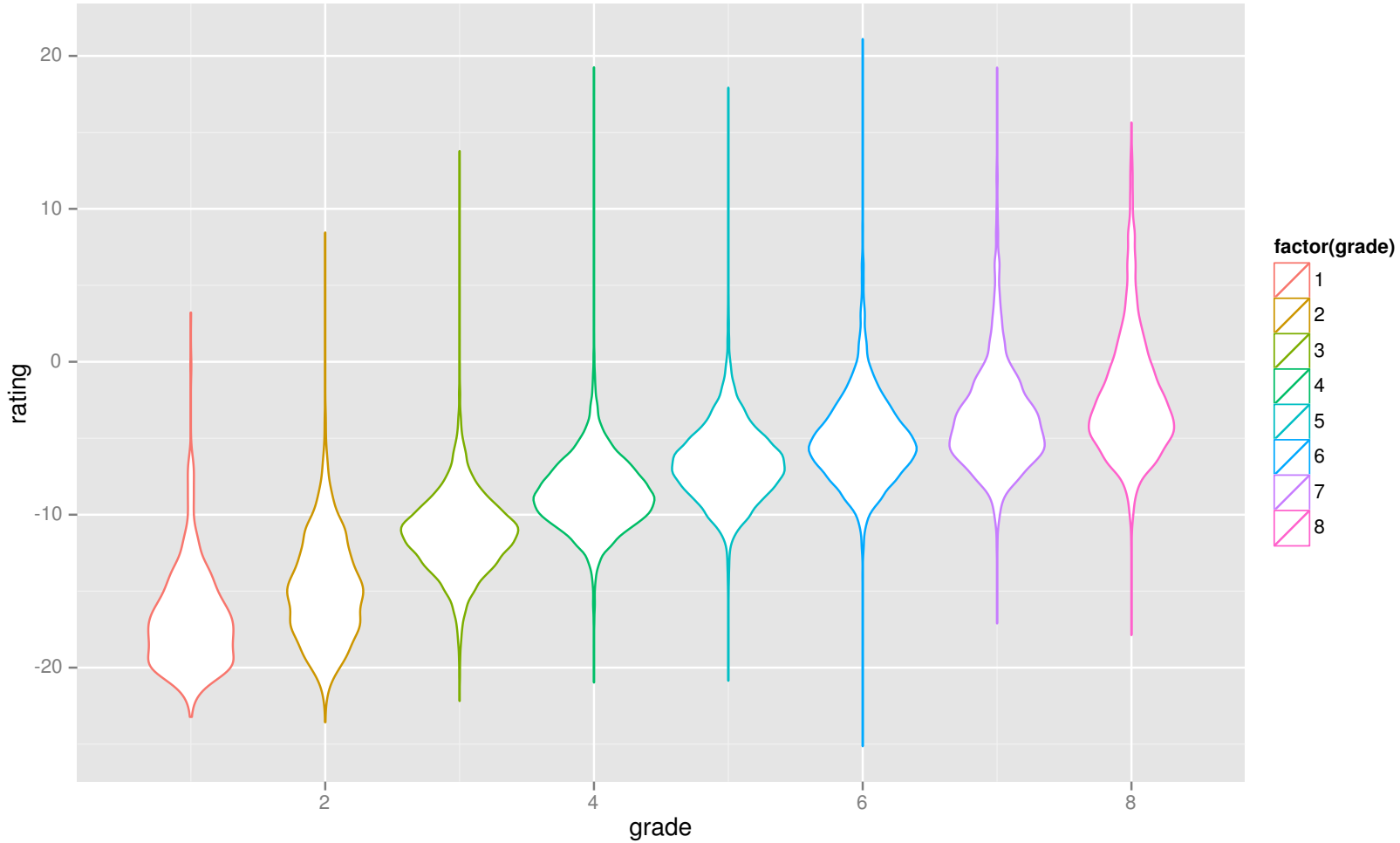
```
ggplot(sumData, aes(x = date_of_birth, y = rating,
       colour = factor(grade), fill = factor(grade))) +
       geom_point() + geom_smooth(col = "black", method = "lm",
       formula = y ~ poly(x, 2))
```

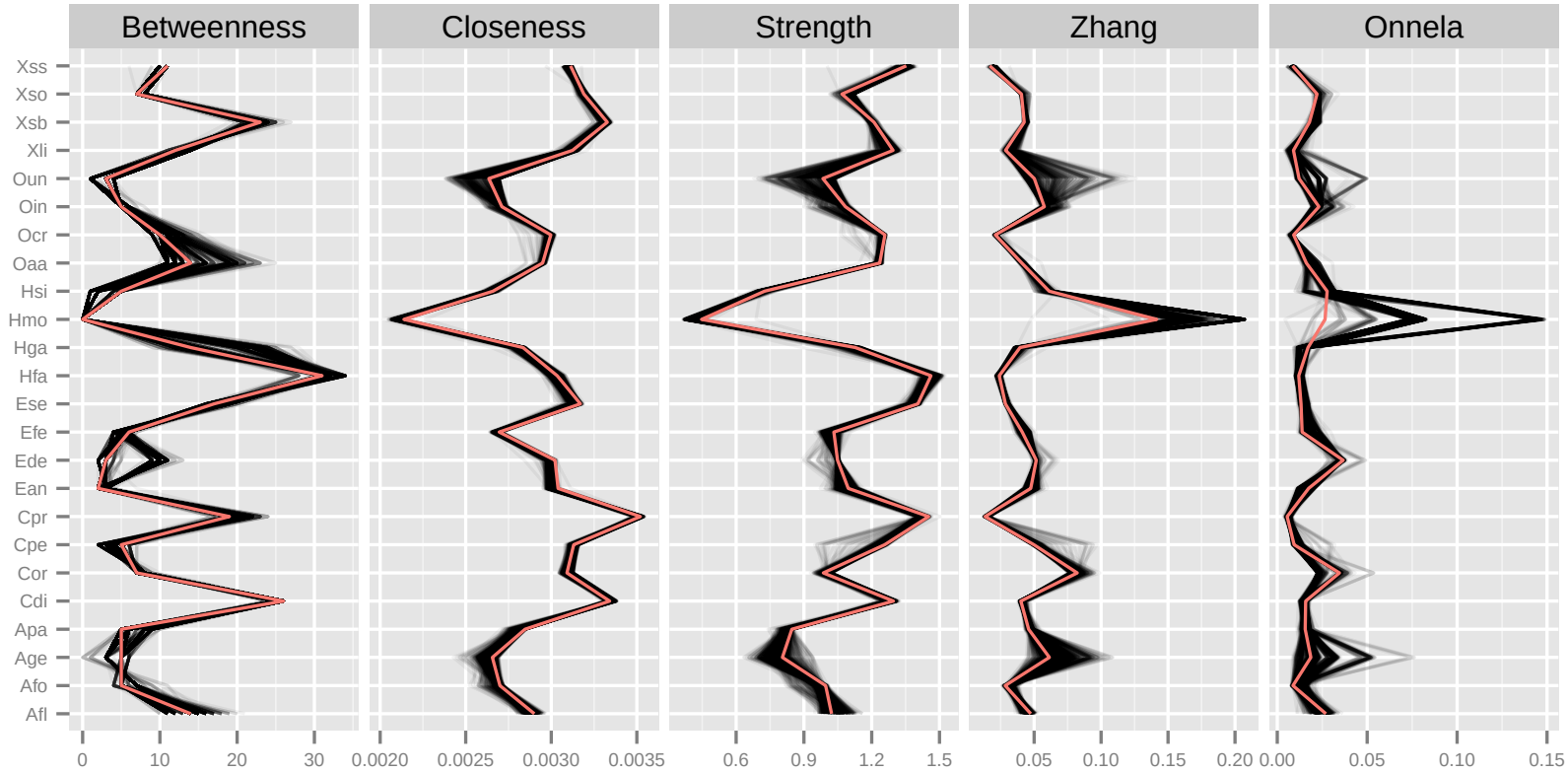


```
ggplot(sumData, aes(x = grade)) + geom_histogram()
```

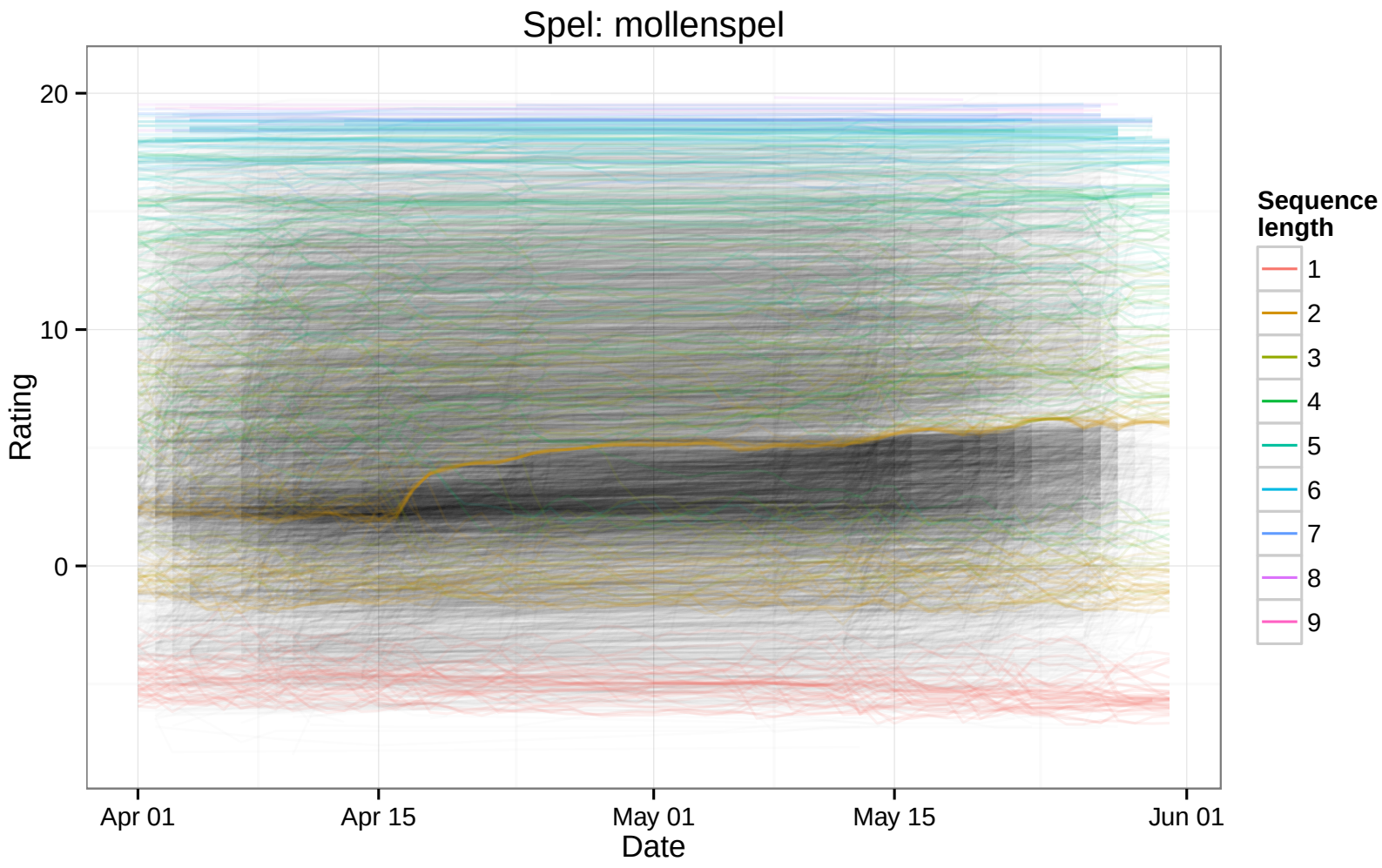


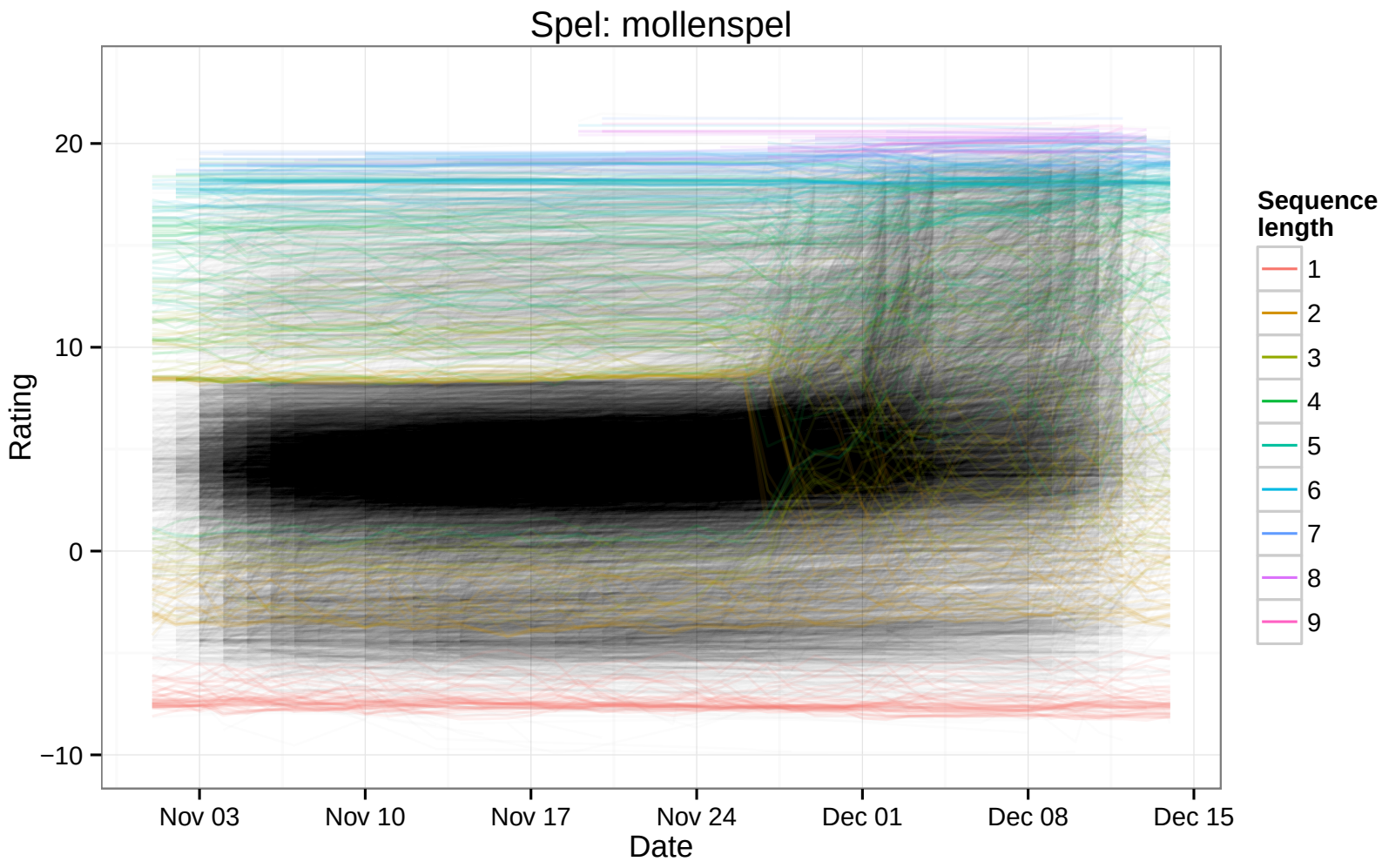
```
ggplot(sumData,
  aes(x = grade, y = rating, colour = factor(grade))
) + geom_violin()
```

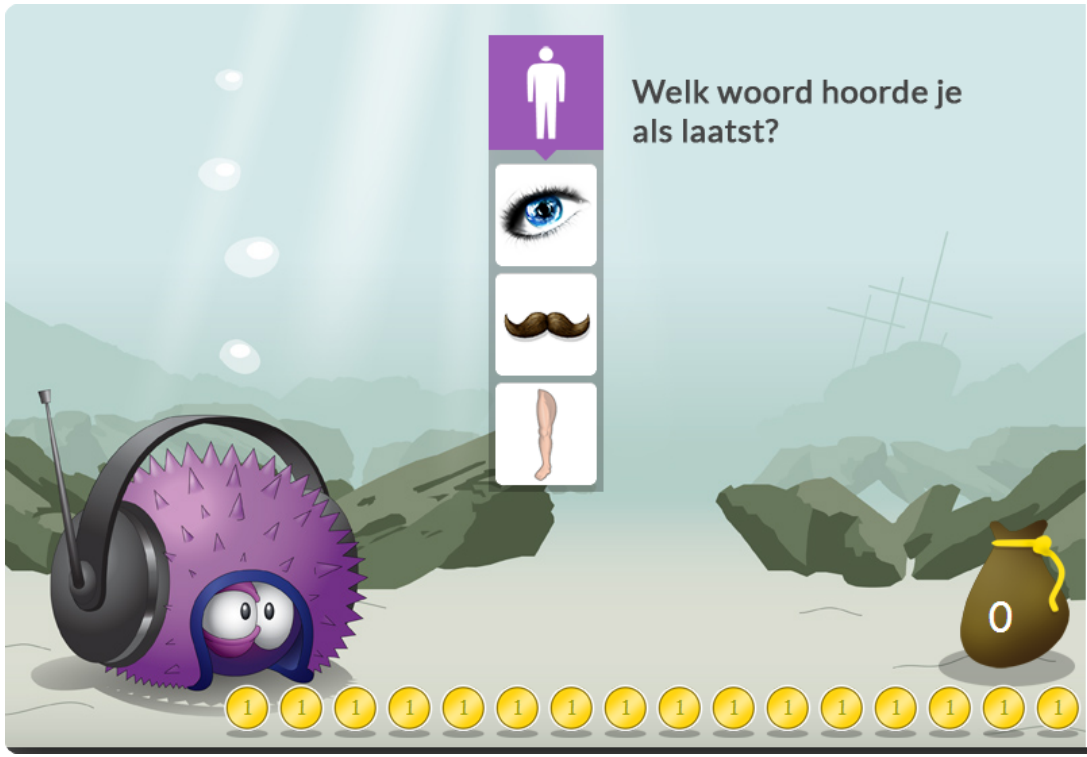



Betweenness
Closeness
Strength
Zhang
Onnela
Xss
Xso
Xsb
Xli
Oun
Oin
Ocr
Oaa
Hsi
Hmo
Hga
Hfa
Ese
Efe
Ede
Ean
Cpr
Cpe
Cor
Cdi
Apa
Age
Afo
Afl
0
10
20
30
0.0020
0.0025
0.0030
0.0035
0.6
0.9
1.2
1.5
0.05
0.10
0.15
0.20
0.00
0.05
0.10
0.15


Onthoud de volgorde van de mollen
0

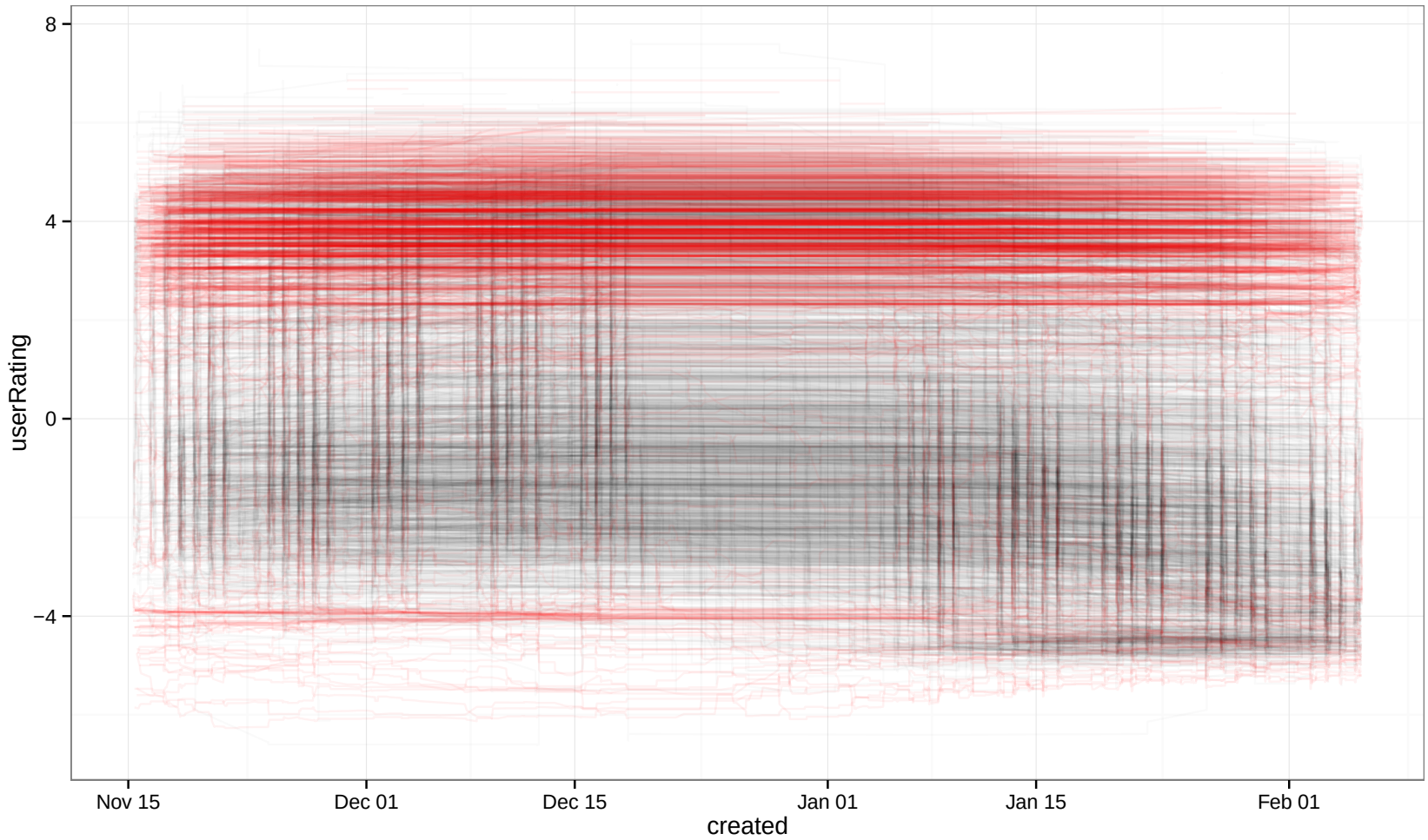

Spel: mollenspel
20
10
0
Rating
Apr 01
Apr 15
May 01
May 15
Jun 01
Date
Sequence length
1
2
3
4
5
6
7
8
9


Spel: mollenspel
Rating
20
10
0
−10
Nov 03
Nov 10
Nov 17
Nov 24
Dec 01
Dec 08
Dec 15
Date
Sequence length
1
2
3
4
5
6
7
8
9

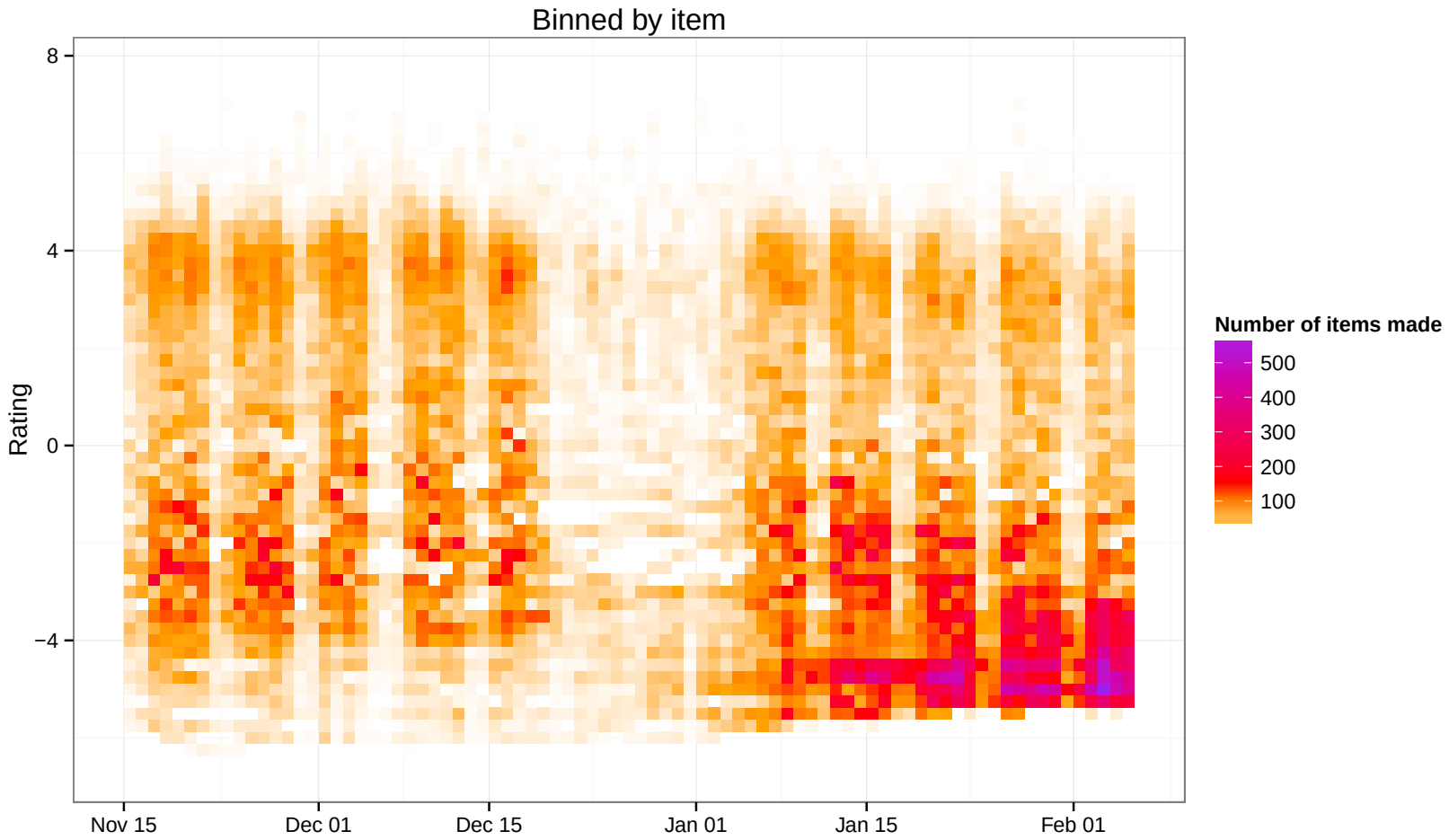


Luister naar alle woorden. Kies daarna per groep het laatste woord dat je hebt gehoord. Klik voor meer uitleg op de luchtbel in de haven. Daar vind je ook een overzicht van alle groepen en de woorden die daarbij horen.

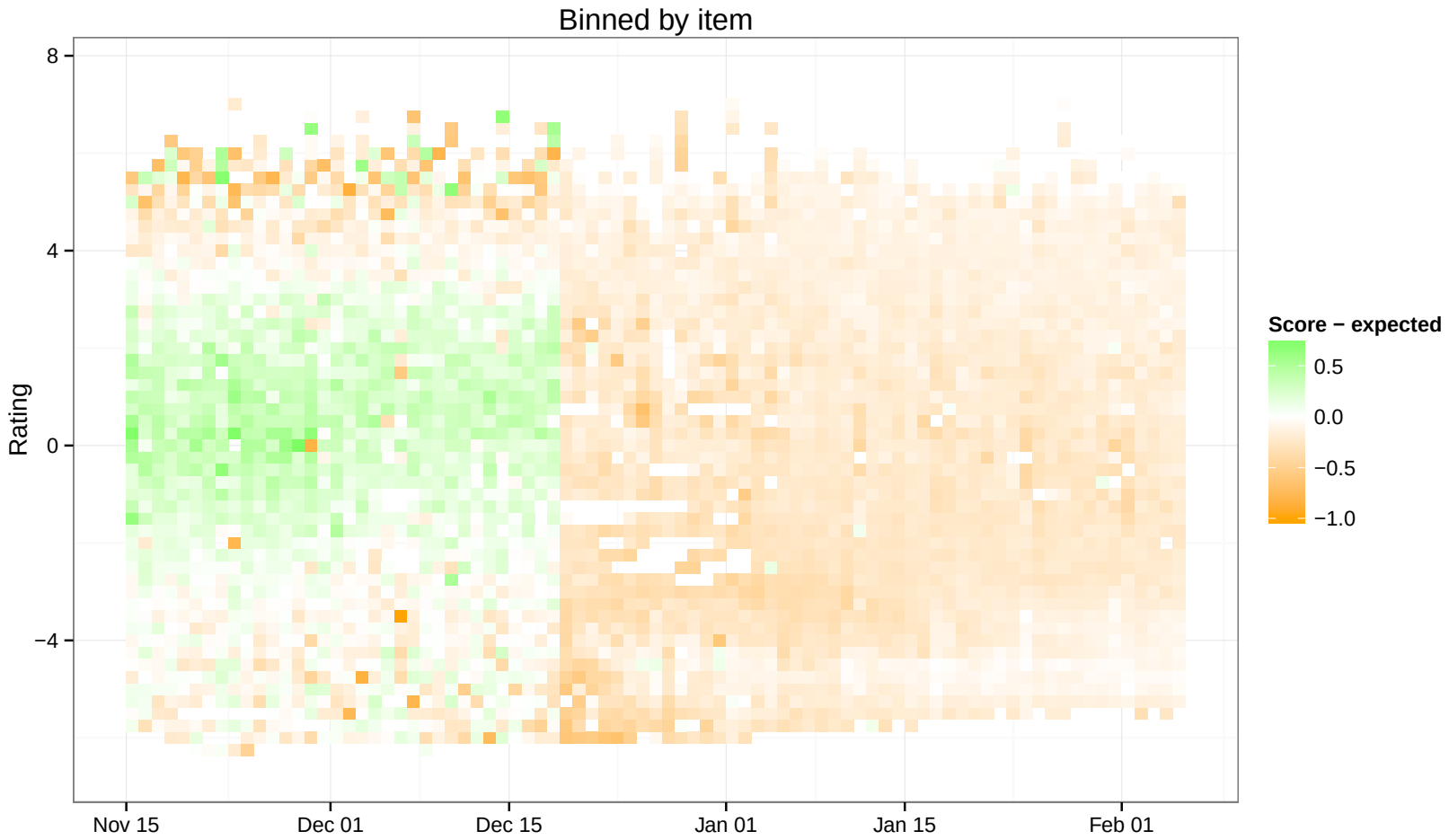
woord.pdf


8
4
0
−4
userRating
Nov 15
Dec 01
Dec 15
Jan 01
Jan 15
Feb 01
created

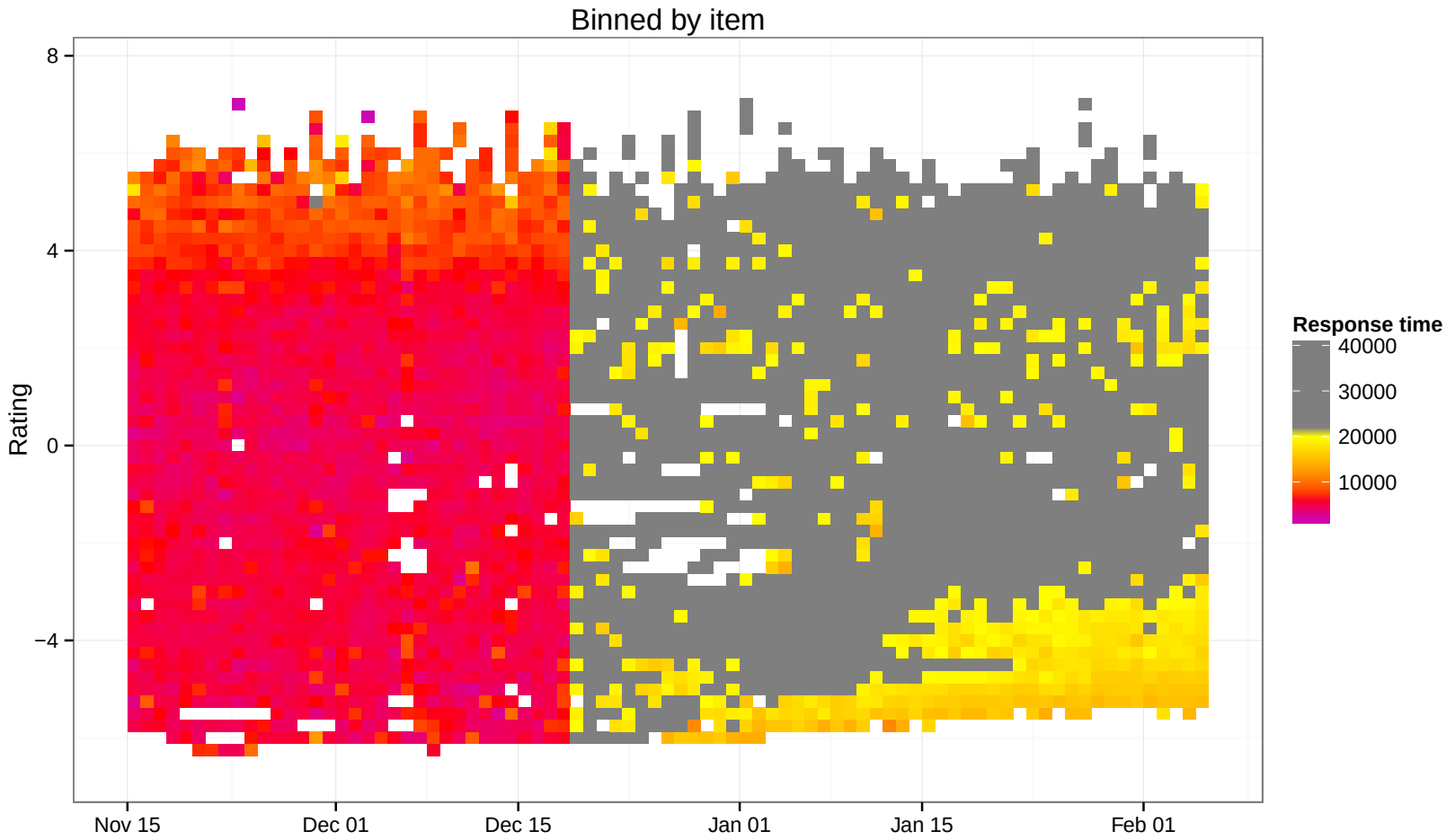
woord.pdf


Binned by item
Rating
8
4
0
−4
Nov 15
Dec 01
Dec 15
Jan 01
Jan 15
Feb 01
Number of items made
500
400
300
200
100

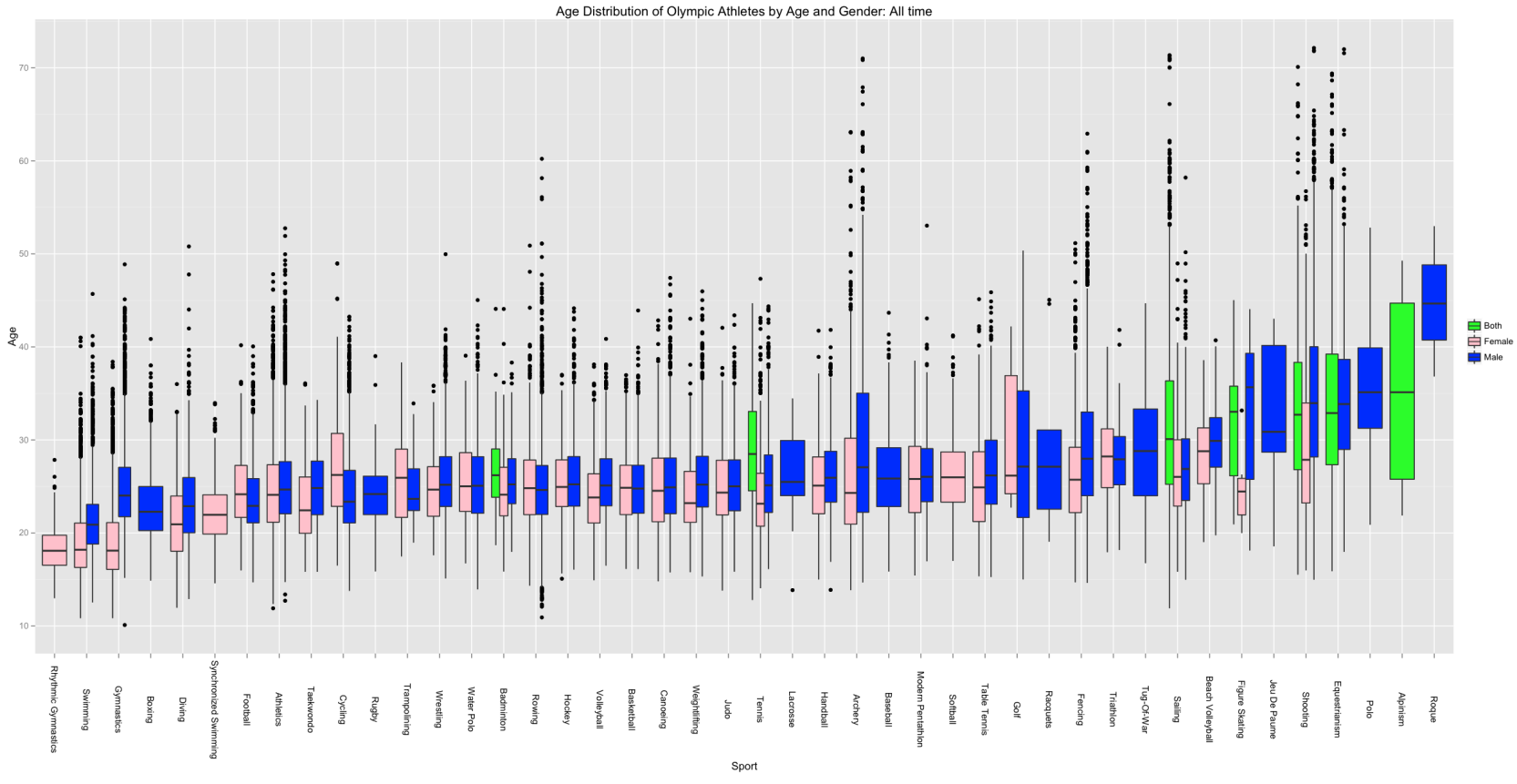
woord.pdf


Binned by item
Rating
8
4
0
−4
Nov 15
Dec 01
Dec 15
Jan 01
Jan 15
Feb 01
Score − expected
0.5
0.0
−0.5
−1.0

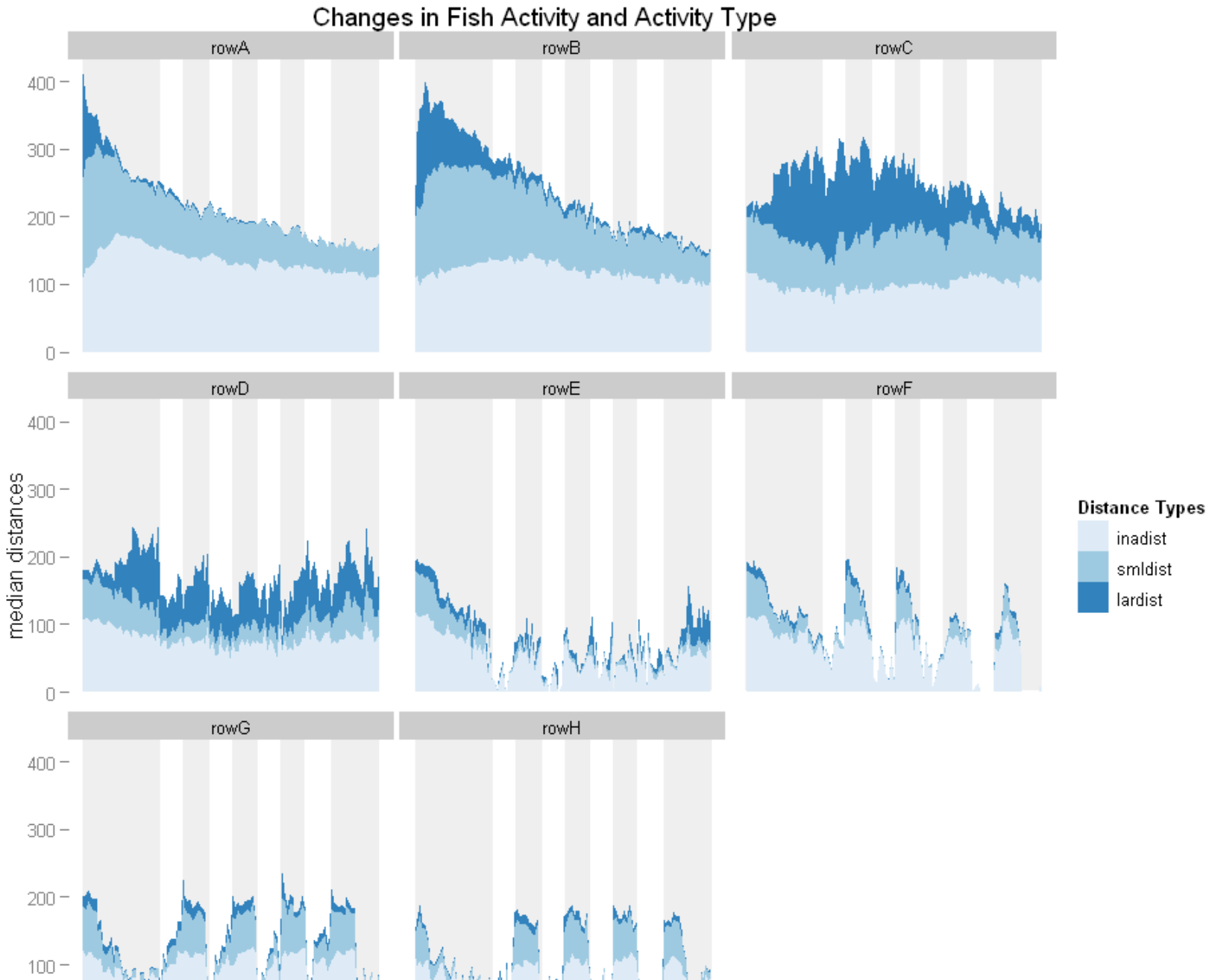
woord.pdf


Binned by item
Rating
8
4
0
−4
Nov 15
Dec 01
Dec 15
Jan 01
Jan 15
Feb 01
Response time
40000
30000
20000
10000

More ggplot2

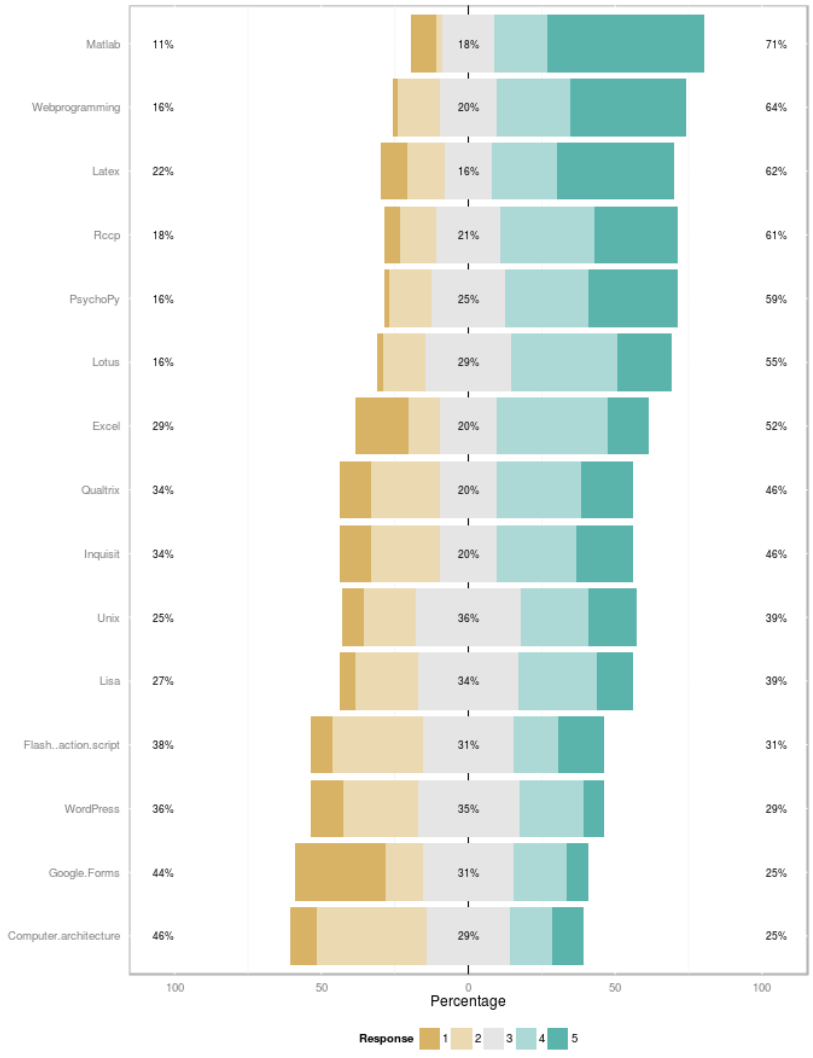


More ggplot2



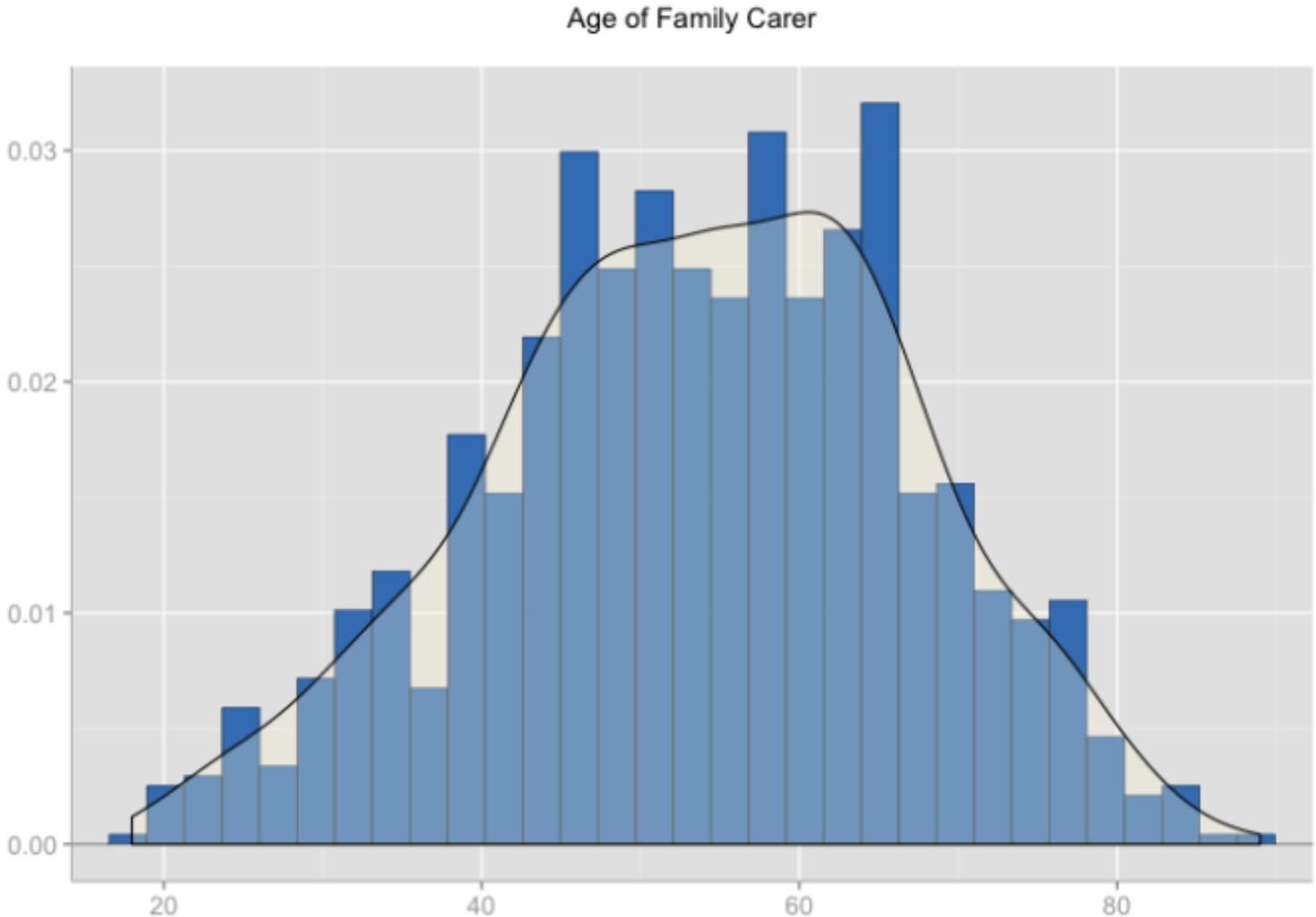
More ggplot2

likert package (Bryer & Speerschneider, 2013)



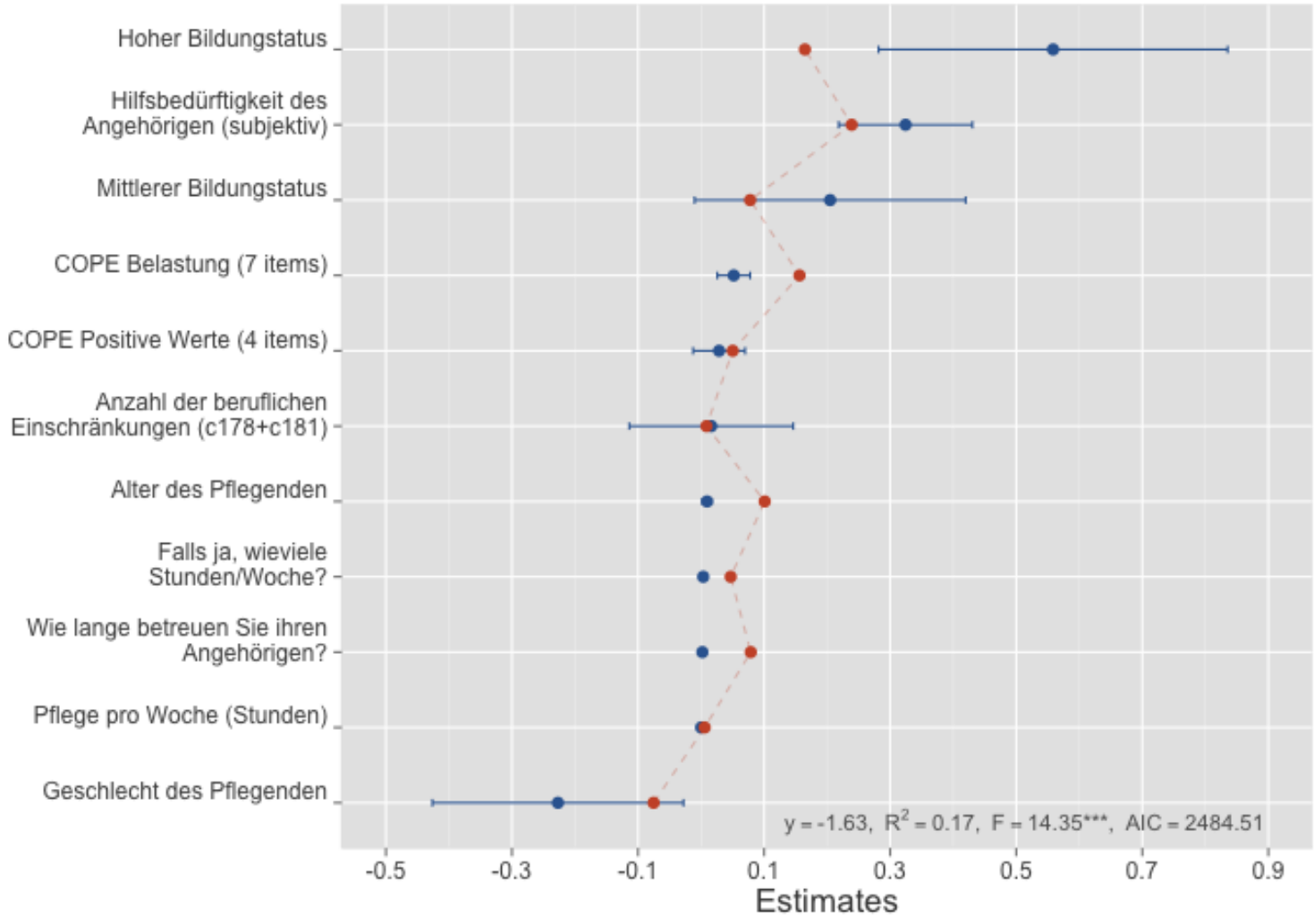
More ggplot2

sjPlot package (Lüdecke, 2014)



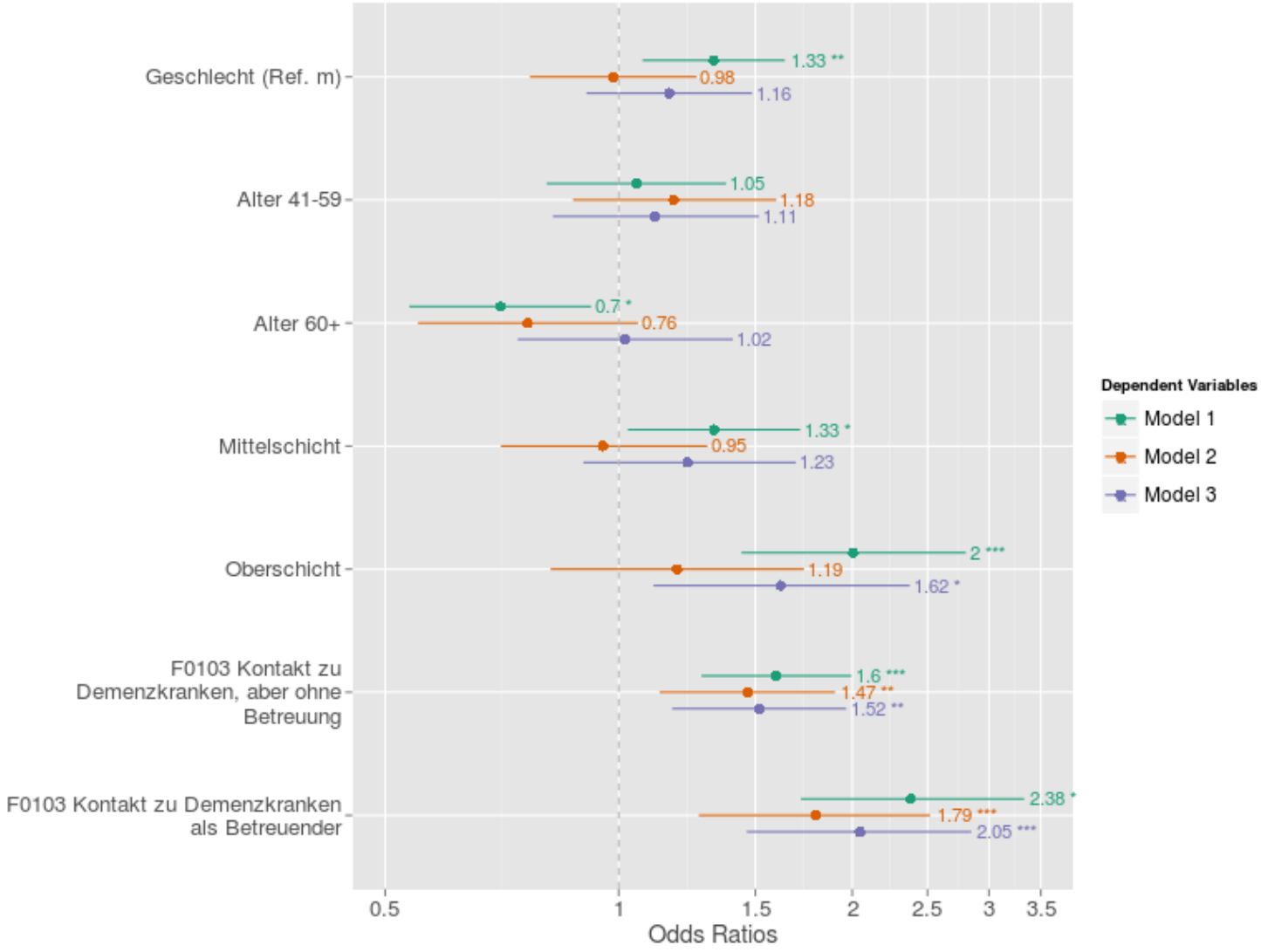
More ggplot2

sjPlot package (Lüdecke, 2014)



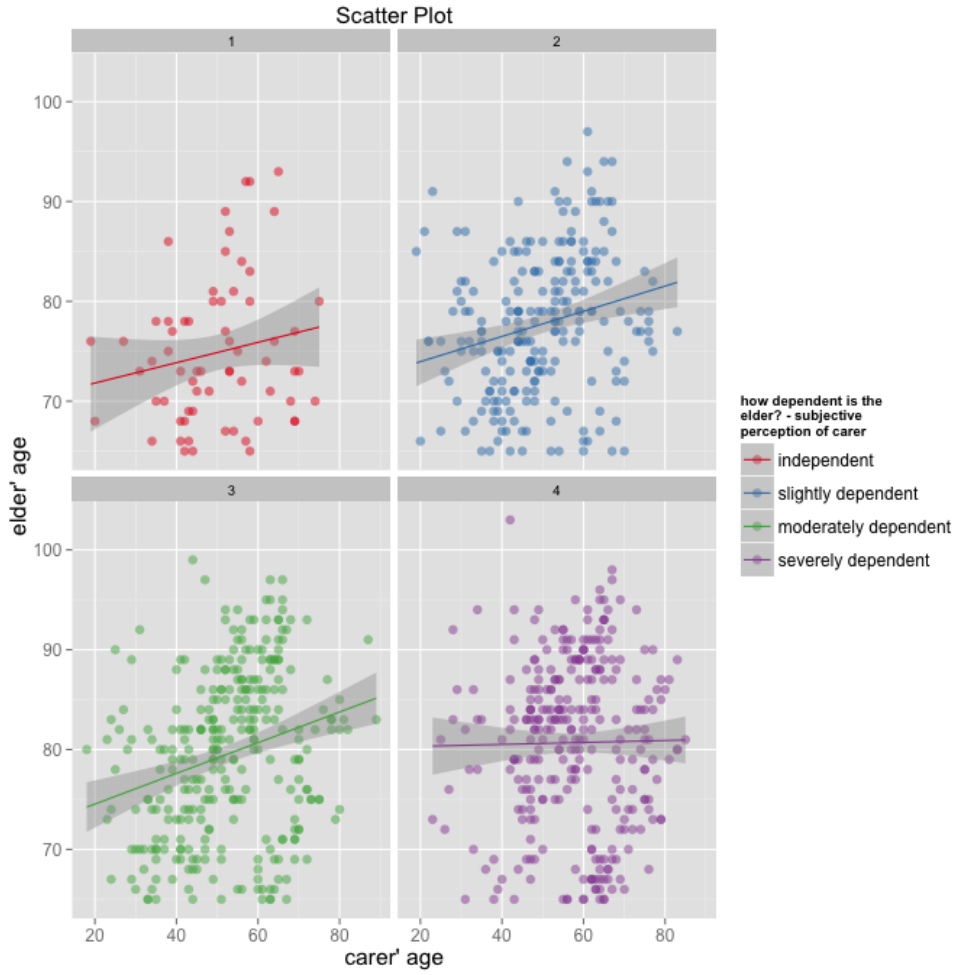
More ggplot2

sjPlot package (Lüdecke, 2014)



More ggplot2

sjPlot package (Lüdecke, 2014)



ggplot2 conclusion

- **ggplot2** can create very complex visualizations with minimal codes
- Automatizes convenient things such as
  - Margins
  - Legend
- Documentation: `http://docs.ggplot2.org/`

Thank you for your attention!

Exercises are on `http://sachaepskamp.com/files/ggplot2_exercises.html`

References I

Bryer, J., & Speerschneider, K. (2013). likert: Functions to analyze and visualize likert type items [Computer software manual]. Retrieved from `http://CRAN.R-project.org/package=likert` (R package version 1.1)

Lüdecke, D. (2014). sjplot: sjplot - data visualization for statistics in social science [Computer software manual]. Retrieved from `http://CRAN.R-project.org/package=sjPlot` (R package version 1.3)

Wickham, H. (2009). *ggplot2: elegant graphics for data analysis*. Springer New York. Retrieved from `http://had.co.nz/ggplot2/book`

Wilkinson, L., Wills, D., Rope, D., Norton, A., & Dubbs, R. (2006). *The grammar of graphics*. Springer.