Replication file for the Russian intermarriage paper

Alexey Bessudnov

5 February 2020

The file has been compiled with R version 3.6.2 and R Markdown version 2.0.

```
# Attaching the packages.
library(tidyverse)
# tidyverse version 1.3.0
library(knitr)
# knitr version 1.26
library(gnm)
# gnm version 1.1.0
library(lmtest)
# lmtest version 0.9.37
```

Read the data

The data contain six tables for the following cities: Moscow, Rostov, Kazan, Ufa, Makhachkala and Vladikavkaz. For each city the data represent the number of married couples by wife's and husband's ethnicities, only for locally born wives and for wives in three age groups (16 to 35, 36 to 50 and over 50), taken from the 2010 Russian census.

```
Kazan <- read_csv("data/Kazan.csv")
Moscow <- read_csv("data/Moscow.csv")
Vladikavkaz <- read_csv("data/Vladikavkaz.csv")
Makhachkala <- read_csv("data/Makhachkala.csv")
Ufa <- read_csv("data/Ufa.csv")
Rostov <- read_csv("data/Rostov.csv")
```

Prepare the data

First we want to collapse the data tables combining marriages across all three age groups.

```
# function to collapse data tables
collapseData <- function(df) {
        df %>%
        group_by(ethn.wife, ethn.husband) %>%
        summarise(
                Freq = sum(Freq)
        )
}
```

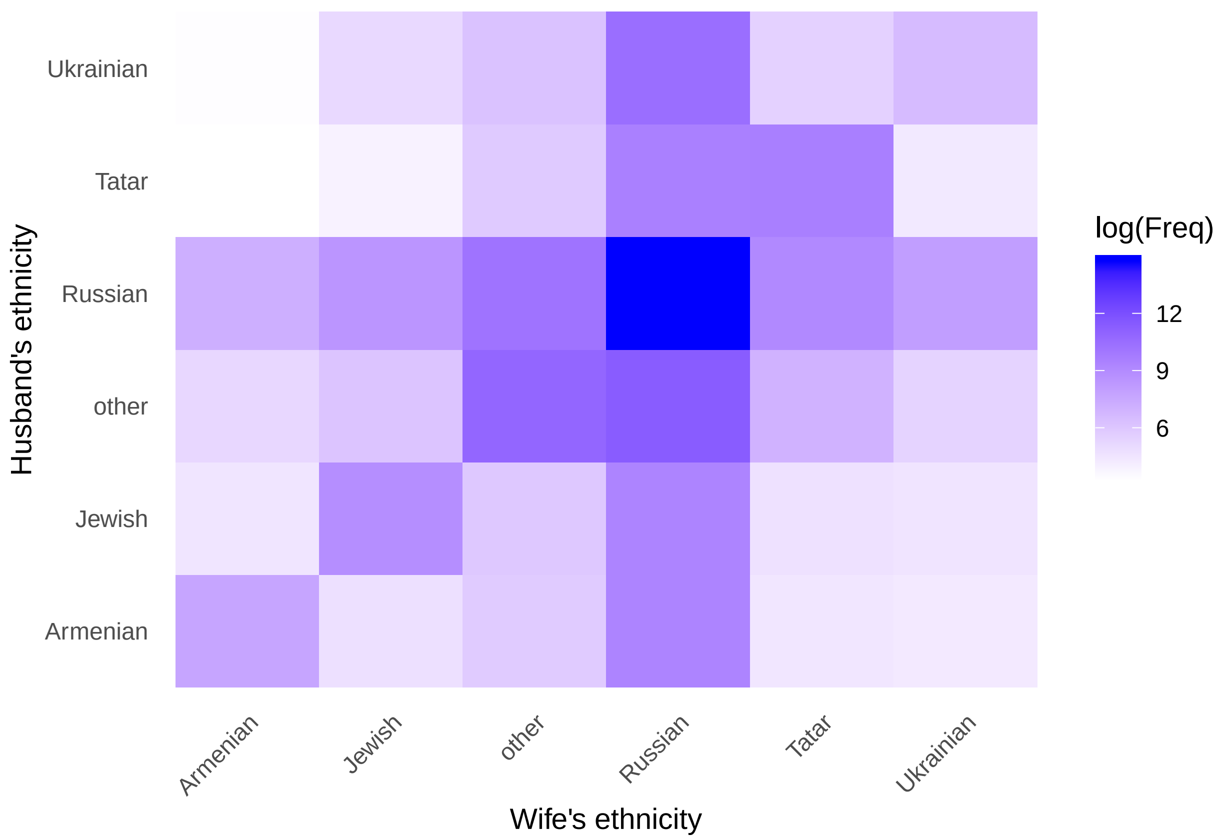
Visualise contingency tables for intermarriages

The heatmaps represent logged frequencies in the contingency tables of ethnicities of wives and husbands. Grey cells are cells with zero observations (so that log is not defined). Note that the data are for locally born women only.

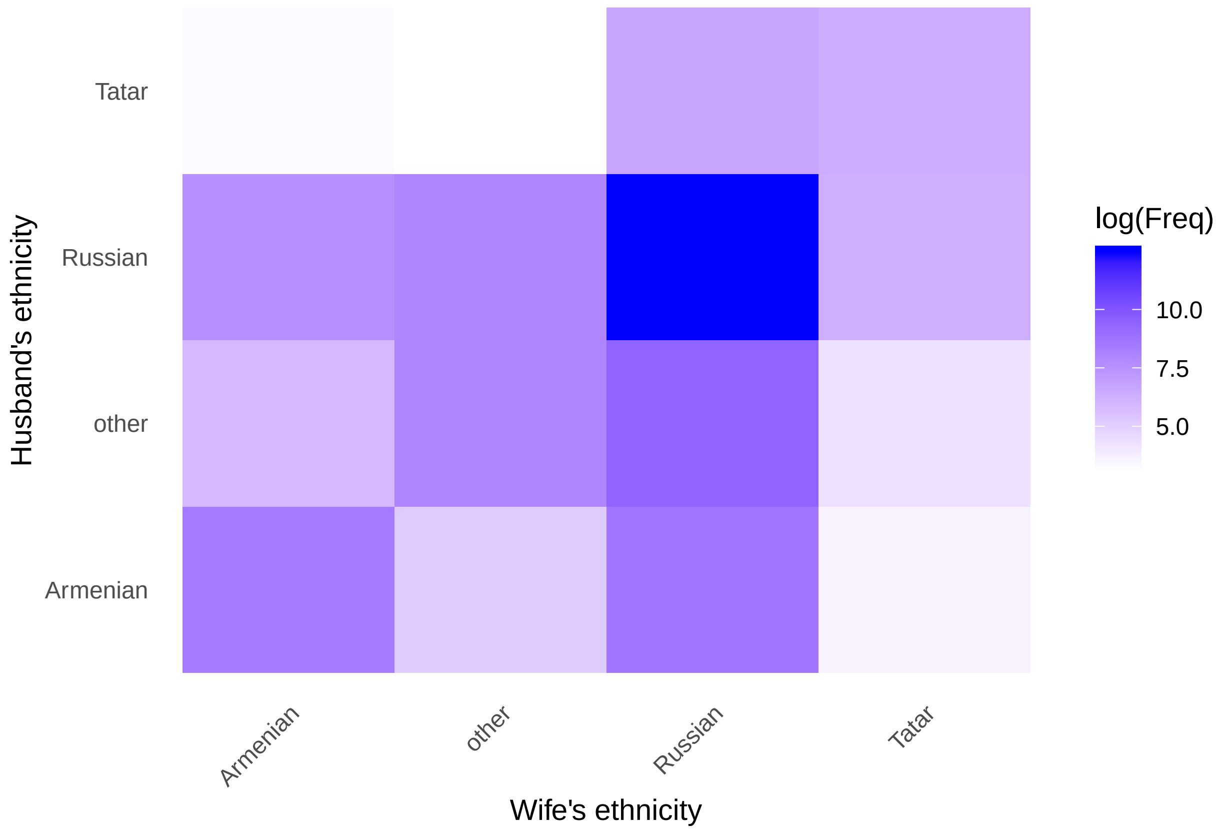
This is Figure 1 in the paper.

```
# function to produce heat maps
heatMap <- function(df){
        collapseData(df) %>%
                ggplot(aes(x = ethn.wife, y = ethn.husband, fill = log(Freq))) +
                geom_tile() +
                xlab("Wife's ethnicity") +
                ylab("Husband's ethnicity") +
                        scale_fill_gradient(low = "white", high = "blue") +
                        theme_classic() +
                        theme(axis.text.x = element_text(angle = 45, hjust = 1),
                              axis.line = element_blank(),
                              axis.ticks = element_blank())
}

# combine data tables for six cities into a list
sixCities <- list(Moscow, Rostov, Kazan, Ufa, Vladikavkaz, Makhachkala)
# assigning names to the elements
names(sixCities) <- c("Moscow", "Rostov", "Kazan", "Ufa", "Vladikavkaz", "Makhachkala")

sixCities %>% map(heatMap)
```

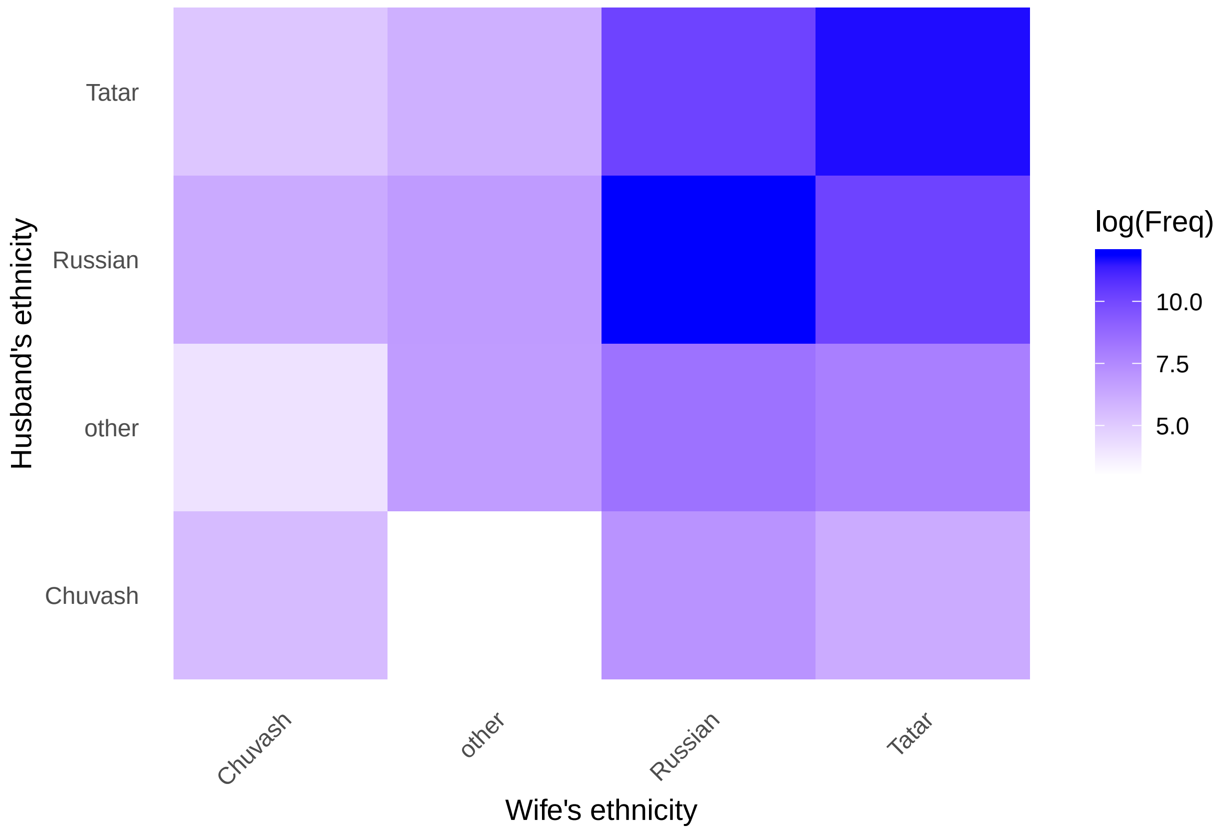
```
## $Moscow
```


Ukrainian
Tatar
Russian
other
Jewish
Armenian
Husband's ethnicity
Armenian
Jewish
other
Russian
Tatar
Ukrainian
Wife's ethnicity
log(Freq)
12
9
6

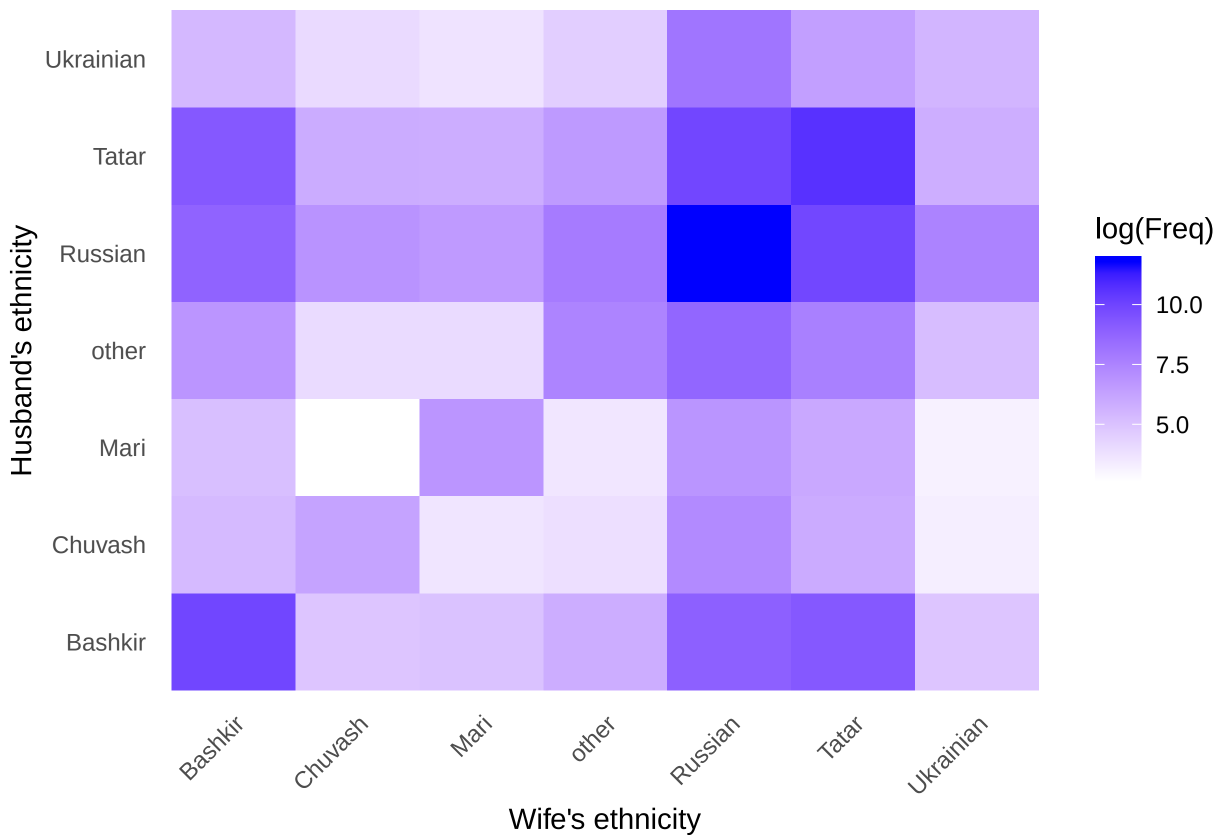
```
##
## $Rostov
```


Tatar
Russian
other
Armenian
Husband's ethnicity
log(Freq)
10.0
7.5
5.0
Armenian
other
Russian
Tatar
Wife's ethnicity

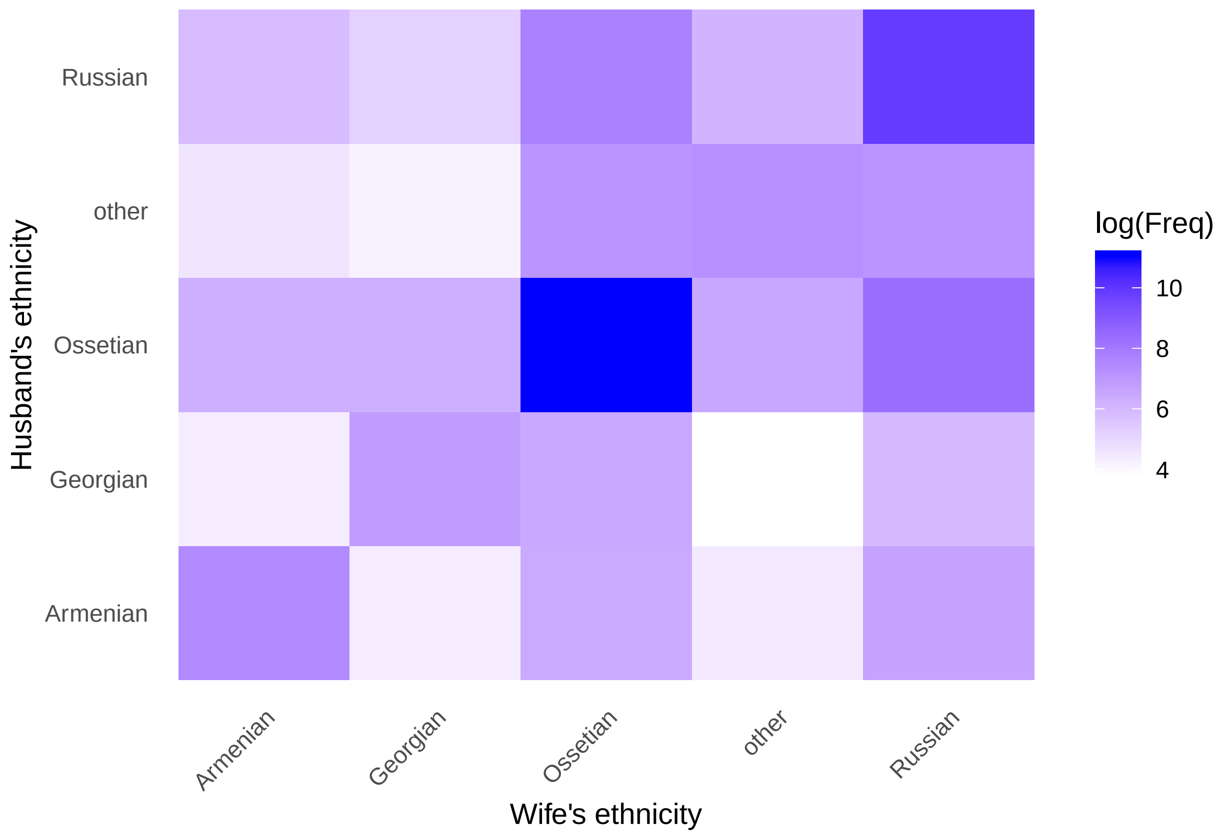
```
##
## $Kazan
```


Tatar
Russian
other
Chuvash
Husband's ethnicity
Chuvash
other
Russian
Tatar
Wife's ethnicity
log(Freq)
10.0
7.5
5.0

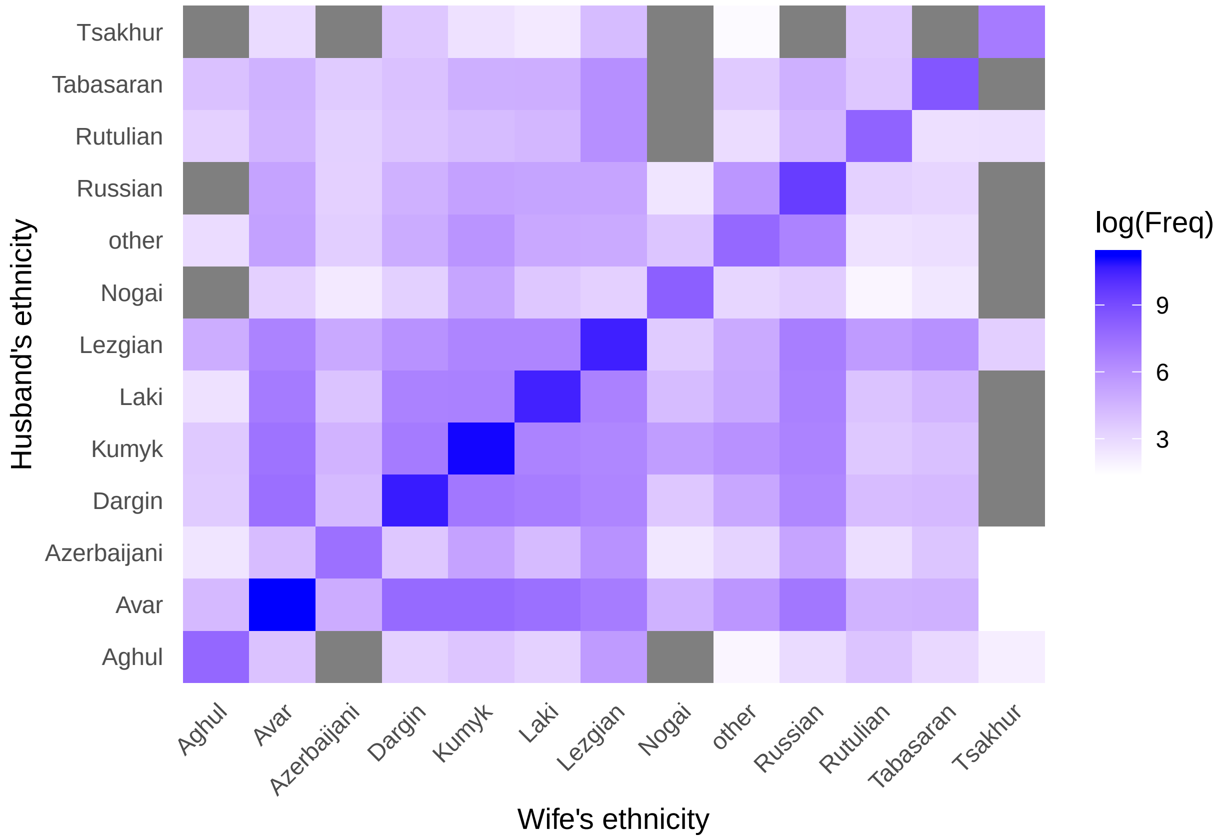
```
##
## $Ufa
```




```
##
## $Vladikavkaz
```




```
##
## $Makhachkala
```



Percent by ethnic group for men and women

See Table 1 in the paper.

Distribution of cases by ethnic group, separately for women and men.

```
# Functions to summarise percentages by ethnic group for women and men

percentByGroupWife <- function(df){
        collapseData(df) %>%
                group_by(ethn.wife) %>%
                summarise(
                        n = sum(Freq)
                ) %>%
                mutate(perc = n / sum(n) * 100)
}

percentByGroupHusband <- function(df){
        collapseData(df) %>%
                group_by(ethn.husband) %>%
                summarise(
                        n = sum(Freq)
                ) %>%
                mutate(perc = n / sum(n) * 100)
}
```

Percentages for women.

```r
sixCities %>% map(percentByGroupWife)
```

```
## $Moscow
## # A tibble: 6 x 3
##   ethn.wife        n   perc
##   <chr>        <dbl>  <dbl>
## 1 Armenian      3930  0.140
## 2 Jewish       12656  0.452
## 3 other        79646  2.85
## 4 Russian    2673121 95.5
## 5 Tatar        25738  0.919
## 6 Ukrainian     4350  0.155
##
## $Rostov
## # A tibble: 4 x 3
##   ethn.wife       n   perc
##   <chr>       <dbl>  <dbl>
## 1 Armenian     7156  2.44
## 2 other        6389  2.18
## 3 Russian    278585 94.9
## 4 Tatar        1282  0.437
##
## $Kazan
## # A tibble: 4 x 3
##   ethn.wife       n   perc
##   <chr>       <dbl>  <dbl>
## 1 Chuvash      1029  0.313
## 2 other        2275  0.692
## 3 Russian    174371 53.0
## 4 Tatar      151214 46.0
##
## $Ufa
## # A tibble: 7 x 3
##   ethn.wife       n   perc
##   <chr>       <dbl>  <dbl>
## 1 Bashkir     39342 13.1
## 2 Chuvash      2122  0.707
## 3 Mari         2287  0.762
## 4 other        5667  1.89
## 5 Russian    170522 56.8
## 6 Tatar       77160 25.7
## 7 Ukrainian    2862  0.954
##
## $Vladikavkaz
## # A tibble: 5 x 3
##   ethn.wife      n  perc
##   <chr>      <dbl> <dbl>
## 1 Armenian    2829  2.82
## 2 Georgian    1841  1.83
## 3 Ossetian   66814 66.5
## 4 other       2716  2.70
## 5 Russian    26244 26.1
##
```

```
## $Makhachkala
## # A tibble: 13 x 3
##    ethn.wife       n   perc
##    <chr>       <dbl>  <dbl>
##  1 Aghul        2936  0.875
##  2 Avar        79603 23.7
##  3 Azerbaijani  2354  0.702
##  4 Dargin      49991 14.9
##  5 Kumyk       74005 22.1
##  6 Laki        42517 12.7
##  7 Lezgian     44795 13.4
##  8 Nogai        4129  1.23
##  9 other        4119  1.23
## 10 Russian     20164  6.01
## 11 Rutulian     3689  1.10
## 12 Tabasaran    6051  1.80
## 13 Tsakhur      1149  0.342
```

Percentages for men.

```
sixCities %>% map(percentByGroupHusband)
```

```
## $Moscow
## # A tibble: 6 x 3
##   ethn.husband       n   perc
##   <chr>          <dbl>  <dbl>
## 1 Armenian       14709  0.525
## 2 Jewish         19143  0.684
## 3 other         139628  4.99
## 4 Russian      2559553 91.4
## 5 Tatar          29708  1.06
## 6 Ukrainian      36700  1.31
##
## $Rostov
## # A tibble: 4 x 3
##   ethn.husband      n   perc
##   <chr>         <dbl>  <dbl>
## 1 Armenian      10626  3.62
## 2 other         15313  5.22
## 3 Russian      265987 90.7
## 4 Tatar          1486  0.506
##
## $Kazan
## # A tibble: 4 x 3
##   ethn.husband      n   perc
##   <chr>         <dbl>  <dbl>
## 1 Chuvash        2071  0.630
## 2 other          8349  2.54
## 3 Russian      170172 51.7
## 4 Tatar        148297 45.1
##
## $Ufa
## # A tibble: 7 x 3
##   ethn.husband      n   perc
##   <chr>         <dbl>  <dbl>
```

```
## 1 Bashkir       39504 13.2
## 2 Chuvash        2630  0.877
## 3 Mari           2512  0.837
## 4 other         11226  3.74
## 5 Russian      163826 54.6
## 6 Tatar         75654 25.2
## 7 Ukrainian      4610  1.54
##
## $Vladikavkaz
## # A tibble: 5 x 3
##   ethn.husband     n  perc
##   <chr>        <dbl> <dbl>
## 1 Armenian      3326  3.31
## 2 Georgian      2132  2.12
## 3 Ossetian     67945 67.6
## 4 other         4045  4.03
## 5 Russian      22996 22.9
##
## $Makhachkala
## # A tibble: 13 x 3
##    ethn.husband     n   perc
##    <chr>        <dbl>  <dbl>
##  1 Aghul         3058  0.911
##  2 Avar         83075 24.8
##  3 Azerbaijani   2795  0.833
##  4 Dargin       50893 15.2
##  5 Kumyk        73408 21.9
##  6 Laki         42424 12.6
##  7 Lezgian      44316 13.2
##  8 Nogai         3944  1.18
##  9 other         4367  1.30
## 10 Russian      15718  4.68
## 11 Rutulian      3924  1.17
## 12 Tabasaran     6299  1.88
## 13 Tsakhur       1281  0.382
```

Percent married within group

See Table 1 in the paper.

The tables below show the percentages married within group for the ethnic groups in six cities, separately for men and women (locally born women only).

```
# a function to produce tables with % married within group
marriedWithin <- function(df){
        # number of women
        df1 <- df %>%
                group_by(ethn.wife) %>%
                summarise(
                        nWomen = sum(Freq)
                )
        # number of men
        df2 <- df %>%
                group_by(ethn.husband) %>%
```

```
            summarise(
                    nMen = sum(Freq)
            ) %>%
            rename(ethn.wife = ethn.husband)

    # number of people married within their group

    df3 <- df %>%
            filter(ethn.wife == ethn.husband) %>%
            group_by(ethn.wife) %>%
            summarise(
                    nWithin = sum(Freq)
            )

    # put it all together

    df1 %>%
            left_join(df2, by = c("ethn.wife")) %>%
            left_join(df3, by = c("ethn.wife")) %>%
            mutate(percWomenWithin = nWithin / nWomen * 100) %>%
            mutate(percMenWithin = nWithin / nMen * 100) %>%
            rename(ethnicity = ethn.wife) %>%
            select(-nWithin) %>%
            select(ethnicity, percWomenWithin, nWomen, percMenWithin, nMen) %>%
            filter(ethnicity != "other")
}

sixCities %>% map(marriedWithin)
```

```
## $Moscow
## # A tibble: 5 x 5
##   ethnicity percWomenWithin  nWomen percMenWithin    nMen
##   <chr>               <dbl>   <dbl>         <dbl>   <dbl>
## 1 Armenian             57.4    3930         15.3    14709
## 2 Jewish               55.0   12656         36.3    19143
## 3 Russian              94.0 2673121         98.2  2559553
## 4 Tatar                58.4   25738         50.6    29708
## 5 Ukrainian            17.0    4350          2.02   36700
##
## $Rostov
## # A tibble: 3 x 5
##   ethnicity percWomenWithin nWomen percMenWithin   nMen
##   <chr>               <dbl>  <dbl>         <dbl>  <dbl>
## 1 Armenian             66.1   7156          44.5  10626
## 2 Russian              93.4 278585          97.9 265987
## 3 Tatar                46.3   1282          40.0   1486
##
## $Kazan
## # A tibble: 3 x 5
##   ethnicity percWomenWithin nWomen percMenWithin   nMen
##   <chr>               <dbl>  <dbl>         <dbl>  <dbl>
## 1 Chuvash              26.1   1029          13.0   2071
## 2 Russian              81.4 174371          83.4 170172
## 3 Tatar                80.1 151214          81.7 148297
```

```
##
## $Ufa
## # A tibble: 6 x 5
##   ethnicity percWomenWithin nWomen percMenWithin   nMen
##   <chr>               <dbl>  <dbl>         <dbl>  <dbl>
## 1 Bashkir             52.5   39342         52.2   39504
## 2 Chuvash             24.4    2122         19.7    2630
## 3 Mari                39.8    2287         36.2    2512
## 4 Russian             76.8  170522         79.9  163826
## 5 Tatar               55.9   77160         57.0   75654
## 6 Ukrainian            8.91   2862          5.53   4610
##
## $Vladikavkaz
## # A tibble: 4 x 5
##   ethnicity percWomenWithin nWomen percMenWithin  nMen
##   <chr>               <dbl>  <dbl>         <dbl> <dbl>
## 1 Armenian             61.9   2829          52.7  3326
## 2 Georgian             52.6   1841          45.4  2132
## 3 Ossetian             92.7  66814          91.2 67945
## 4 Russian              74.9  26244          85.5 22996
##
## $Makhachkala
## # A tibble: 12 x 5
##    ethnicity   percWomenWithin nWomen percMenWithin  nMen
##    <chr>                 <dbl>  <dbl>         <dbl> <dbl>
##  1 Aghul                  86.3   2936          82.9  3058
##  2 Avar                   92.4  79603          88.5 83075
##  3 Azerbaijani            72.9   2354          61.4  2795
##  4 Dargin                 89.9  49991          88.3 50893
##  5 Kumyk                  91.4  74005          92.2 73408
##  6 Laki                   88.3  42517          88.5 42424
##  7 Lezgian                88.3  44795          89.2 44316
##  8 Nogai                  86.1   4129          90.1  3944
##  9 Russian                71.4  20164          91.6 15718
## 10 Rutulian               80.8   3689          75.9  3924
## 11 Tabasaran              85.4   6051          82.0  6299
## 12 Tsakhur                94.6   1149          84.9  1281
```

Odds ratios for ethnic endogamy across six cities

See Table 1 in the paper.

```
# a function to collapse data tables to 2x2 tables
collapse2x2 <- function(df, ethn){
        df %>%
                mutate(ethn2F = ifelse(ethn.wife == ethn, 1, 0)) %>%
                mutate(ethn2M = ifelse(ethn.husband == ethn, 1, 0)) %>%
                group_by(ethn2F, ethn2M) %>%
                summarise(
                        Freq = sum(Freq)
                )
}

# a function to calculate log odds ratios with 95% confidence intervals
```

```
oddsRatio <- function(df){
     # freq is a vector of counts
     freq <- df %>% pull(Freq)
     # a is the number of marriages where both husband and wife are not from group i
     # as.numeric() added to avoid integer overflow
     a <- as.numeric(freq[1])
     # b and c are the number of intermarriages
     b <- as.numeric(freq[2])
     c <- as.numeric(freq[3])
     # d is the number of endogamous intermarriages for group i
     d <- as.numeric(freq[4])
     # calculating log oddsratio
     logOR <- log((a*d) / (b*c))
     # calculating the standard error for log odds ratio (see https://www.ncbi.nlm.nih.gov/pmc/articles/
     se <- sqrt(1/a + 1/b + 1/c + 1/d)
     # calculating 95% confidence intervals
     lowerCI <- logOR - 1.96*se
     upperCI <- logOR + 1.96*se
     return(c(logOR, lowerCI, upperCI))
}

# a function to loop over all ethnic groups and age groups to produce a data frame with log ORs and CIs
ethnOR <- function(df){
        # Initialise a data frame  for the results
        n  <- length(levels(df$ethn.wife)) * 3 # 3 age groups
        results <- data.frame(city = rep(NA, n),
                              ethnGroup = rep(NA, n),
                              ageGroup = rep(NA, n),
                              logOR = rep(NA, n),
                              lowerCI = rep(NA, n),
                              upperCI = rep(NA, n))
        k <- 1
        for (i in levels(as.factor(df$ethn.wife))) {
                for (j in levels(as.factor(df$age.wife))){
                        dfNew <- filter(df, age.wife == j)
                        res <- collapse2x2(dfNew, i) %>%
                                oddsRatio()
                        results[k,1] <- df$city[1]
                        results[k,2] <- i
                        results[k,3] <- j
                        results[k,4] <- res[1]
                        results[k,5] <- res[2]
                        results[k,6] <- res[3]
                        k <- k + 1
                }
        }
        results
}

# same function as above (to produce ORs), but collapsing all age groups together.

ethnOR2 <- function(df){
        # Initialise a data frame  for the results
```

```r
        n  <- length(as.factor(levels(df$ethn.wife)))
        results <- data.frame(city = rep(NA, n),
                              ethnGroup = rep(NA, n),
                              logOR = rep(NA, n),
                              lowerCI = rep(NA, n),
                              upperCI = rep(NA, n))
        k <- 1
        for (i in levels(as.factor(df$ethn.wife))) {
                dfNew <- collapseData(df)
                res <- collapse2x2(dfNew, i) %>%
                        oddsRatio()
                results[k,1] <- df$city[1]
                results[k,2] <- i
                results[k,3] <- res[1]
                results[k,4] <- res[2]
                results[k,5] <- res[3]
                k <- k + 1
        }
        results
}

# produce a table with ORS for all cities

sixCities %>%
        map_df(ethnOR2)
```

```
##          city  ethnGroup    logOR  lowerCI  upperCI
## 1      Moscow   Armenian 5.705479 5.639857 5.771101
## 2      Moscow     Jewish 5.627833 5.588552 5.667114
## 3      Moscow      other 3.966289 3.950347 3.982232
## 4      Moscow    Russian 3.332329 3.319777 3.344881
## 5      Moscow      Tatar 5.576025 5.546401 5.605649
## 6      Moscow  Ukrainian 2.755451 2.675678 2.835224
## 7      Rostov   Armenian 4.530279 4.474947 4.585610
## 8      Rostov      other 3.045851 2.993544 3.098159
## 9      Rostov    Russian 3.119585 3.083279 3.155890
## 10     Rostov      Tatar 5.641520 5.513571 5.769470
## 11      Kazan    Chuvash 4.159571 4.013013 4.306130
## 12      Kazan      other 3.312038 3.224746 3.399331
## 13      Kazan    Russian 2.977552 2.959879 2.995225
## 14      Kazan      Tatar 3.110666 3.092584 3.128748
## 15        Ufa    Bashkir 2.648479 2.623760 2.673197
## 16        Ufa    Chuvash 3.811524 3.703621 3.919428
## 17        Ufa       Mari 4.805135 4.708067 4.902204
## 18        Ufa      other 2.645712 2.586156 2.705268
## 19        Ufa    Russian 2.271800 2.254981 2.288619
## 20        Ufa      Tatar 2.001979 1.983535 2.020423
## 21        Ufa  Ukrainian 1.883285 1.751248 2.015323
## 22 Vladikavkaz   Armenian 4.597734 4.506958 4.688510
## 23 Vladikavkaz   Georgian 4.530661 4.422453 4.638868
## 24 Vladikavkaz   Ossetian 4.079002 4.038560 4.119445
## 25 Vladikavkaz      other 3.690593 3.605869 3.775318
## 26 Vladikavkaz    Russian 4.148901 4.104361 4.193441
## 27 Makhachkala      Aghul 8.292627 8.156925 8.428330
```

```
## 28 Makhachkala        Avar  5.747753  5.714519  5.780987
## 29 Makhachkala Azerbaijani  6.721803  6.612975  6.830630
## 30 Makhachkala      Dargin  6.028406  5.989666  6.067145
## 31 Makhachkala       Kumyk  6.166029  6.129315  6.202743
## 32 Makhachkala        Laki  6.098798  6.057875  6.139722
## 33 Makhachkala     Lezgian  6.112423  6.071834  6.153013
## 34 Makhachkala       Nogai  8.565135  8.432380  8.697889
## 35 Makhachkala       other  5.525981  5.449300  5.602662
## 36 Makhachkala     Russian  6.386929  6.324836  6.449023
## 37 Makhachkala    Rutulian  7.292368  7.188555  7.396180
## 38 Makhachkala   Tabasaran  7.435590  7.343424  7.527756
## 39 Makhachkala     Tsakhur 10.315557 10.023480 10.607634
```

```
# produce a data frame with log ORs for all six cities; remove missing values;
# reorder levels for age groups

ORData <- sixCities %>%
        map_df(ethnOR) %>%
        filter(is.finite(logOR)) %>%
        mutate(ageGroup = fct_relevel(ageGroup, "16-35", "36-50", ">50"))

head(ORData)
```

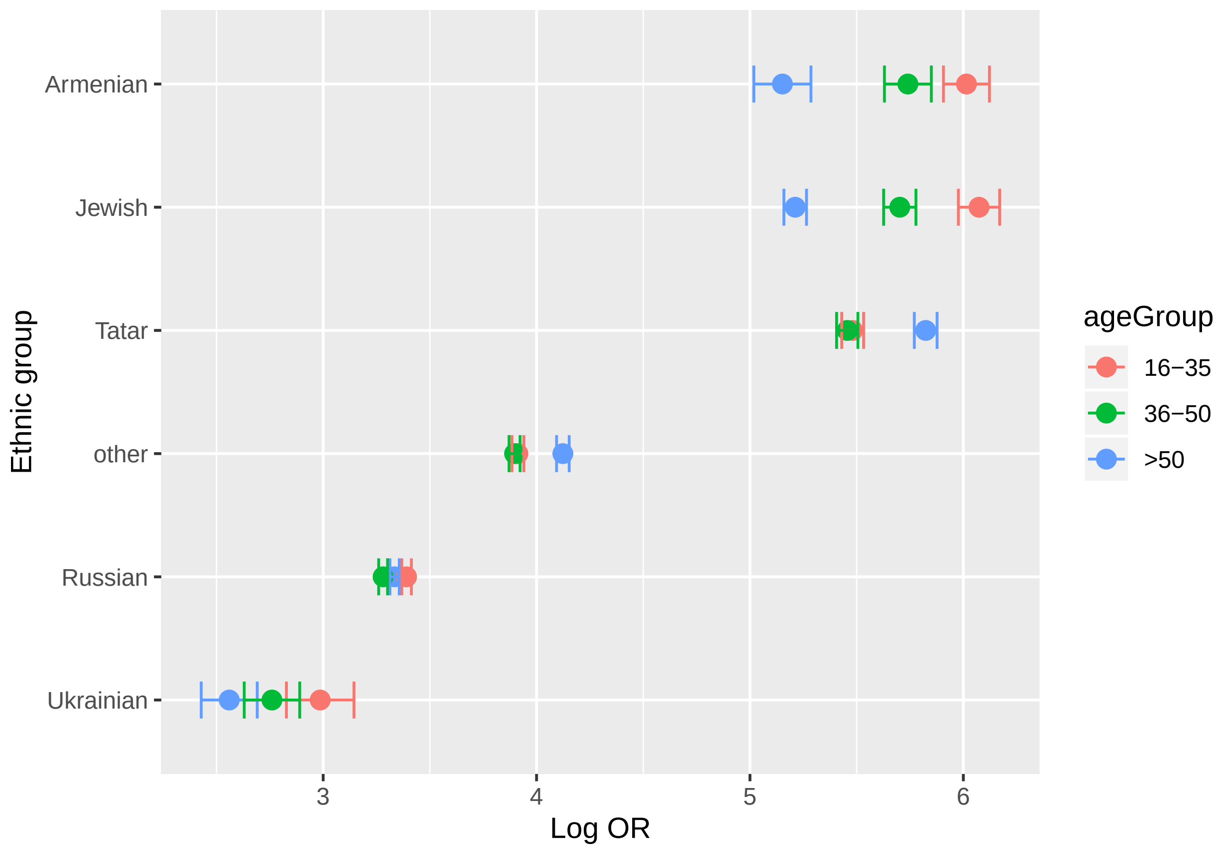
```
##     city ethnGroup ageGroup    logOR  lowerCI  upperCI
## 1 Moscow  Armenian      >50 5.152011 5.018191 5.285831
## 2 Moscow  Armenian    16-35 6.014574 5.906776 6.122372
## 3 Moscow  Armenian    36-50 5.740118 5.630176 5.850060
## 4 Moscow    Jewish      >50 5.212030 5.159099 5.264961
## 5 Moscow    Jewish    16-35 6.073428 5.976779 6.170078
## 6 Moscow    Jewish    36-50 5.702149 5.626474 5.777825
```

```
# a function to produce dot plots with confidence intervals

plotOR <- function(x){
        ORData %>%
        filter(city == x) %>%
        ggplot(aes(x = fct_reorder(ethnGroup, logOR), y = logOR, colour = ageGroup)) +
        geom_point(size = 3) +
        geom_errorbar(aes(ymin = lowerCI, ymax = upperCI), width= 0.3) +
        coord_flip() +
        xlab("Ethnic group") +
        ylab("Log OR")
}
```

See Figure 2 in the paper.

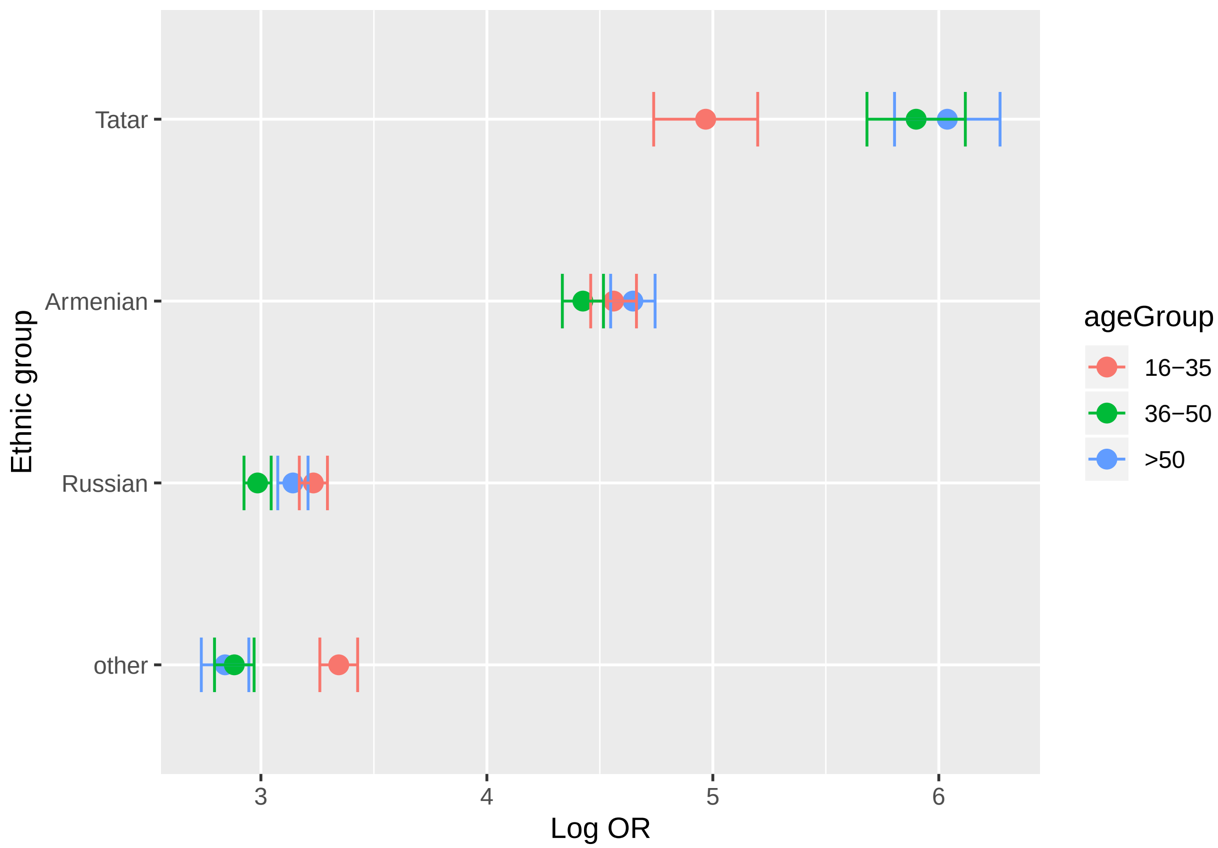
Moscow

```
plotOR("Moscow")
```



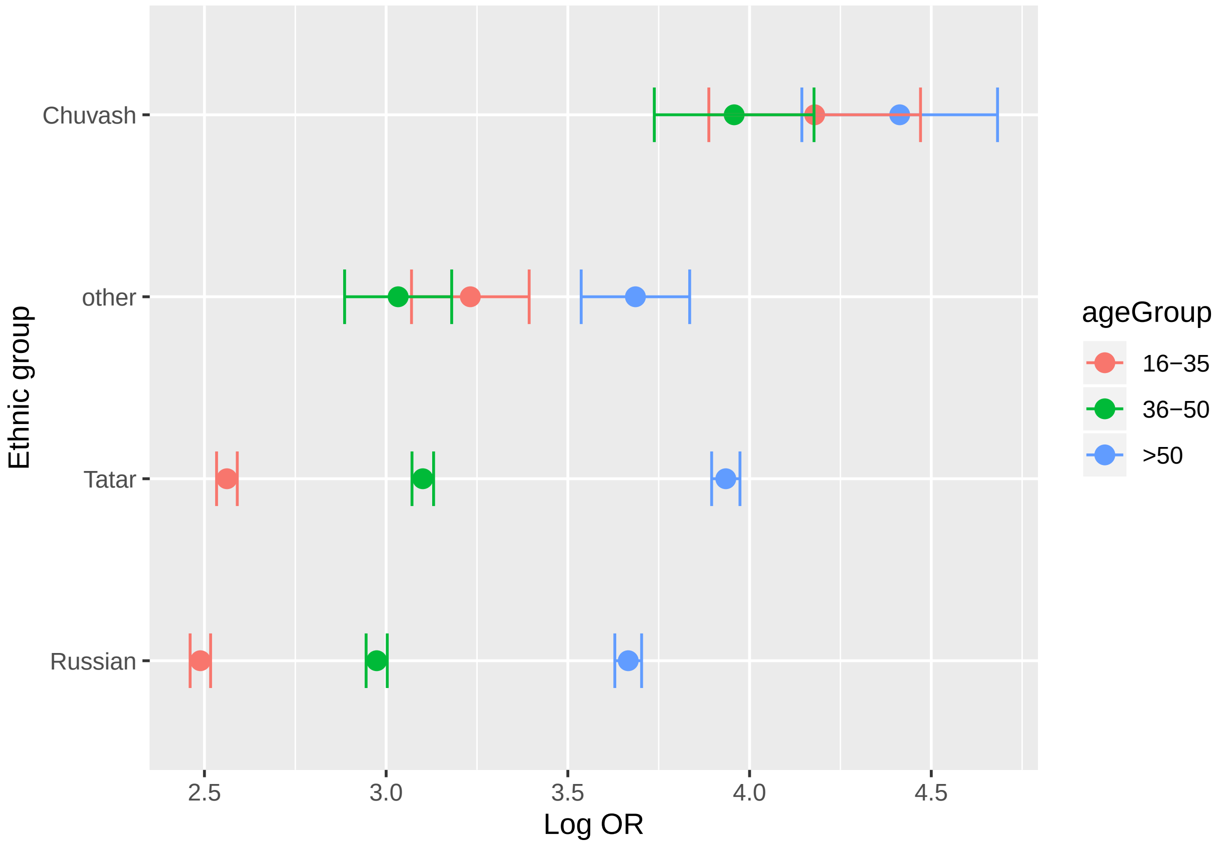
Rostov

```
plotOR("Rostov")
```



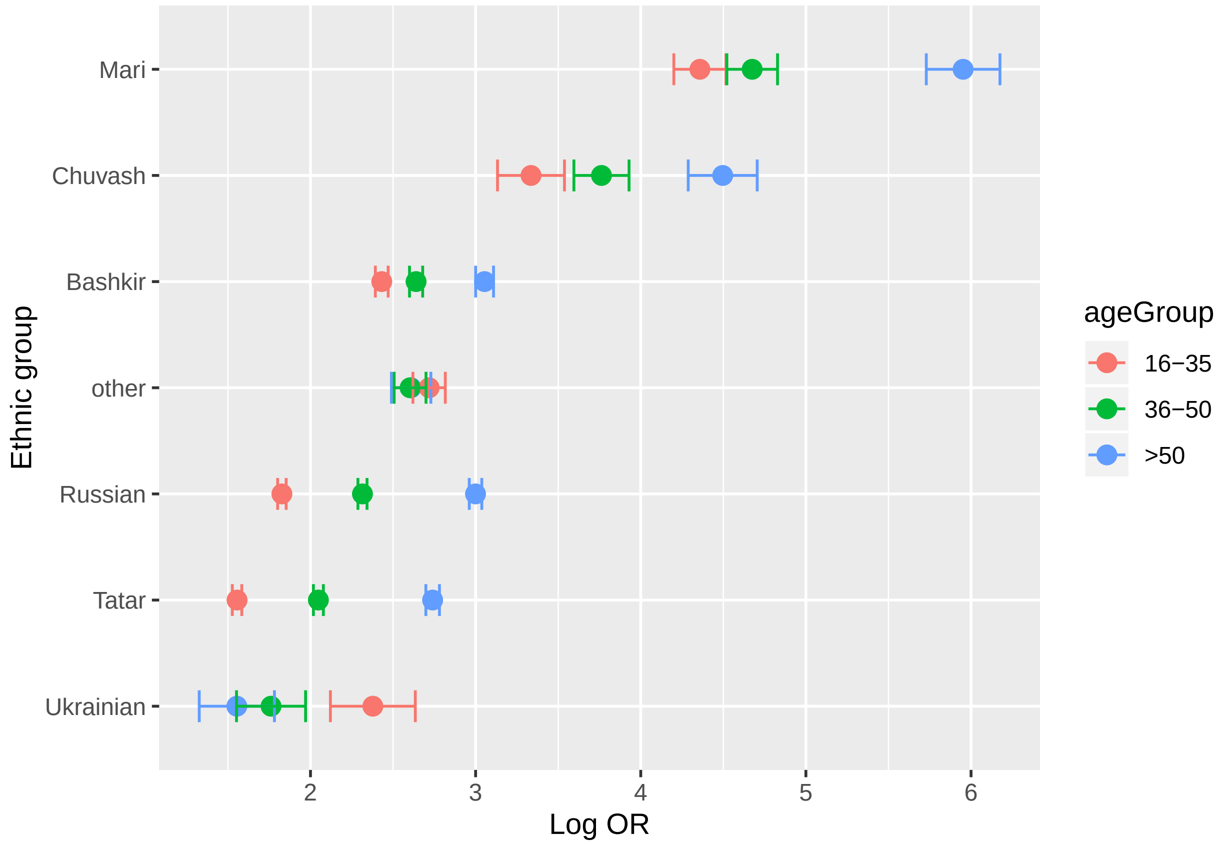
Kazan

```
plotOR("Kazan")
```



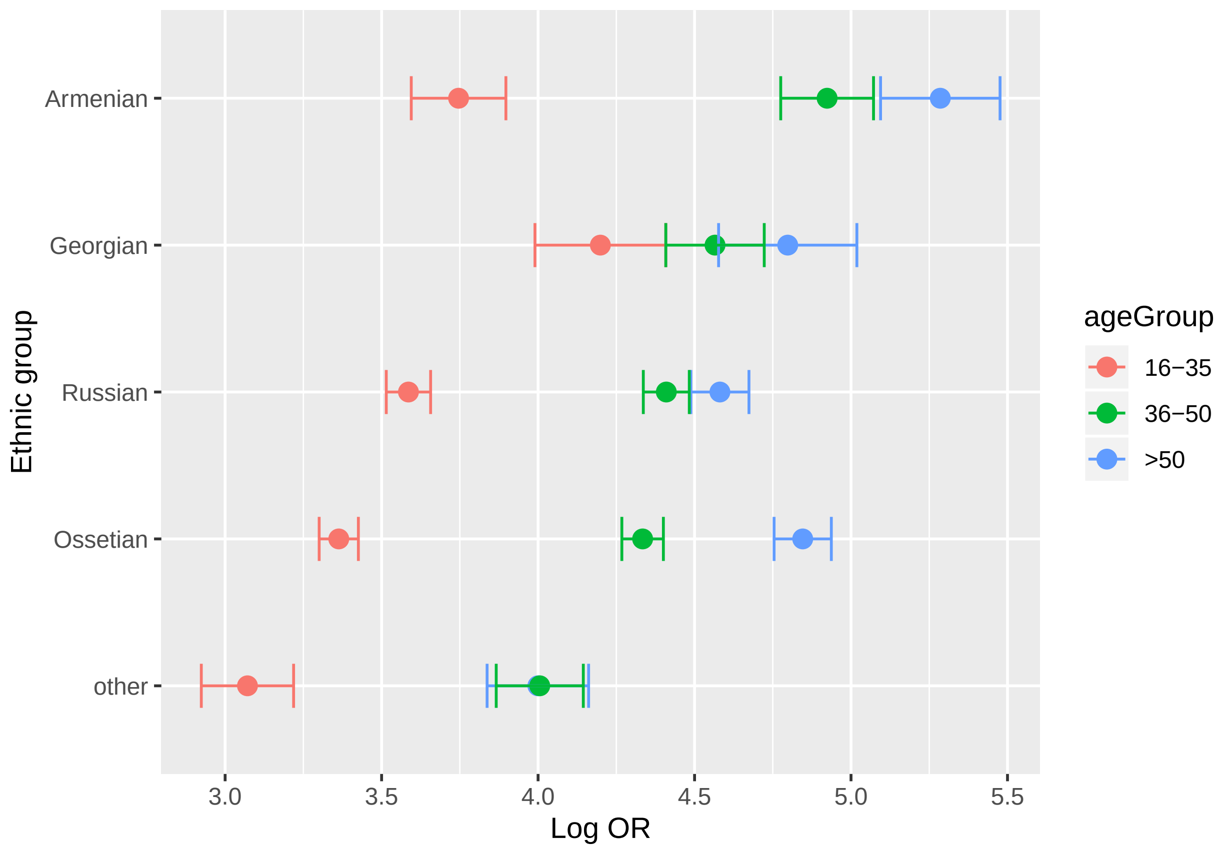
Ufa

```
plotOR("Ufa")
```



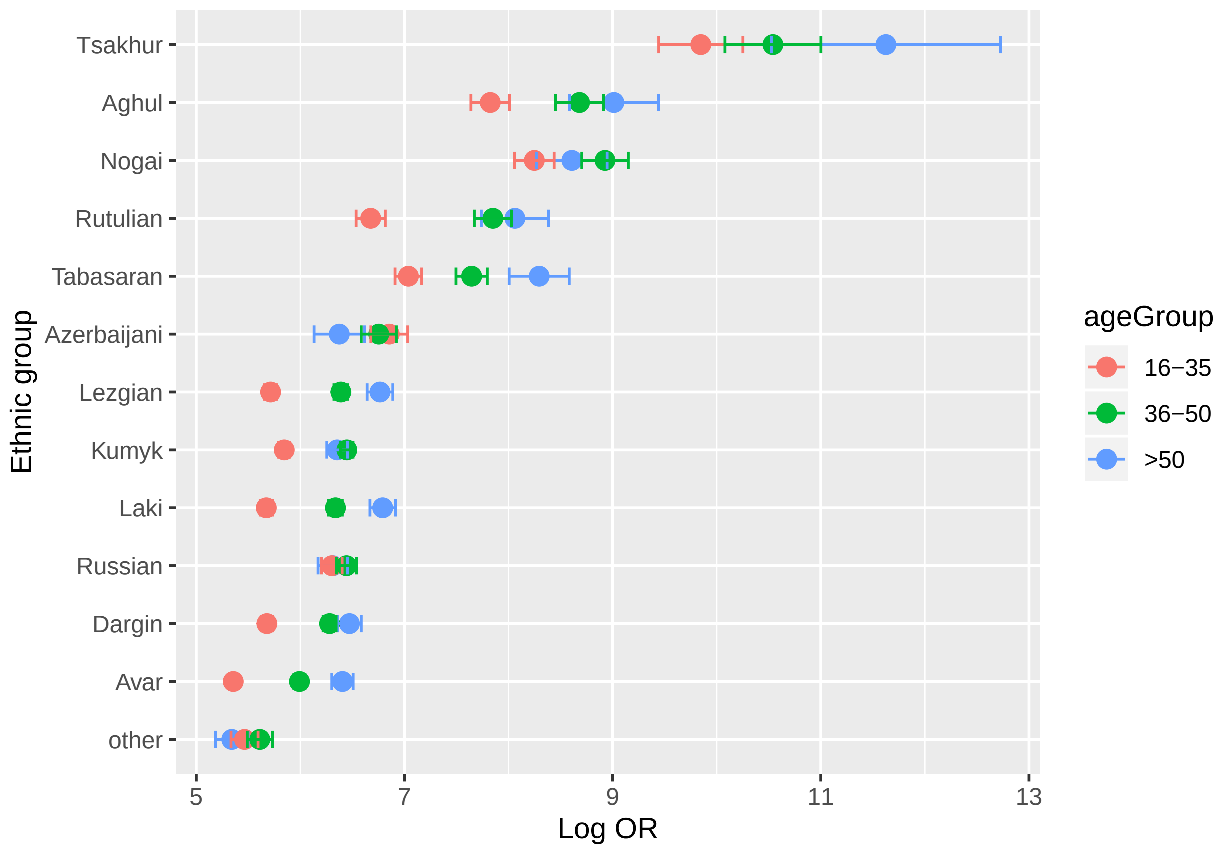
Vladikavkaz

```
plotOR("Vladikavkaz")
```



Makhachkala

```r
plotOR("Makhachkala")
```



Symmetrical odds ratios

Symmetrical odds ratios are odds ratios that involve a pair of ethnic groups. For example, the odds of a Russian woman marrying a Russian rather than a Tatar man divided by the odds of a Tatar woman marrying a Russian rather than a Tatar man. See an application in the social mobility research in Bukodi and Goldthorpe. (2019). Social Mobility and Education in Britain, ch. 4.

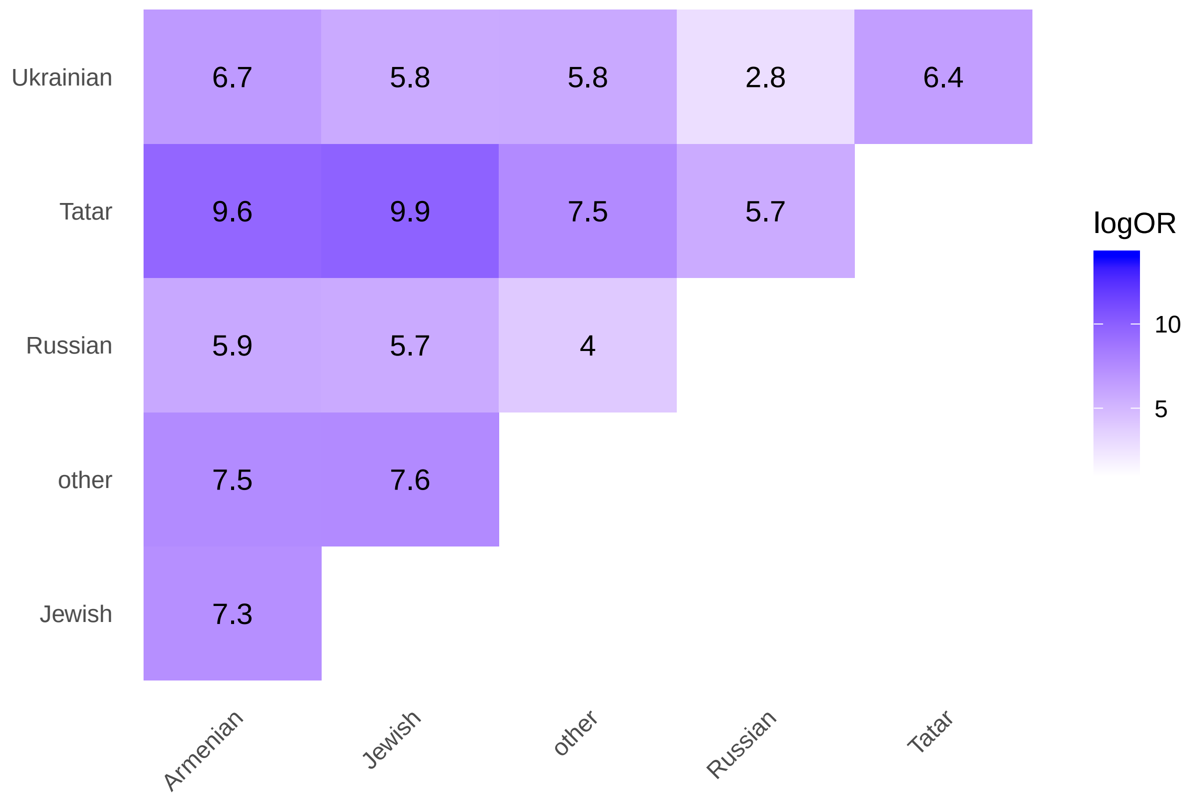
See Appendix B in the paper.

```
# For each city, we want to loop over all pairs of ethnic groups and calculate the odds ratio.

# A function to produce a data frame of log odds ratios for each pair of ethnicities.

symmOR <- function(df){
        # create an empty data frame to store results
        result <- tibble(ethn.wife = character(),
                    ethn.husband = character(),
                    logOR = numeric())
        k <- 1
        # loop over ethnicities of wives and husbands
        for (i in levels(as.factor(df$ethn.wife))) {
                for (j in levels(as.factor(df$ethn.husband))){
                        if (i != j) {
                                # calculate and save log OR
                                res <- df %>%
                                        filter(ethn.wife == i | ethn.wife == j) %>%
```

```
                                        filter(ethn.husband == i | ethn.husband == j) %>%
                                        collapseData() %>%
                                        oddsRatio()
                                result[k, 1] <- i
                                result[k, 2] <- j
                                result[k, 3] <- res[1]
                                k <- k + 1
                        }
                }
        }
        # remove duplicates (ORs are symmetrical)
        result <- result %>%
                mutate(key = paste0(pmin(ethn.wife, ethn.husband),
                                    pmax(ethn.wife, ethn.husband), sep = "")) %>%
                distinct(key, .keep_all = TRUE) %>%
                select(-key) %>%
                # re-order factors as in the original data
                mutate(ethn.wife = factor(ethn.wife, levels = levels(as.factor(df$ethn.wife)))) %>%
                mutate(ethn.husband = factor(ethn.husband, levels = levels(as.factor(df$ethn.husband)))
}

# a function to plot the symmetrical odds ratios

plotSymmOR <- function(df){
        df %>%
                symmOR() %>%
                ggplot(aes(x = ethn.wife, y = ethn.husband, fill = logOR)) +
                        geom_tile() +
                        geom_text(aes(label = round(logOR, 1))) +
                        xlab("") +
                        ylab("") +
                        scale_fill_gradient(low = "white", high = "blue", limits = c(1, 14)) +
                        theme_classic() +
                        theme(axis.text.x = element_text(angle = 45, hjust = 1),
                              axis.line = element_blank(),
                              axis.ticks = element_blank())
}
```

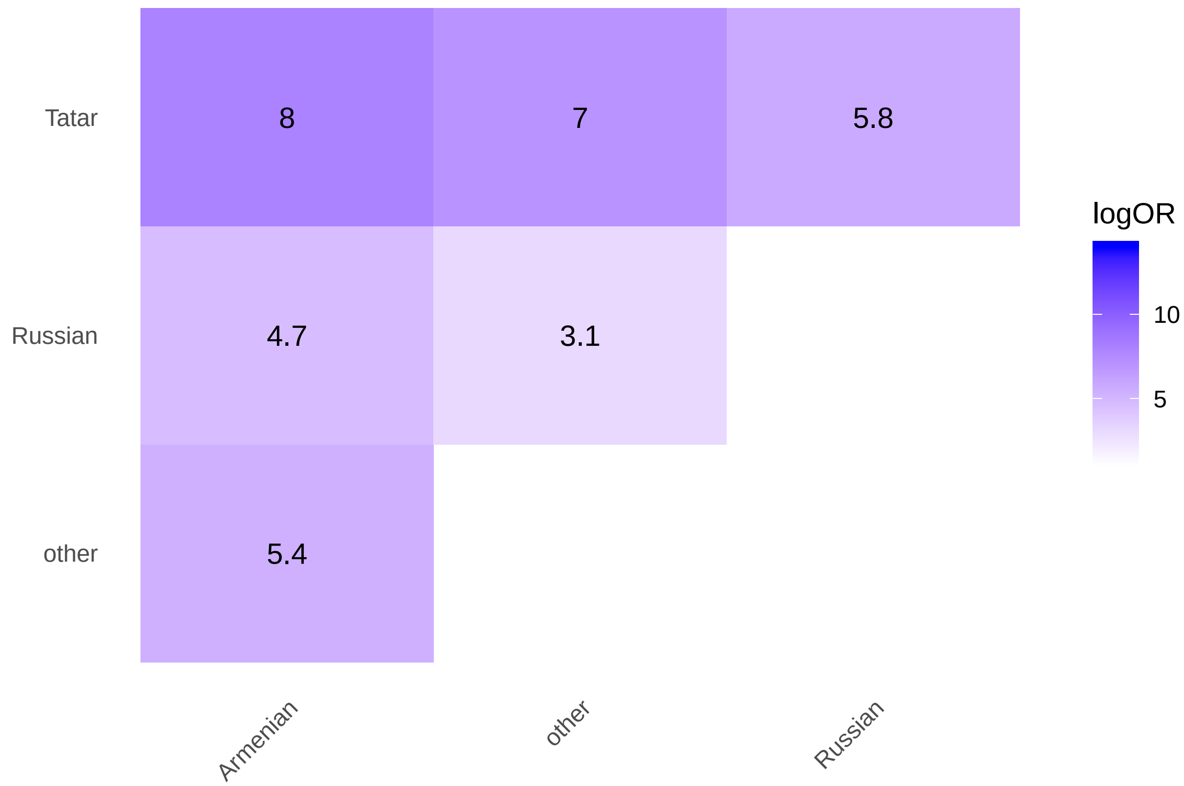
Moscow

```
plotSymmOR(Moscow)
```



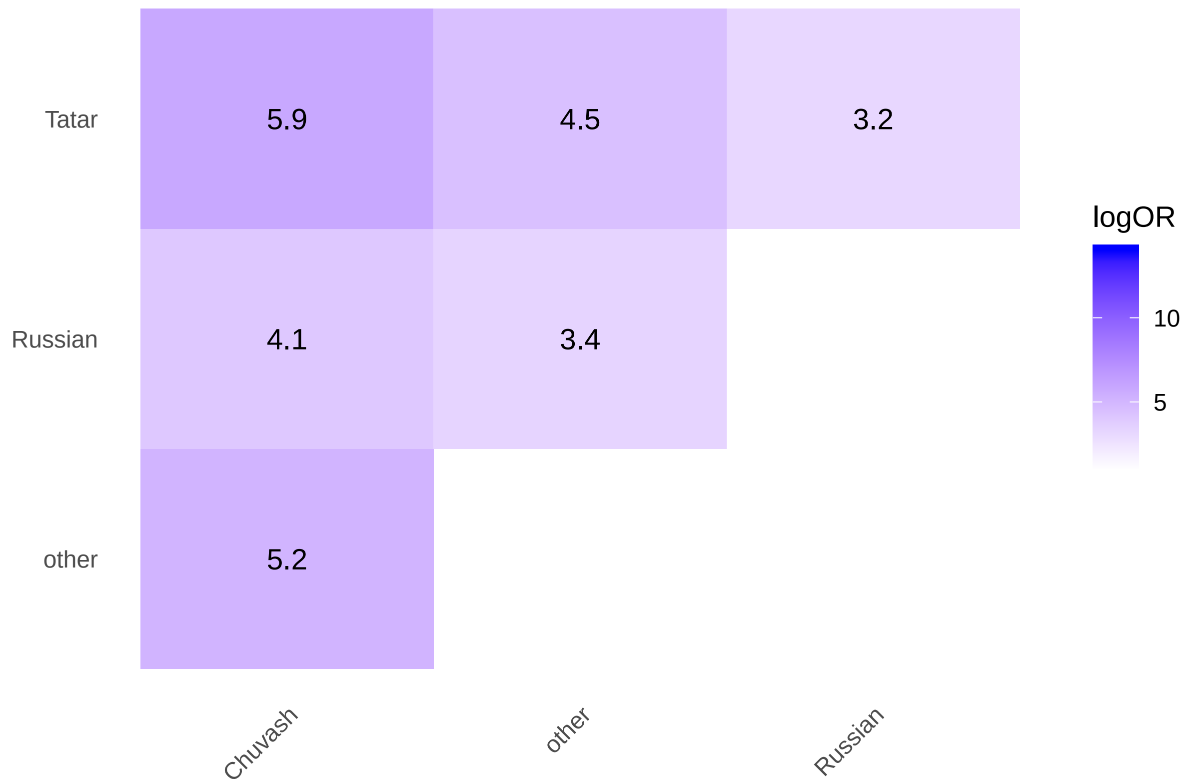
Rostov

```r
plotSymmOR(Rostov)
```



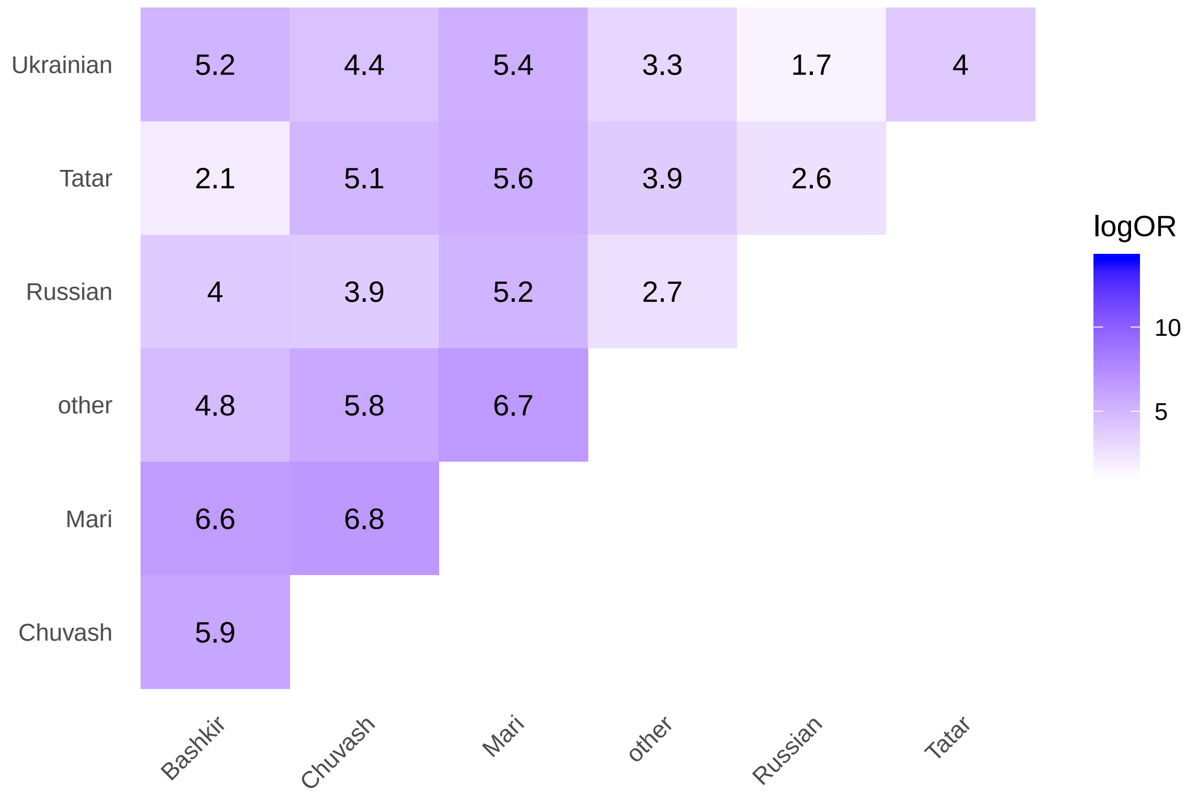
Kazan

```
plotSymmOR(Kazan)
```



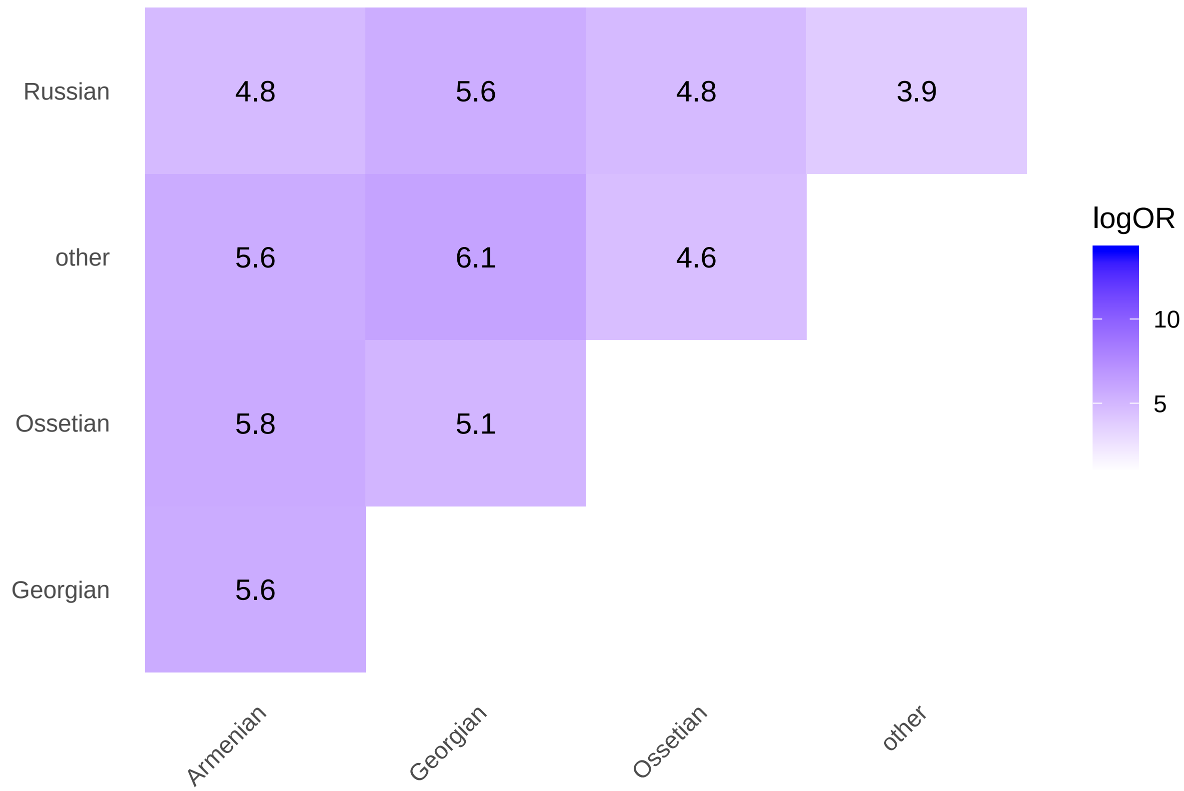
Ufa

```
plotSymmOR(Ufa)
```



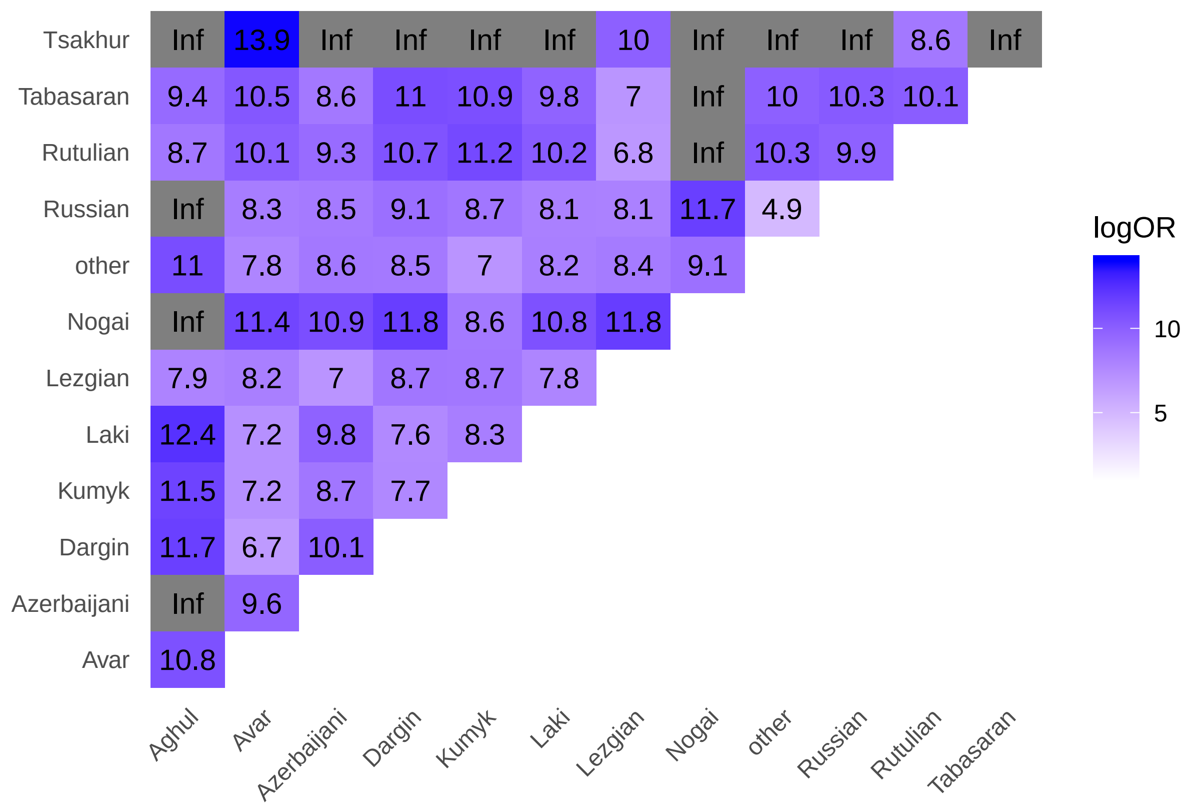
Vladikavkaz

```
plotSymmOR(Vladikavkaz)
```



Makhachkala

```
plotSymmOR(Makhachkala)
```



Functions to fit the models

For each city I do the following:

1) Fit the constant intermarriage rates model (WH + WA + HA).
2) Fit the unidiff model (WA + HA + beta*WH).
3) Compare G2, BIC, dissimilarity index (on the scale from 0 to 100).

```
# a function to fit three models and produce a table with goodness-of-fit statistics
modelIntermarriage <- function(df){
        set.seed(15)
        # fit the models
        # constant
        constant <- gnm(Freq ~ ethn.wife*age.wife + age.wife*ethn.husband + ethn.wife*ethn.husband,
                        family = poisson, data = df)
        # unidiff
        unidiff<- gnm(Freq ~ ethn.wife*age.wife + ethn.husband*age.wife
                      + Mult(Exp(age.wife),
       ethn.wife:ethn.husband), family = poisson, ofInterest = "[.]age.wife",
       data = df, verbose = FALSE)
        # a tibble to store the results, with three rows
        res <- tibble(
                model = c("constant", "unidiff", "unidiff vs. constant"),
                G2 = numeric(3),
                pvalue = numeric(3),
                BIC = numeric(3),
```

```
                diss = numeric(3)
        )
        # populating the tibble
        res$G2 <- c(deviance(constant), deviance(unidiff), lrtest(unidiff, constant)$Chisq[2])
        res$pvalue <- c(NA, NA, lrtest(unidiff, constant)$`Pr(>Chisq)`[2])
        res$BIC <- round(c(BIC(constant), BIC(unidiff), NA))

        # calculating dissimilarity indices
        dissdf <- df %>%
        mutate(fittedConstant = constant$fitted) %>%
        mutate(diffConstant = abs(fittedConstant - Freq)) %>%
        mutate(fittedUnidiff = unidiff$fitted) %>%
        mutate(diffUnidiff = abs(fittedUnidiff - Freq)) %>%
        summarise(
                dissConstant = sum(diffConstant) / (2 * sum(Freq)) * 100,
                dissUnidiff = sum(diffUnidiff) / (2 * sum(Freq)) * 100
        )

        res$diss <- c(dissdf[[1,1]], dissdf[[1,2]], NA)
        return(res)
}

# a function to estimate unidiff contrasts and return coefficients with quasi standard errors

collectUnidiff <- function(df){
        set.seed(15)
        # re-estimate unidiff
        unidiff <- gnm(Freq ~ ethn.wife*age.wife + ethn.husband*age.wife + Mult(Exp(age.wife),
                        ethn.wife:ethn.husband), family = poisson,
                       ofInterest = "[.]age.wife", data = df, verbose = FALSE)
        # get contrasts
        myContrasts <- getContrasts(unidiff, ofInterest(unidiff))
        unidiffContrasts <- tibble(
                ageGroup = levels(as.factor(df$age.wife)),
                coef = myContrasts$qvframe$estimate,
                se = myContrasts$qvframe$quasiSE,
                city = df$city[[1]]) %>%
                mutate(ageGroup = fct_relevel(ageGroup, ">50", "36-50", "16-35"))
      return(unidiffContrasts)
}

# Unidiff models cannot be reliably estimated for Makhachkala because of the sparseness of the data set
# I recode the data for Makhachkala to a smaller number of groups, keeping 6 largest groups only.

Makhachkala <- Makhachkala %>%
        mutate(ethn.wife = fct_recode(ethn.wife,
                                      other = "Tabasaran",
                                      other = "Nogai",
                                      other = "Rutulian",
                                      other = "Aghul",
                                      other = "Azerbaijani",
                                      other = "Tsakhur")) %>%
        mutate(ethn.husband = fct_recode(ethn.husband,
```

```r
                                     other = "Tabasaran",
                                   other = "Nogai",
                                   other = "Rutulian",
                                   other = "Aghul",
                                   other = "Azerbaijani",
                                   other = "Tsakhur")) %>%
        group_by(ethn.wife, ethn.husband, age.wife) %>%
        summarise(
                Freq = sum(Freq)
        ) %>%
        mutate(city = "Makhachkala") %>%
        ungroup()

# Update sixCities
sixCities <- list(Moscow, Rostov, Kazan, Ufa, Vladikavkaz, Makhachkala)
names(sixCities) <- c("Moscow", "Rostov", "Kazan", "Ufa", "Vladikavkaz", "Makhachkala")
```

Models for six cities

Procucing summary statistics for the models and a plot with the unidiff coefficients.

This is Table 2 and Figure 3 in the paper.

For each city, the tables show:

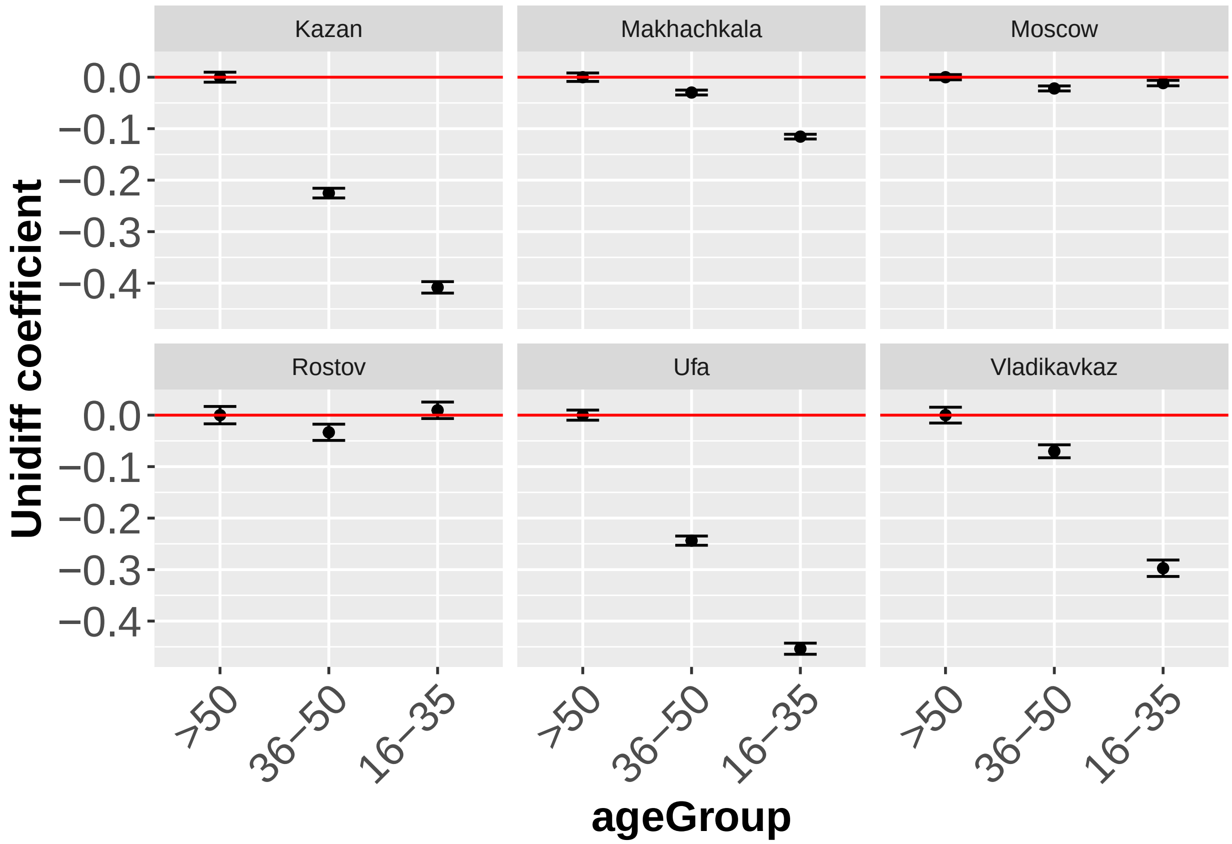
1) the constant intermarriage rate model (the intermarriage rates do not change over time),
2) the unidiff model (the intermarriage rates change over time),
3) the comparison between 2) and 1).

```r
sixCities %>%
        map(modelIntermarriage)
```

```
## $Moscow
## # A tibble: 3 x 5
##   model                   G2   pvalue   BIC   diss
##   <chr>                <dbl>    <dbl> <dbl>  <dbl>
## 1 constant             863.  NA        1962  0.190
## 2 unidiff              824.  NA        1933  0.154
## 3 unidiff vs. constant  38.5  4.27e-9    NA NA
##
## $Rostov
## # A tibble: 3 x 5
##   model                   G2    pvalue   BIC   diss
##   <chr>                <dbl>     <dbl> <dbl>  <dbl>
## 1 constant             200.  NA          673  0.360
## 2 unidiff              184.  NA          664  0.285
## 3 unidiff vs. constant  16.1  0.000319    NA NA
##
## $Kazan
## # A tibble: 3 x 5
##   model                   G2 pvalue   BIC   diss
##   <chr>                <dbl>  <dbl> <dbl>  <dbl>
## 1 constant             3331.     NA  3834  3.01
## 2 unidiff               190.     NA   701  0.300
## 3 unidiff vs. constant 3141.      0    NA NA
```

```
##
## $Ufa
## # A tibble: 3 x 5
##   model                  G2 pvalue   BIC  diss
##   <chr>               <dbl>  <dbl> <dbl> <dbl>
## 1 constant            5513.     NA  6921  4.37
## 2 unidiff             1684.     NA  3102  1.93
## 3 unidiff vs. constant 3829.      0    NA NA
##
## $Vladikavkaz
## # A tibble: 3 x 5
##   model                  G2     pvalue   BIC   diss
##   <chr>               <dbl>      <dbl> <dbl>  <dbl>
## 1 constant            1100. NA          1818  2.84
## 2 unidiff              254. NA           981  0.839
## 3 unidiff vs. constant  846.  2.04e-184    NA NA
##
## $Makhachkala
## # A tibble: 3 x 5
##   model                  G2     pvalue   BIC   diss
##   <chr>               <dbl>      <dbl> <dbl>  <dbl>
## 1 constant            1538. NA          3051  1.67
## 2 unidiff              578. NA          2101  0.640
## 3 unidiff vs. constant  960.  3.56e-209    NA NA
```

```
unidiff.df <- sixCities %>%
        map(collectUnidiff) %>%
        bind_rows()


# plot unidiff contrasts for all six cities
unidiff.df %>%
                ggplot(aes(x = ageGroup, y = coef)) +
                geom_point() +
                geom_errorbar(aes(ymin = coef - 2*se, ymax = coef + 2*se), width = 0.3) +
                geom_hline(yintercept = 0, colour = "red") +
                theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
                ylab("Unidiff coefficient") +
                facet_wrap(~ city) +
                theme(axis.text = element_text(size = 16),
                        axis.title = element_text(size = 16, face="bold"))
```


Kazan
Makhachkala
Moscow
Rostov
Ufa
Vladikavkaz
0.0
−0.1
−0.2
−0.3
−0.4
Unidiff coefficient
>50
36–50
16–35
ageGroup