R-Blatt 4: Visualisierung statistischer Konzepte - Lösungen

Statistical Aspects (09-202-2413)

Janne Pott

Last compiled on 16 September, 2022

Session Setup

```
rm(list = ls())
time0<-Sys.time()

source("../sourceFile.R")
setwd(pathToExercise)

knitr::opts_chunk$set(echo = TRUE)
```

Verwandschaftsmatrix

Bitte laden Sie das Objekt *verwandtschaft.RData* mittels *load()*. Das R-Objekt enthält zwei Datensätze: *genotypes*, eine Genotyp-Matrix von 10 Personen mit 30,000 SNPs, und *allelfreq*, ein Vektor der Allelfrequenzen pro SNP aus *genotypes* bezüglich Allel B.

a) Die Verwandtschaftsmatrix K kann man mittels Matrix-Operation bestimmen. Definieren Sie dazu eine Hilfsmatrix h, die auf die Allelfrequenzen und Anzahl der SNPs adjustiert ist, und bilden Sie das Matrixprodukt $K = h^T * h$.

$$h_{m,i} = \frac{(g_{m,i} - 2 * p_{m,B})}{\sqrt{M * 4 * p_{m,B} * p_{m,A}}}$$

b) Warum gilt:

$$\hat{K}_{i,i} \approx 0.5$$

c) Wie viele paarweise Verwandtschaften (von Grad 1,2, ... , unverwandt) beobachten Sie?

d) Welche Familienstruktur könnte die beobachteten Verwandtschaftsbeziehungen erklären?

XY-Plots

Bitte laden Sie das Objekt *XYPlots.RData* mittels *load()*. Das R-Objekt enthält zwei Datensätze: *daten*, eine Matrix von 300 Personen und 300 gonosomalen SNPs (*X:i* bzw. *Y:i*), wobei die SNPs sowohl als Genotypen und Intensitäten pro Allel gegeben sind (AA = 0, AB = 1, BB = 2), und *samples*, ein Tabelle der gemessenen Individuen aus *daten* mit Geschlechtsinformationen.

Die Heterozygosität eines Samples ist der Anteil der AB Genotypen. Das genetische Geschlecht kann typischerweise über die Chr. X Heterozygosität bestimmt werden, da Männer in der Regel hier nur haploid

sind. Zusätzlich können die Bindungsintensitäten von X- und Y-SNPs mit berücksichtigt werden. In wenigen Ausfällen ist die Intensität der X- bzw. Y-SNPs zu verrauscht um eine Aussage zu treffen. Diese werden dann als *unknown* klassifiziert.

a) Für den XY-Plot brauchen wir die Gesamtintensitäten pro Sample. Bilden Sie daher zuerst den Mittelwert der Intensität für Allel A und B pro SNP und bilden Sie dann die jeweilige mittlere Intensität aller X-SNPs und aller Y-SNPs pro Sample!
b) Zusätzlich brauchen wir die Heterozygosität auf Chromosom X. Bestimmen Sie dazu pro Sample die Genotyphäufigkeiten (AA, AB, BB) aller SNPs von Chromosom X und berechnen Sie den Anteil von AB an allen Genotypen!
c) Sie sollten nun ein Objekt mit den Variablen *ID*, *X-Intensität*, *Y-Intensität, X-Heterozygosität*, *sex_datenbank* und *sex_computed* pro Sample haben. Erzeugen Sie nun folgende drei Plots und markieren Sie in diesen Plots Ausreißer (widersprüchliche Geschlechtsangabe, zu hohe/niedrige Intensitäten, auffällige Heterozygosität):
    i. X-Intensität – Y-Intensität
    ii. X-Intensität – X-Heterozygosität
    iii. Y-Intensität – X-Heterozygosität

Principal Component Analysis (PCA)

In dieser Aufgabe sollen die Eigenvektoren von genetischen Daten erzeugt werden. Dafür wird ein Teil der Daten von 1000 Genomes (1KG, Phase 1, release 3) verwendet. Zusätzlich zu R wird hier PLINK verwendet, da dieses Programm effizienter große Datenmengen verarbeiten kann.

Diese Aufgabe ist als Tutorial aufgebaut: jeder Schritt wird angefangen und soll von Ihnen vervollständigt werden.

Als erstes soll der Pfad zu PLINK definiert werden. Als kleiner Test wird Plink einmal aufgerufen.

- Falls 127 zurückggeben wird, hat R die Plink .exe nicht gefunden - bitte Pfad prüfen!

```
plink_call<-pathToPLINK2

# test if plink can start
system(plink_call)
```

Datenvorbereitung - SNPs filtern

Überprüfen Sie in R, ob alle SNPs von *mySNPs.txt* in 1KG sind. Hierfür sollte man am besten das *1KG_PCA.pvar* File verwenden (tab-delimited). Nutzen sie zum Einlesen der Befehl *fread()* aus dem Paket *data.table*. Filtern Sie nach den SNPs in der Schnittmenge und erstellen Sie ein gefiltertes Text-File mySNPs_filtered.txt!

Hinweis: Es sollten am Ende 16,132 SNPs sein!

```
myTab<-read.table(paste0(pathToData,"mySnps.txt"))
dim(myTab)
```

```
## [1] 224458      1
```

```
rslist<-fread(paste0(pathToData,"1KG_PCA.pvar"),sep="\t",stringsAsFactors=F)
dim(rslist)
```

```
## [1] 37844     5
```

```
head(rslist)
```

```
##    #CHROM     POS         ID REF ALT
## 1:      1  752566  rs3094315   A   G
## 2:      1  754105 rs12184325   C   T
## 3:      1  754182  rs3131969   G   A
## 4:      1  768448 rs12562034   G   A
## 5:      1  798959 rs11240777   G   A
## 6:      1 1036959 rs11579015   T   C
```

```
# to do: filter for SNPs in overlap and save as mySNPs_filtered.txt
```

Datenvorbereitung - Samples filtern

Erstellen Sie eine Sample Liste mit Individuen aus Asien, Afrika und Europa! Nutzen Sie hierfür das **1KG_PCA.psam** File (space-delimited). Wir wollen eine möglichst große Menge an Samples, aber jeder Herkunft sollte gleich oft vorhanden sein! Ziehen Sie zufällig aus der jeweiligen Teilmenge und speichern Sie ihre Liste als **mySamples.txt** ab!

Hinweis: Es sollten am Ende 3*246 Individuen sein!

```
fam.data<-read.table(paste0(pathToData,"1KG_PCA.psam"),stringsAsFactors=F,sep="\t")
dim(fam.data)
```

```
## [1] 1092    3
```

```
head(fam.data)
```

```
##        V1          V2 V3
## 1 HG00096 EUR_HG00096  1
## 2 HG00097 EUR_HG00097  2
## 3 HG00099 EUR_HG00099  2
## 4 HG00100 EUR_HG00100  2
## 5 HG00101 EUR_HG00101  1
## 6 HG00102 EUR_HG00102  2
```

```
ethno<-substr(fam.data$V2,1,3)
table(ethno)
```

```
## ethno
## AFR AMR ASN EUR
## 246 181 286 379
```

```
# to do: choose randomly individuals from AFR, ASN and EUR to obtain
#        same sample size per ethnicity and save as mySamples.txt
```

Datenvorbereitung - SNPs prunen

Jetzt prunen Sie die SNPs mit PLINK, d.h. Sie prüfen, welche SNPs in hohem LD miteinander sind. Folgende Parameter sollten Sie setzen: a. Input:–bfile 1KG_PCA b. SNPs einschränken: –extract mySNPs_filtered.txt c. Samples einschränken: –keep mySamples.txt d. LD-Pruning-Parameter festlegen: –indep-pairwise 50 5 0.2 e. Output: –out pruned_filter

Was bedeuten die drei Zahlen hinter dem –indep-pairwise Befehl?

Hinweis: Es sollten am Ende 9371 SNPs sein.

```
call1<-paste0(plink_call,
              " --pfile ",pathToData,"/1KG_PCA",
              " --extract ",pathToData,"/mySnps_filtered.txt",
              " --keep ",pathToData,"/mySamples.txt",
              " --indep-pairwise 50 5 0.2",
              " --out ",pathToData,"/pruning_filter")
system(call1)
```

Datenvorbereitung - Datensatz erstellen

Erstellen Sie jetzt mit PLINK ein neues .bed-File, dass nur noch die geprunten SNPs und die gewünschten Samples (aus 2) enthält (–bfile, –extract, –keep, und –make-bed).

```
call2<-paste0(plink_call,
              " --pfile ",pathToData,"1KG_PCA",
              " --extract ",pathToData,"pruning_filter.prune.in",
              " --keep ",pathToData,"mySamples.txt",
              " --make-bed --out ",pathToData,"pruned_data")
system(call2)
```

PCA berechnen

Jetzt kann mit den neuen Files die PCA ausgerechnet werden (–bfile, –pca, –out):

```
call3<-paste0(plink_call,
              " --bfile ",pathToData,"pruned_data",
              " --pca --out ",pathToData,"pca_out")
system(call3)
```

PCA auswerten

Laden Sie beide Outputs der PCA in R ein! Wie sind die Daten aufgebaut?

Erstellen Sie einen Plot der ersten beiden Vektoren mit Ethnien-Färbung! Was kann man daraus schließen?

Berechnen Sie den Anteil der erklärten Varianz a. durch den ersten Eigenvektor und b. durch die ersten beiden Eigenvektoren!

Was würden Sie erwarten, wenn alle 4 Ethnien in die Analyse eingeflossen wären? Wo würden Sie die Amerikaner einordnen? Rechnen Sie das nach!

```
pca2values<-read.table(paste0(pathToData,"pca_out.eigenval"))$V1
pca2vector<-read.table(paste0(pathToData,"pca_out.eigenvec"),stringsAsFactors=F,sep="\t")

# to do: estimate explained variance by the first two PCs and plot by ethnicities
```

Session Information

```
sessionInfo()
```

```
## R version 4.1.0 (2021-05-18)
## Platform: x86_64-suse-linux-gnu (64-bit)
## Running under: openSUSE Leap 15.3
##
## Matrix products: default
## BLAS:   /usr/lib64/R/lib/libRblas.so
## LAPACK: /usr/lib64/R/lib/libRlapack.so
##
## locale:
##  [1] LC_CTYPE=de_DE.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=de_DE.UTF-8        LC_COLLATE=de_DE.UTF-8
##  [5] LC_MONETARY=de_DE.UTF-8    LC_MESSAGES=de_DE.UTF-8
##  [7] LC_PAPER=de_DE.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=de_DE.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] ivpack_1.2        AER_1.2-9         sandwich_3.0-1    lmtest_0.9-39
##  [5] car_3.0-11        carData_3.0-5     qqman_0.1.8       meta_5.1-1
##  [9] nlme_3.1-155      Hmisc_4.4-2       ggplot2_3.3.5     Formula_1.2-4
## [13] survival_3.2-13   lattice_0.20-45   MASS_7.3-55       vioplot_0.3.7
## [17] zoo_1.8-9         sm_2.2-5.7        lubridate_1.8.0   readxl_1.3.1
## [21] data.table_1.14.2 doMC_1.3.7        doParallel_1.0.16 iterators_1.0.13
## [25] foreach_1.5.1     rmarkdown_2.11
##
## loaded via a namespace (and not attached):
##  [1] RColorBrewer_1.1-2  tools_4.1.0         backports_1.4.1
##  [4] utf8_1.2.2          R6_2.5.1            metafor_3.0-2
##  [7] rpart_4.1-15        DBI_1.1.2           colorspace_2.0-2
## [10] nnet_7.3-17         withr_2.4.3         tidyselect_1.1.1
## [13] gridExtra_2.3       curl_4.3            compiler_4.1.0
## [16] htmlTable_2.4.0     xml2_1.3.2          bookdown_0.24
## [19] scales_1.1.1        checkmate_2.0.0     stringr_1.4.0
## [22] digest_0.6.29       foreign_0.8-82      minqa_1.2.4
## [25] rio_0.5.27          base64enc_0.1-3     jpeg_0.1-9
## [28] pkgconfig_2.0.3     htmltools_0.5.2     lme4_1.1-27.1
## [31] fastmap_1.1.0       htmlwidgets_1.5.4   rlang_0.4.12
## [34] rstudioapi_0.13     generics_0.1.1      dplyr_1.0.7
## [37] zip_2.2.0           magrittr_2.0.1      Matrix_1.4-0
## [40] Rcpp_1.0.8          munsell_0.5.0       fansi_1.0.0
## [43] abind_1.4-5         lifecycle_1.0.1     stringi_1.7.6
## [46] yaml_2.2.1          CompQuadForm_1.4.3  mathjaxr_1.4-0
## [49] grid_4.1.0          forcats_0.5.1       crayon_1.4.2
## [52] haven_2.3.1         splines_4.1.0       hms_1.1.1
## [55] knitr_1.37          pillar_1.6.4        boot_1.3-28
## [58] codetools_0.2-18    glue_1.6.0          evaluate_0.14
## [61] calibrate_1.7.7     latticeExtra_0.6-29 png_0.1-7
## [64] vctrs_0.3.8         nloptr_1.2.2.3      cellranger_1.1.0
## [67] gtable_0.3.0        purrr_0.3.4         assertthat_0.2.1
## [70] xfun_0.29           openxlsx_4.2.5      tibble_3.1.6
```

```
## [73] cluster_2.1.2       ellipsis_0.3.2
```

```
message("\nTOTAL TIME : " ,round(difftime(Sys.time(),time0,units = "mins"),3)," minutes")
```

```
##
## TOTAL TIME : 0.122 minutes
```