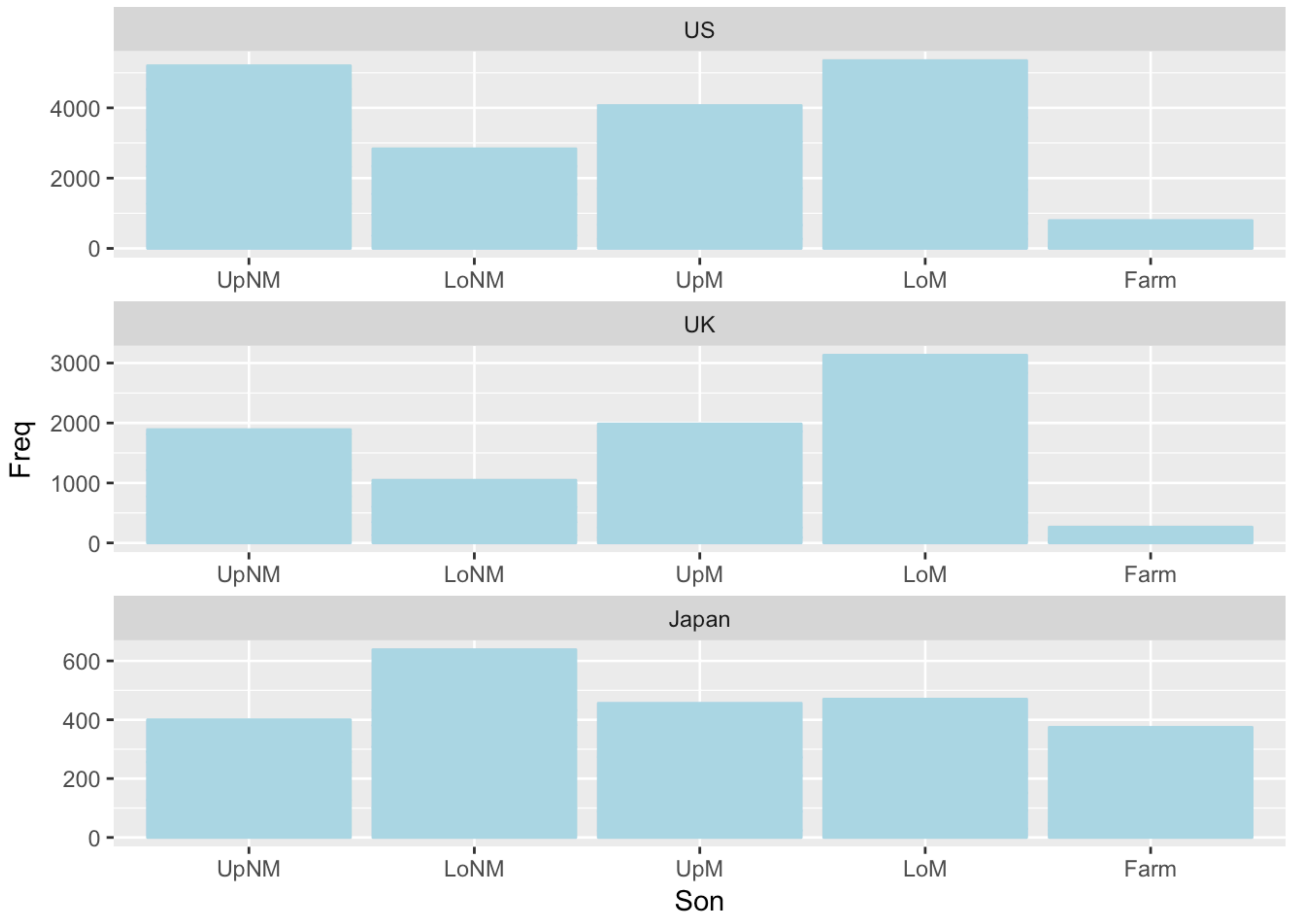
EDAV Hmk2 feedback

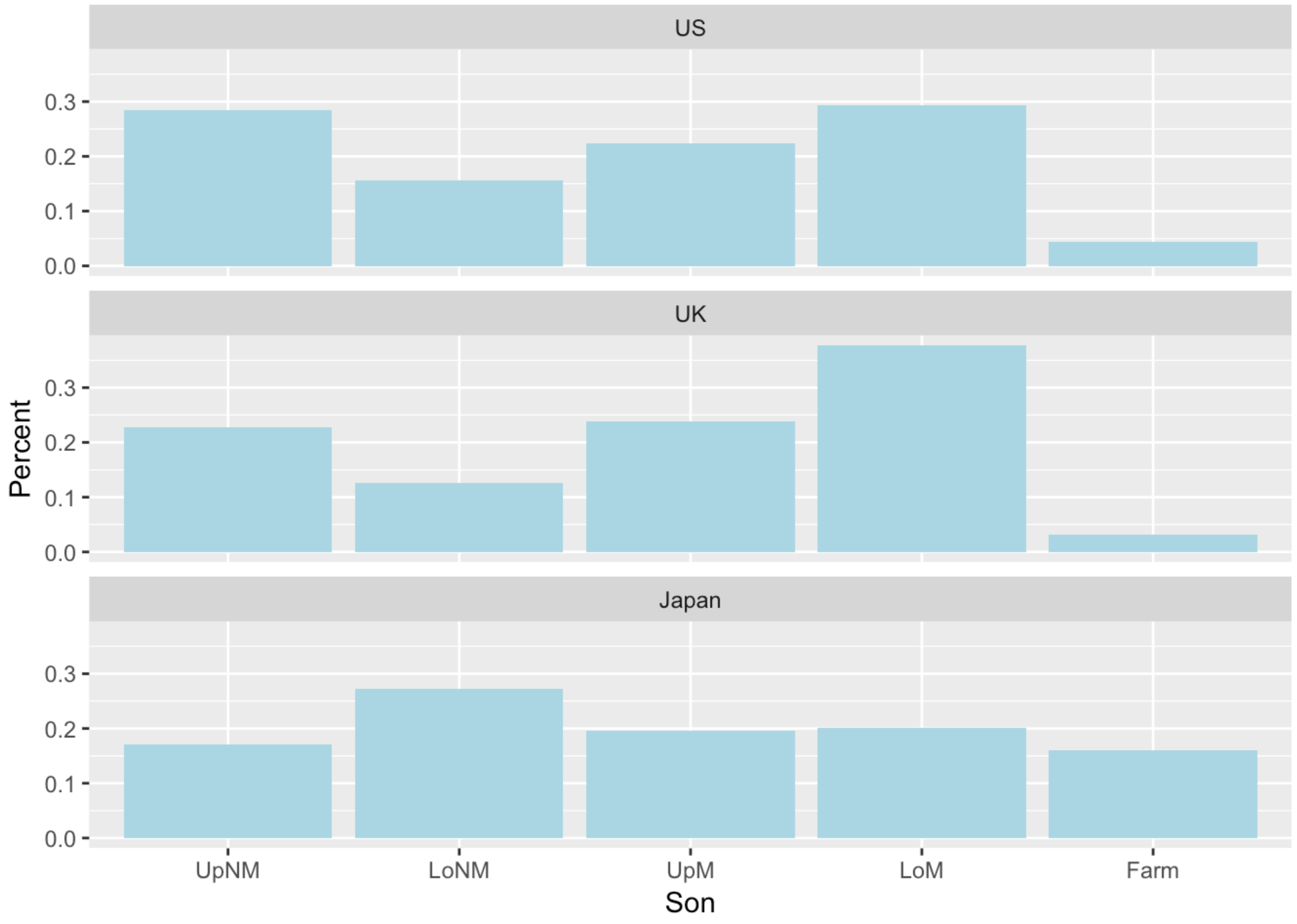
Joyce Robbins

2/18/2017

```r
# Chap 4, p. 73, #5 Occupational Mobility
# Keep the original order of classes, free the scales to compare
# within each country
library(vcdExtra)
library(ggplot2)
ggplot(Yamaguchi87, aes(x = Son, y = Freq)) +
    geom_col(color = "lightblue", fill = "lightblue") +
    facet_wrap(~Country, nrow = 3, scales = "free")
```

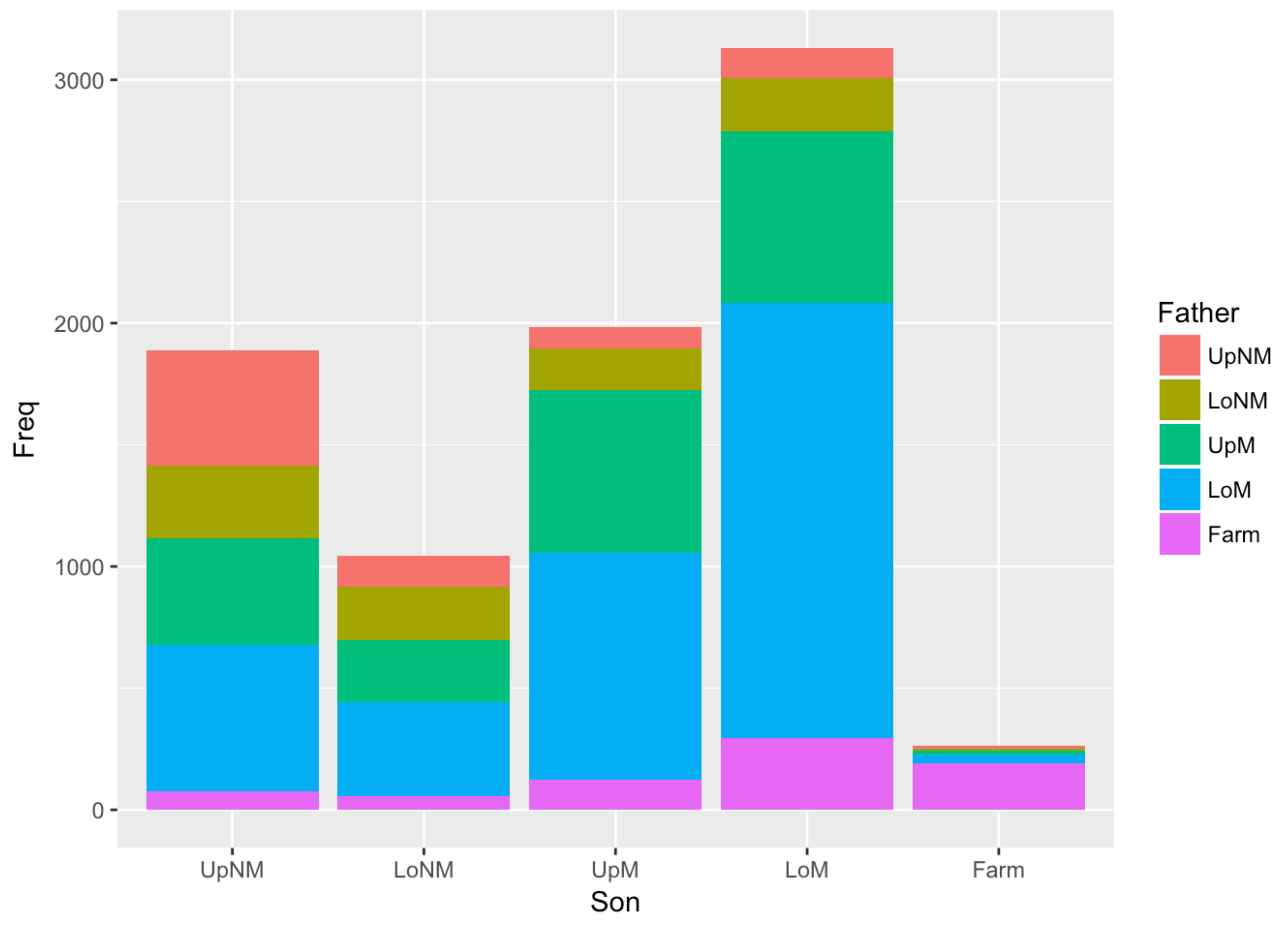


```r
library(dplyr)
# or make a relative frequency histogram, which is essentially
# the same thing
# there does not appear to be a way to do this without calculating
# the percents, see: https://github.com/tidyverse/ggplot2/issues/574

mydf <- Yamaguchi87 %>% group_by(Son, Country) %>%
    summarize(Freq = sum(Freq)) %>% group_by(Country) %>%
    mutate(Percent = Freq / sum(Freq))

ggplot(mydf, aes(x = Son, y = Percent)) +
    geom_col(fill = "lightblue") +
    facet_wrap(~Country, nrow = 3)
```

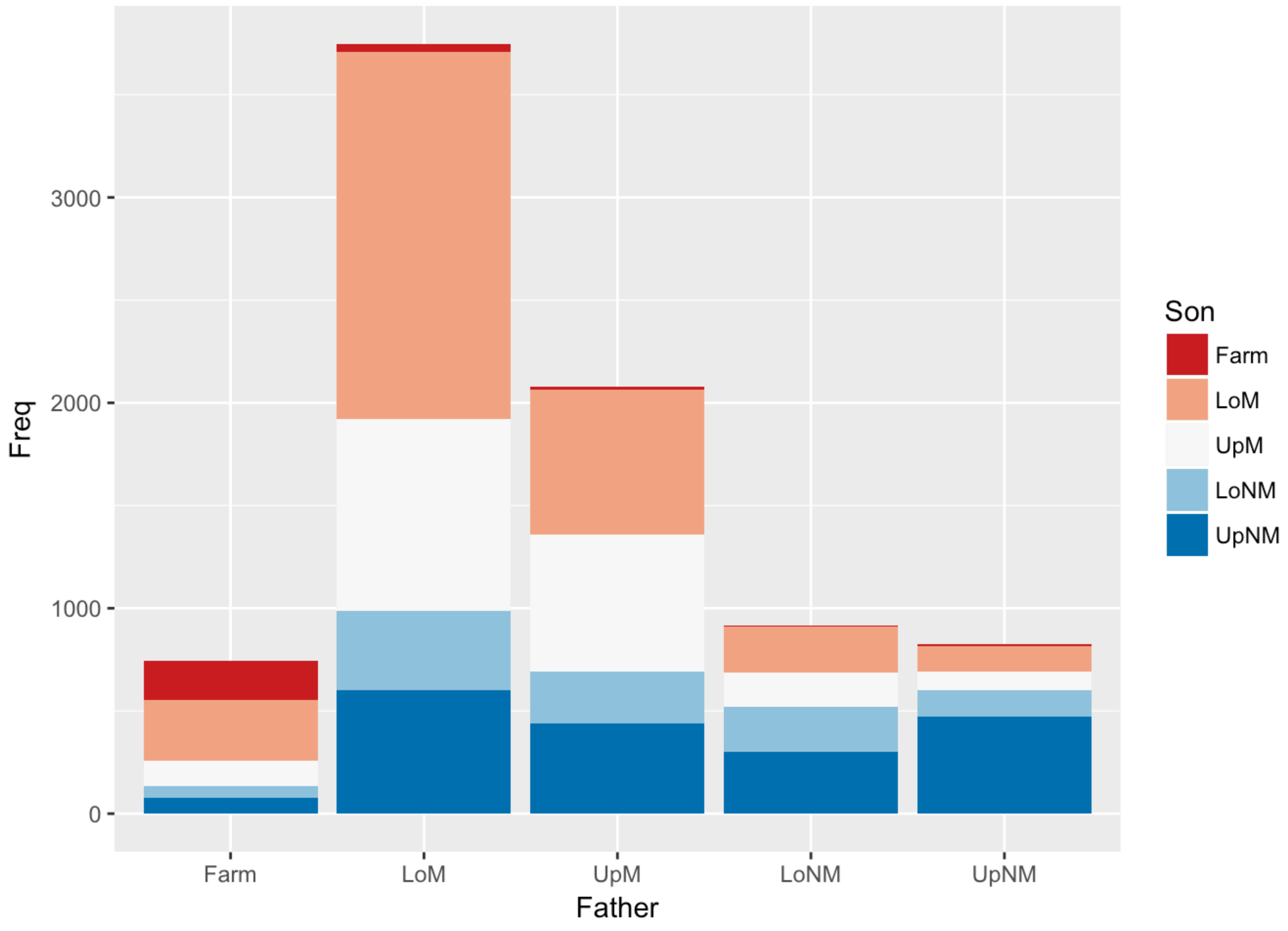


UK Barcharts

```r
## Actual Data
UKdf <- Yamaguchi87 %>% filter(Country == "UK")
ggplot(UKdf, aes(x = Son, y = Freq, fill = Father)) + geom_col()
```

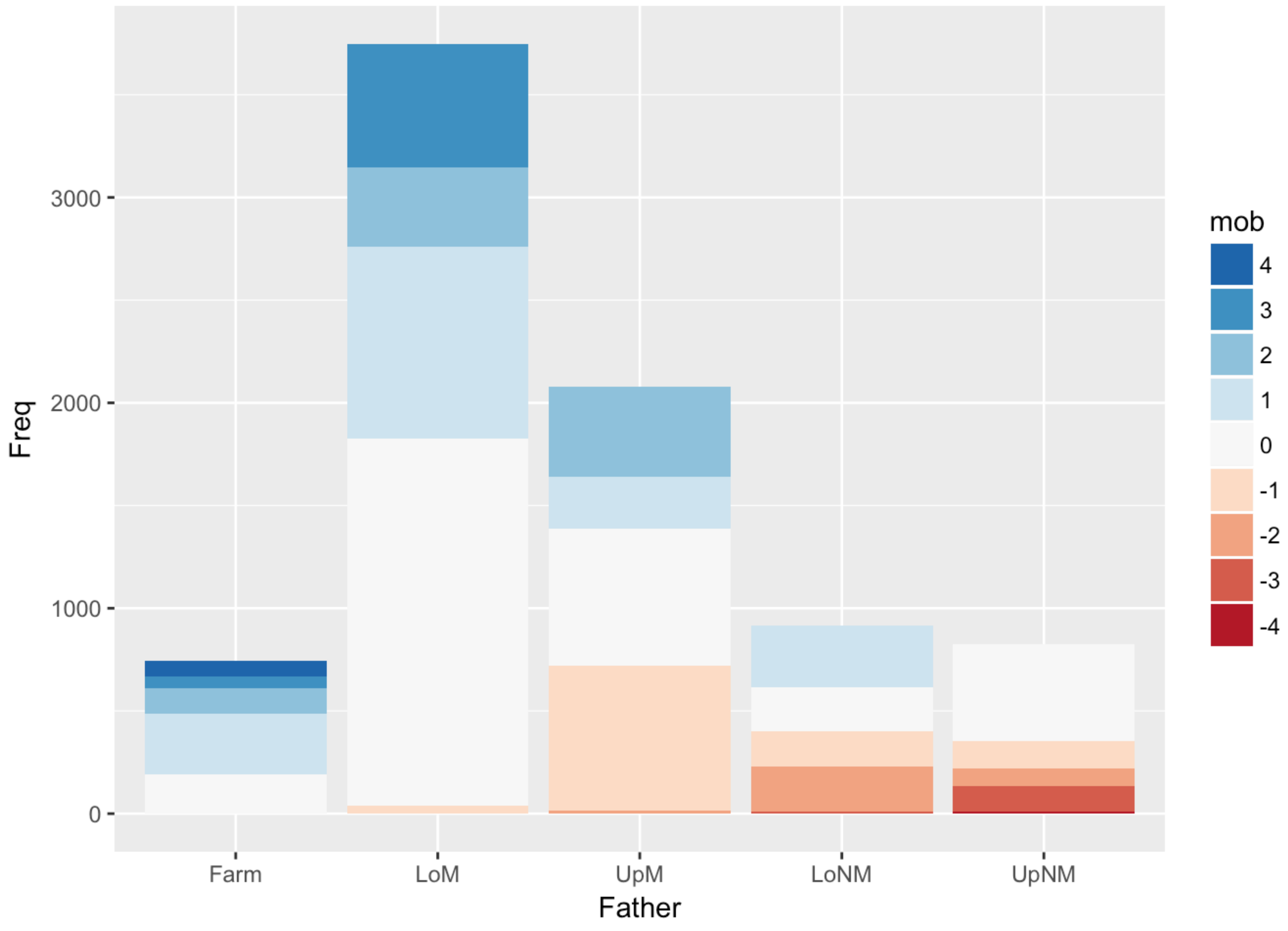


```
UKdf$Son <- factor(UKdf$Son, levels = c("Farm", "LoM", "UpM", "LoNM", "UpNM"))
UKdf$Father <- factor(UKdf$Father,
                      levels = c("Farm", "LoM", "UpM", "LoNM", "UpNM"))
ggplot(UKdf, aes(x = Father, y = Freq, fill = Son)) + geom_col() +
    scale_fill_brewer(palette = "RdBu")
```

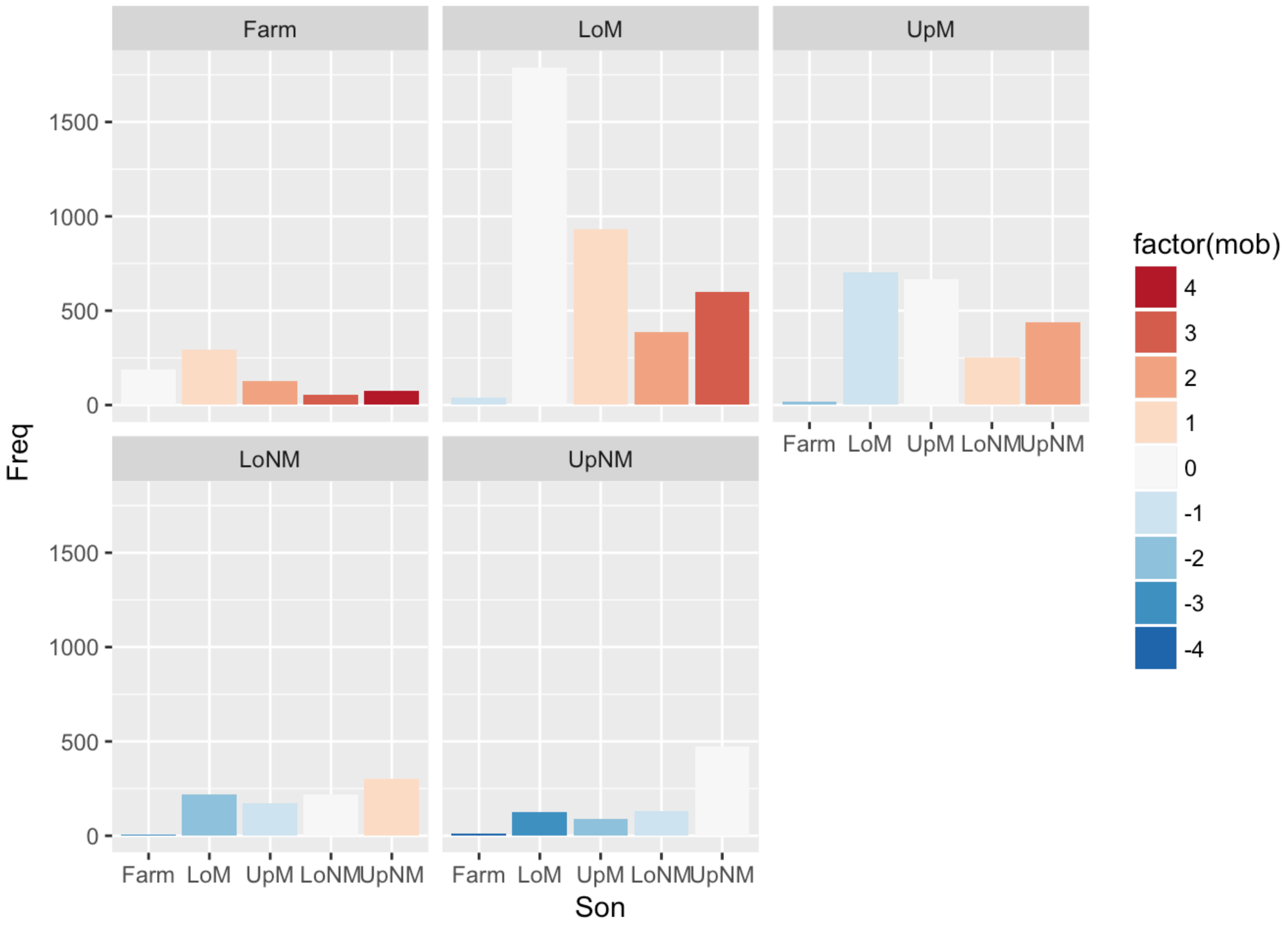


Mobility

```
library(RColorBrewer)
UKmob <- UKdf %>% mutate(mob = as.numeric(Son) - as.numeric(Father))
UKmob$mob <- factor(UKmob$mob, levels = 4:-4)
fills <- rev(brewer.pal(9, 'RdBu'))
ggplot(UKmob, aes(x = Father, y = Freq, fill = mob)) + geom_col() +
    scale_fill_manual(values = fills) + theme(plot.subtitle = element_text(vjust = 1)
,
    plot.caption = element_text(vjust = 1))
```



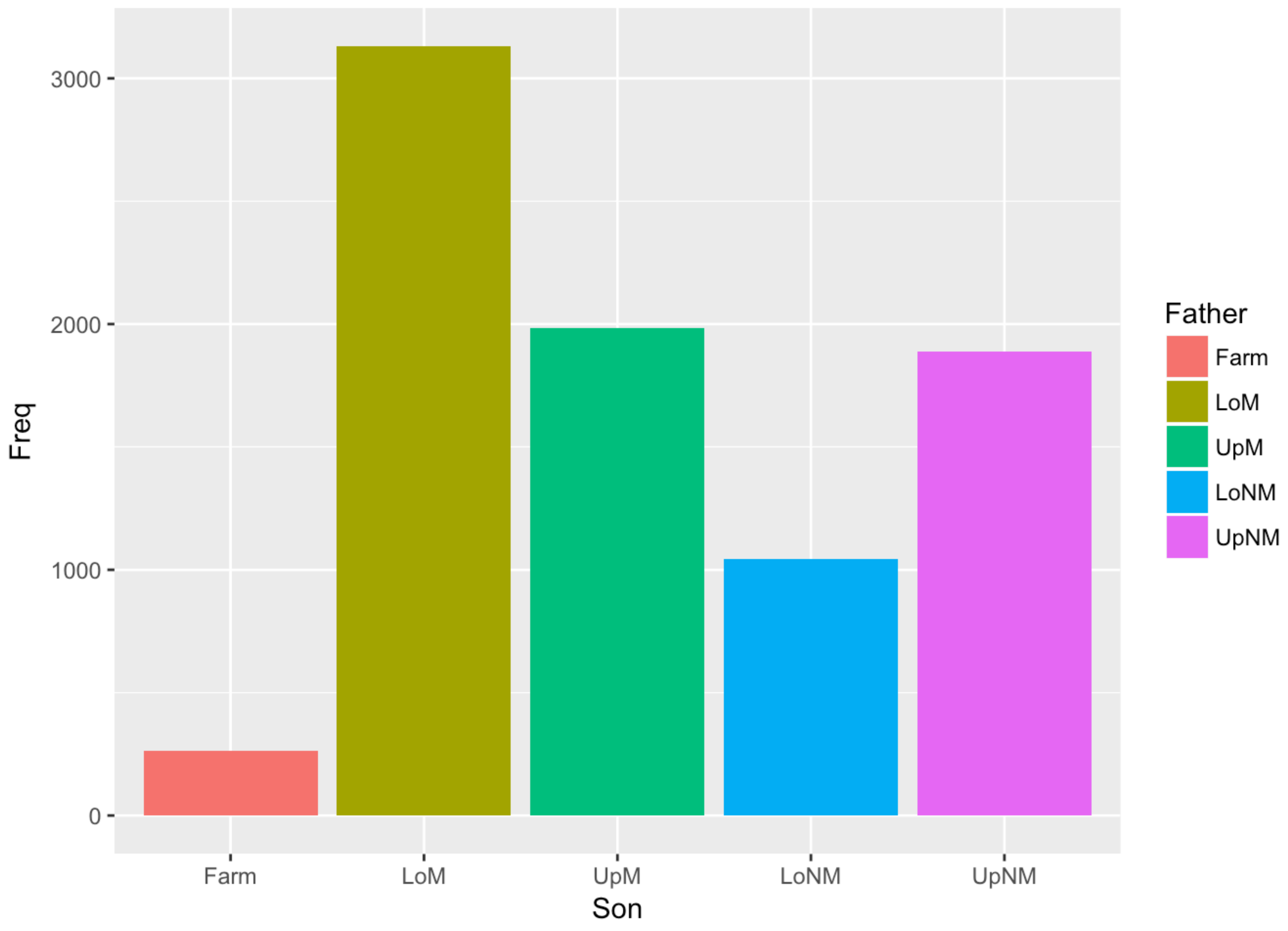
```
ggplot(UKmob, aes(x = Son, y = Freq)) + geom_col(aes(fill = factor(mob))) +
    facet_wrap(~Father) + scale_fill_brewer(palette = "RdBu")
```



```
## Rigged Data (no mobility)
UKrigged <- UKdf %>% group_by(Son) %>%
    mutate(Total = sum(Freq)) %>%
    mutate(Freq = ifelse(Son == Father, Total, 0)) %>%
    select(-Total) %>% ungroup()

UKrigged
```

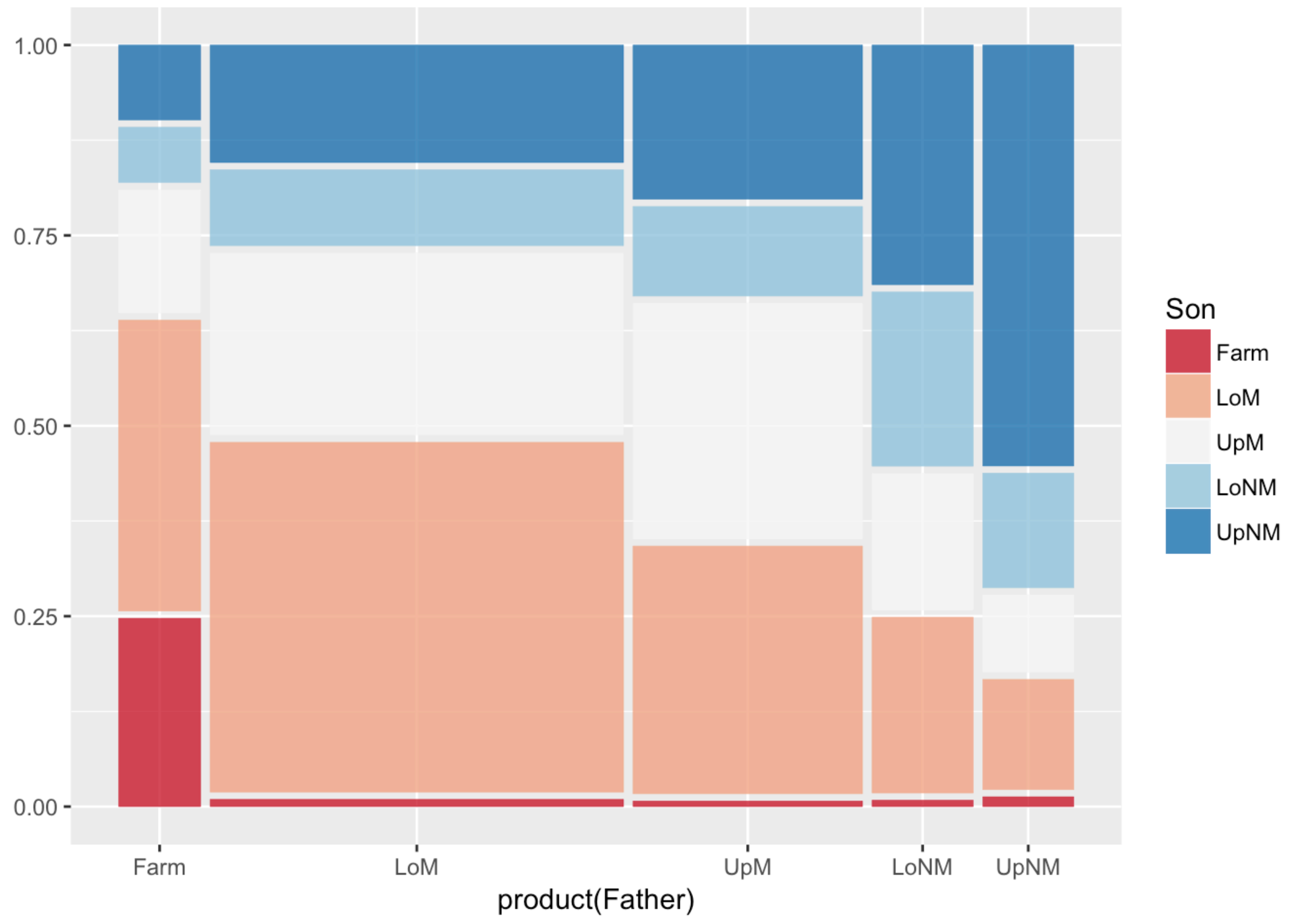
```
## # A tibble: 25 × 4
##        Son Father Country  Freq
##     <fctr> <fctr>  <fctr> <dbl>
## 1     UpNM   UpNM      UK  1889
## 2     LoNM   UpNM      UK     0
## 3      UpM   UpNM      UK     0
## 4      LoM   UpNM      UK     0
## 5     Farm   UpNM      UK     0
## 6     UpNM   LoNM      UK     0
## 7     LoNM   LoNM      UK  1045
## 8      UpM   LoNM      UK     0
## 9      LoM   LoNM      UK     0
## 10    Farm   LoNM      UK     0
## # ... with 15 more rows
```

```
ggplot(UKrigged, aes(x = Son, y = Freq, fill = Father)) + geom_col()
```

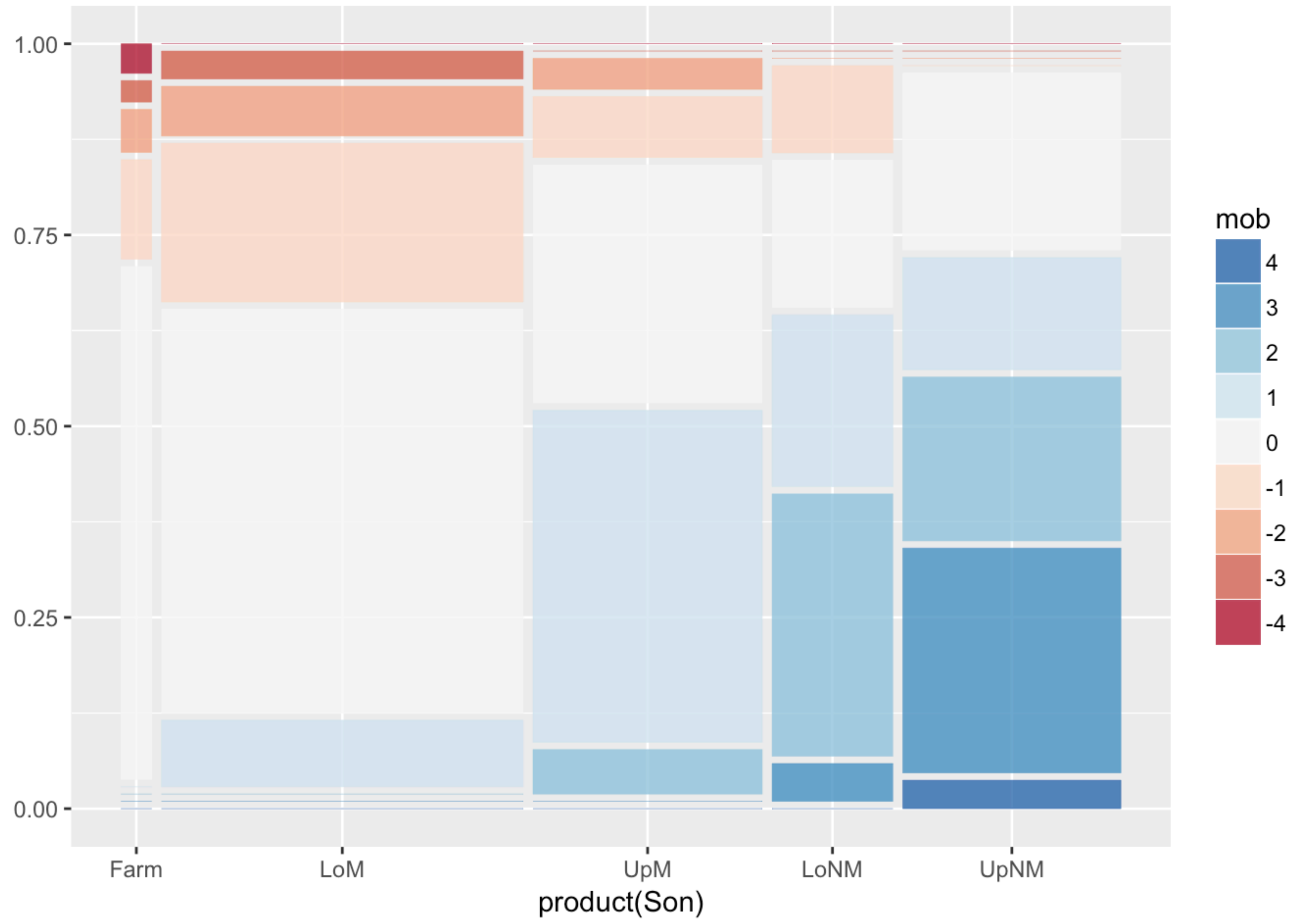


UK Mosaic

```
library(ggmosaic)
ggplot(UKdf, aes(weight = Freq, x = product(Father), fill = Son)) +
    geom_mosaic() + scale_fill_brewer(palette = "RdBu")
```

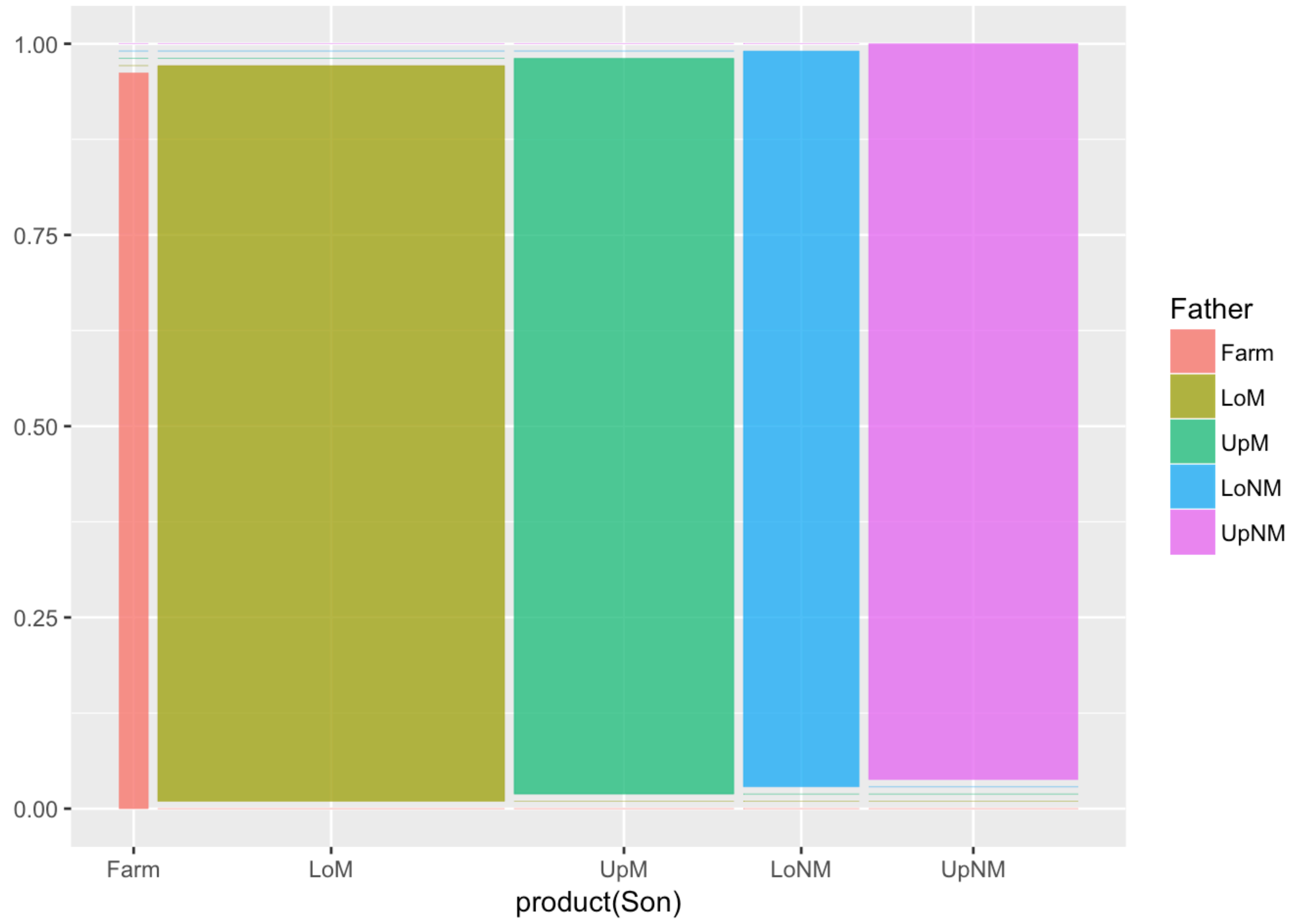


```
ggplot(UKmob, aes(weight = Freq, x = product(Son), fill = mob)) +
    geom_mosaic() + scale_fill_manual(values = fills)
```



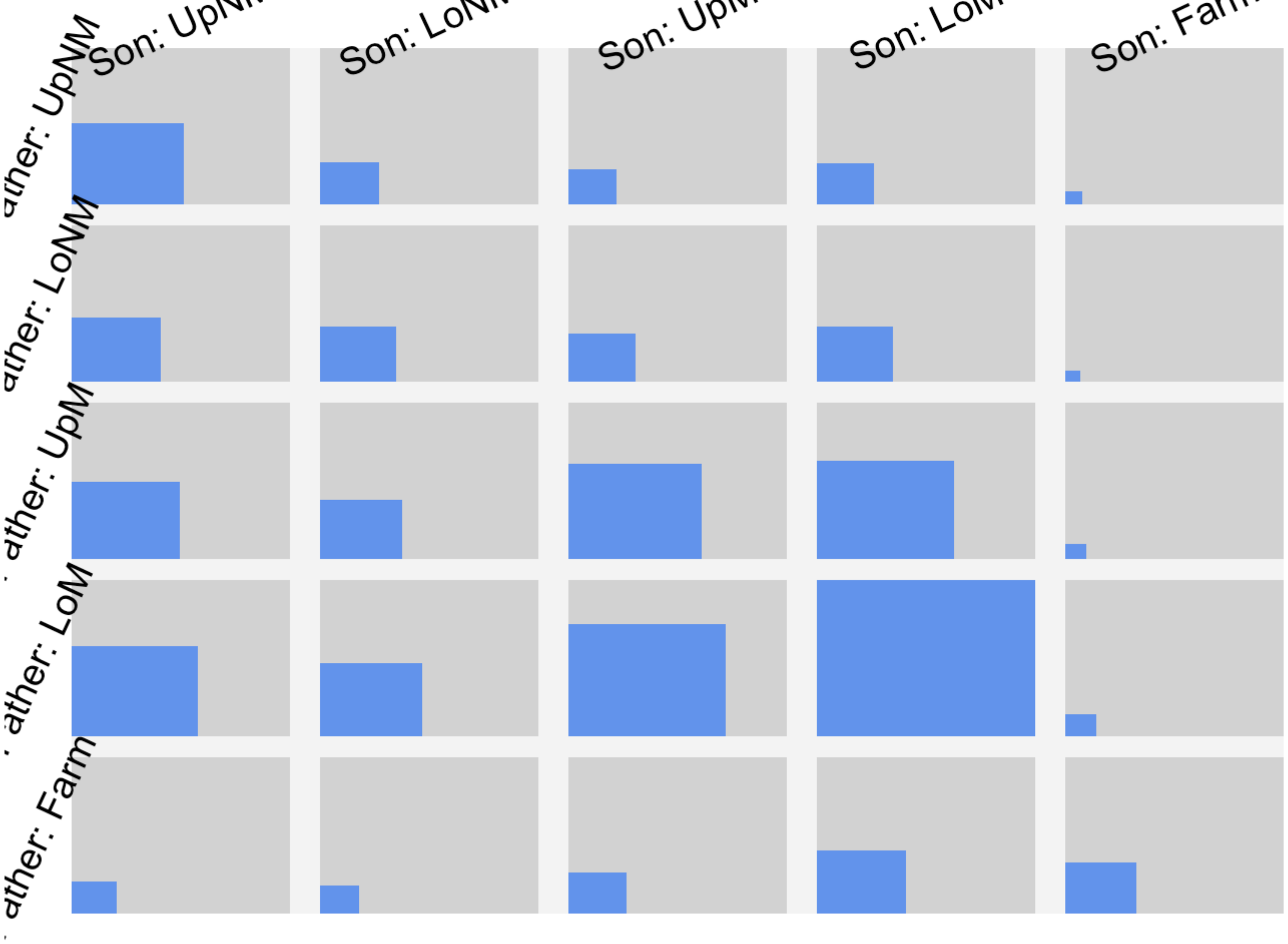
UK Mosaic – rigged data

```
ggplot(UKrigged, aes(weight = Freq, x = product(Son), fill = Father)) +
    geom_mosaic()
```



Fluctuation Diagrams

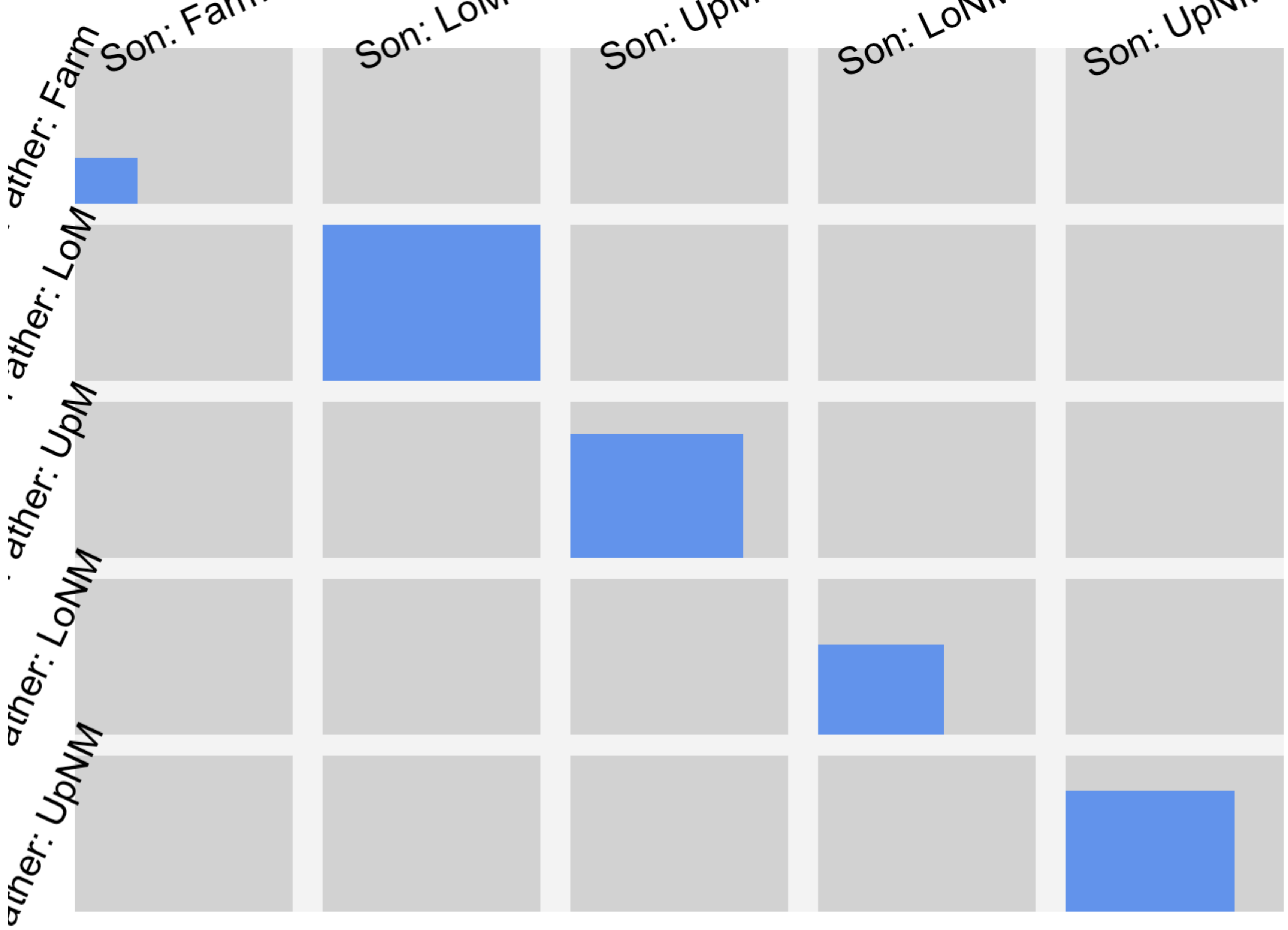
```r
library(vcdExtra) # Yamaguchi87 dataset
library(extracat) # for fluctile()
library(tidyr)
UKdf <- Yamaguchi87 %>% filter(Country == "UK")
# Convert to matrix
UKmat <- UKdf %>% spread(key = Son, value = Freq) %>%
    select(-Father, -Country) %>% as.matrix()
rownames(UKmat) <- paste("Father:", colnames(UKmat))
colnames(UKmat) <- paste("Son:", colnames(UKmat))
fluctile(UKmat, just = "lb", tile.col = "cornflowerblue")
```



```
## viewport[base]
```

Fluctuation Diagram – rigged data

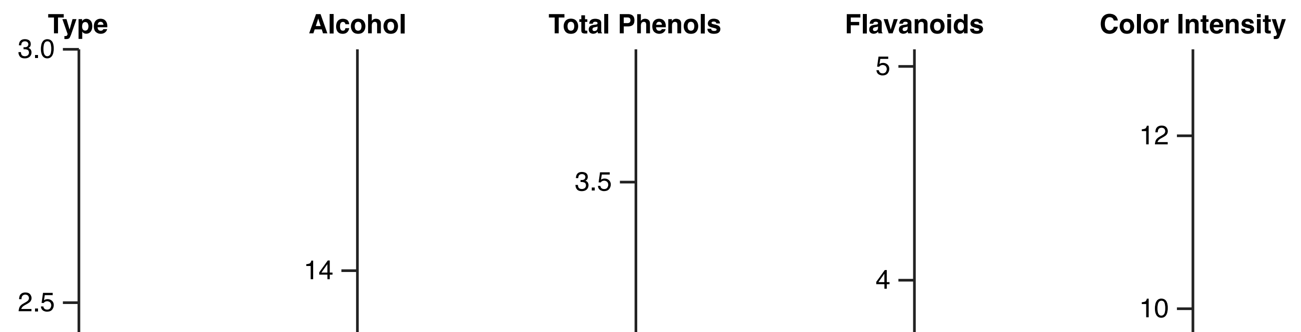
```
# rigged data (no mobility)
UKriggedmat <- UKrigged %>% spread(key = Son, value = Freq) %>%
    select(-Father, -Country) %>% as.matrix()
rownames(UKriggedmat) <- paste("Father:", colnames(UKriggedmat))
colnames(UKriggedmat) <- paste("Son:", colnames(UKriggedmat))

fluctile(UKriggedmat, just = "lb", tile.col = "cornflowerblue")
```

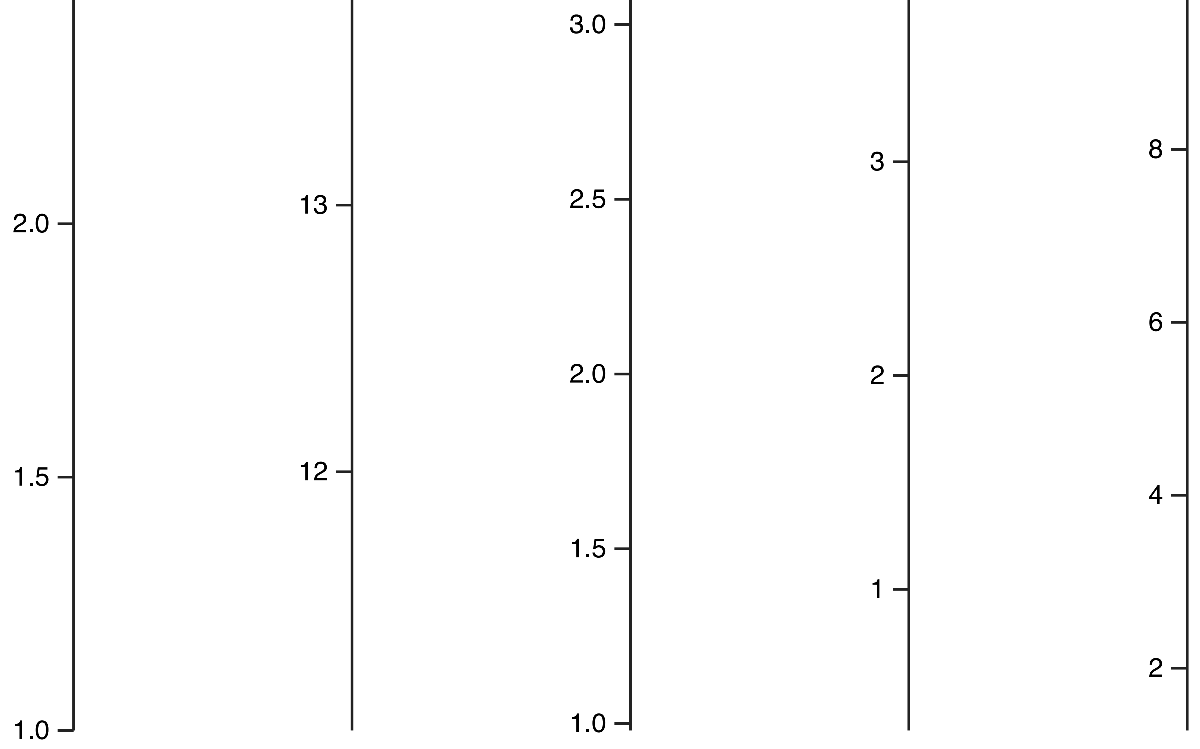


```
## viewport[base]
```

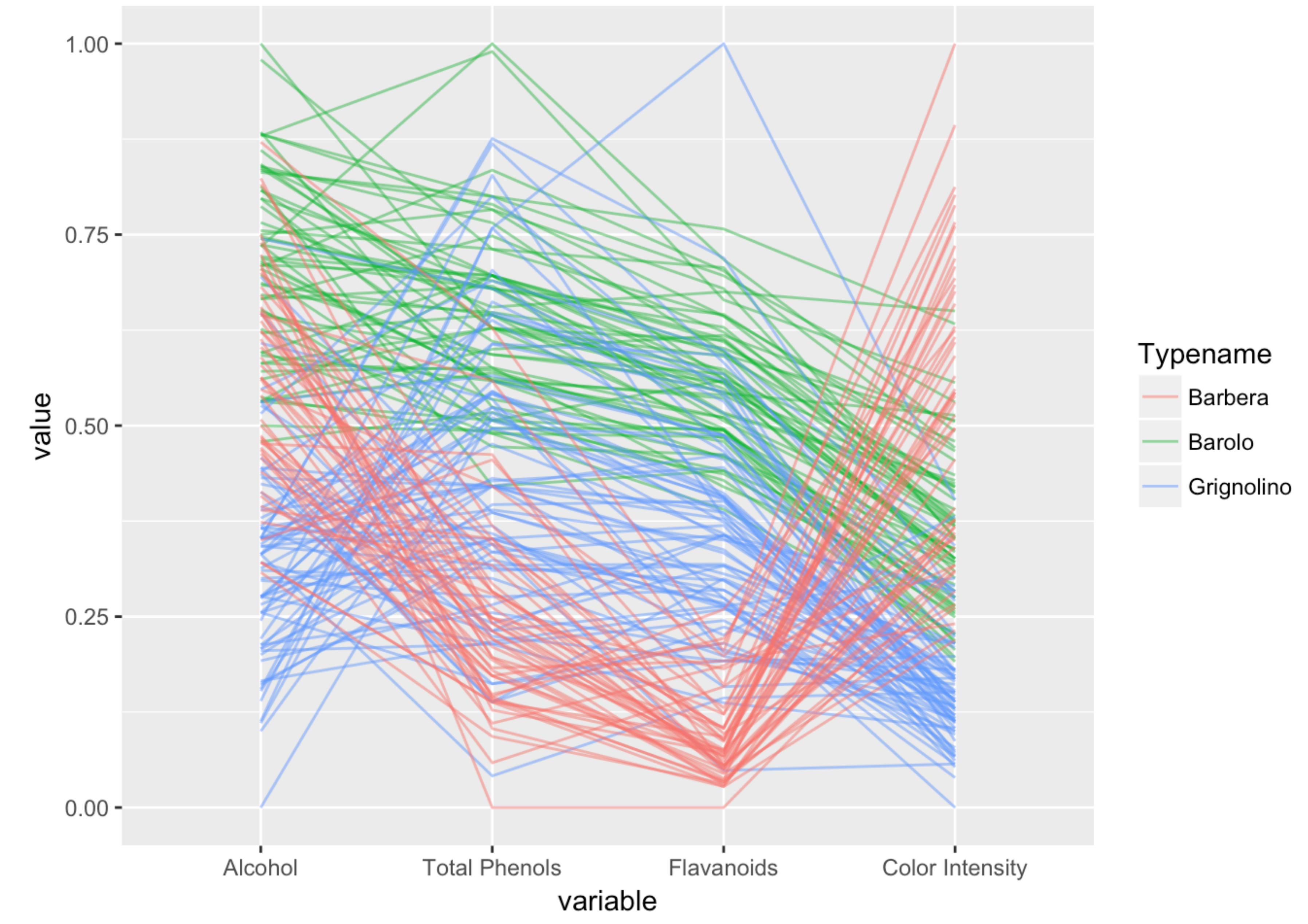
Wine

```
library(parcoords)
library(pgmm)
data("wine")
parcoords(wine[,c(1, 2, 16, 17, 20)], rownames = F, brushMode = "1D-axes",
            reorderable = T, queue = T, alpha = .5,
          color = list(colorScale = htmlwidgets::
                          JS('d3.scale.category10()'),
                        colorBy = "Type"))
```





```
library(GGally)
winetypes <- c("Barolo", "Grignolino", "Barbera")
namewine <- wine %>% mutate(Typename = winetypes[Type])
ggparcoord(namewine, columns = c(2, 16, 17, 20),
           groupColumn = "Typename",
           alpha = .5, scale = "uniminmax")
```



1.00
0.75
0.50
0.25
0.00
value
Alcohol
Total Phenols
Flavanoids
Color Intensity
variable
Typename
Barbera
Barolo
Grignolino