Heritability Analysis of the UK data using the package patherit

Venelin Mitov

Contents

This script contains chunks that perform analysis of the HIV-data. The chunk run-poumm generates scripts with parallel jobs to be executed on a parallel cluster.

```
library(patherit)
```

```
## Loading required package: Rcpp
```

```
library(POUMM)
```

```
##
## Attaching package: 'POUMM'
```

```
## The following object is masked from 'package:patherit':
##
##     nodeTimes
```

```
## The following object is masked from 'package:stats':
##
##     simulate
```

```
library(data.table)
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:POUMM':
##
##     alpha
```

```
library(ape)
library(scales)
```

```
##
## Attaching package: 'scales'
##
## The following object is masked from 'package:POUMM':
##
##     alpha
```

```
library(lmtest)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
AICc <- function(...) {
  objs <- list(...)
  sapply(objs, function(o) {
    if("POUMM" %in% class(o)) {
      aic <- AIC(o)
      k <- length(coef(o))
      n <- nobs(o)
      aic + (2 * k * (k + 1)) / (n - k - 1)
    } else {
      logLik <- -o$value
      k <- if(o$par[1]==0) 3 else 5
      aic <- -2*logLik + 2*k
      n <- o$N
      aic + (2 * k * (k + 1)) / (n - k - 1)
    }
  })
}


r2 <- function(x) round(x, 2)
fmt <- function(x) format(x, nsmall=0, big.mark=",")
r2int <- function(x) paste0('[', round(x[1], 2), ', ', round(x[2], 2), ']')
lrt <- function(fit0, fit1) {
  loglik1 <- logLik(fit1)
  loglik0 <- logLik(fit0)
  df1 <- attr(loglik1, 'df')
  df0 <- attr(loglik0, 'df')
  D <- 2*(loglik1-loglik0)
  p <- pchisq(D, df1-df0, lower.tail = FALSE)
  stars <- ""
  if(p < 0.05) {
    stars <- paste0(stars, "*")
  }
  if(p < 0.01) {
    stars <- paste0(stars, "*")
  }
  if(p < 0.001) {
    stars <- paste0(stars, "*")
  }
  paste0(round(D, 2), stars)
}

filterIQR <- function(x, fact=1.5) {
  IQR <- quantile(x, probs=c(.25,.75));
  x >= IQR[1]-fact*(IQR[2]-IQR[1]) & x <= IQR[2]+fact*(IQR[2]-IQR[1]);
}
```

1 Loading the tree and the set-point viral load values

Below, we provide the code used to load the data. We don't have the right to redistribute the data-files "B_RAxML.1.newick", "ViralLoad.csv" and "hivdata.RData". For obtaining access to this data, we recommend contacting the UK drug resistance cohort or the authors of (Hodcroft et al., 2014).

```
# ukdata
treeUK <- read.tree(file='DATA/HIV/B_RAxML.1.newick')
vUK <- read.table(file='DATA/HIV/ViralLoad.csv', header=T, sep=',', row.names=1)

# test tip-names
all(treeUK$tip.label==rownames(vUK))

# drop NA tips
treeUK <- drop.tip(treeUK, tip=which(is.na(vUK[,1])))
vUK <- as.vector(vUK[treeUK$tip.label, ])
```

2 Store this data as a row in a data.table.

Since there is only one row in the data.table, this operation is not really needed, but working with a data.table rather than a list of objects is not a big difference. In future work, this would facilitate combining the analysis on the UK Data with results from other HIV cohort studies.

```
hivdata <- data.table(id=c(40001),
                      cohort=c('UK_Hodcroft_2014'),
                      subdiv=c('All'),
                      tree=list(treeUK),
                      nTips=c(length(vUK)),
                      v=list(vUK), key='id' )

save(hivdata, file='DATA/HIV/hivdata.RData')
```

3 Estimate the heritability using POUMM, PMM and the ANOVA-PP (ANOVA-CPP) methods.

```
load('DATA/HIV/hivdata.RData')

hivdata[, anH2:=lapply(1, function(i) {
  estimateH2(
    v[[i]], tree[[i]],
    methods = list(PP = list(bootstraps=100, verbose=TRUE),
                   POUMM = list(nSamplesMCMC = 4e6,
                                verbose = TRUE),
                   PMM = list(nSamplesMCMC = 4e6,
                                verbose = TRUE)))
  })]

hivdata[, corrProfile:=lapply(1, function(i) {
  corrProfile(anH2[[i]], verbose = TRUE)
})]
```

```
save(hivdata, file='DATA/HIV/hivdata.RData')
```

For the UK data, we also did a custom CPP analysis by imposing a threshold 10^{-4} for the phylogenetic distance. This is currently not supported in the patherit package, so we write a small script for this analysis:

```
NUK <- length(treeUK$tip.label)
rootTipDistsUK <- nodeTimes(treeUK)[1:NUK]

rootTipDistGroupsUK <- cut(rootTipDistsUK, breaks=seq(.01, 0.3, by=0.02))
names(rootTipDistGroupsUK) <- treeUK$tip.label

tipDistsUK <- dist.nodes(treeUK)[1:NUK,1:NUK]
tipDistsUK <- sapply(1:NUK, function(i) sapply(1:NUK, function(j) if(i>j) tipDistsUK[i,j] else NA))
tipDistsUK <- tipDistsUK[!is.na(tipDistsUK)]
zDistsUK <- sapply(1:NUK, function(i) sapply(1:NUK, function(j) if(i>j) abs(vUK[i]-vUK[j]) else NA))
zDistsUK <- zDistsUK[!is.na(zDistsUK)]
ppUK <- extractPP(treeUK)
setkey(ppUK, i)
ppUK[, z:=vUK[i]]
ppUK[!is.na(idPair), deltaz:=abs(z[1]-z[2]), by=idPair]
```

```
rAOV <- function(i, idPair, z) {
  data <- data.table(gene=idPair, z)
  bootstrap <- boot(data=data[, unique(idPair)], statistic=function(idPP, ids) {
    rA(data=data[idPair%in%idPP[ids]])
    }, R=1000)
  aovReport=rA(data=data, report=TRUE)
  list(tips=list(i), bootstrap=list(bootstrap),
       bCI95lower=boot.ci(bootstrap, type='basic')$basic[4],
       bCI95upper=boot.ci(bootstrap, type='basic')$basic[5],
       rA=aovReport$H2aov,
       CI95lower=aovReport$CI95lower,
       CI95upper=aovReport$CI95upper,
       sigma2G=aovReport$sigma2G,
       sigma2z=aovReport$sigma2G+aovReport$sigma2E,
       n=aovReport$n0,
       N=aovReport$N, K=aovReport$K)
}
```

```
set.seed(10)

hivdata[J(40001),
        analysis.CPP := list(
          list(c(list(pp=ppUK[d<=10^-4]),
                 with(ppUK[d<=10^-4], rAOV(i, idPair, z)))))]
hivdata[J(40001),
        analysis.PP := list(list(c(list(pp=ppUK),
                                   with(ppUK, rAOV(i, idPair, z)))))]

save(hivdata, file='DATA/HIV/hivdata.RData')
```

```
load("DATA/HIV/hivdata.RData")
```

```
smmAnH2_1 <- summary(hivdata$anH2[[1]], useBootstrapsForPP = TRUE)
smmPOUMM_1 <- summary(hivdata$anH2[[1]]$fits$POUMM)
```

```
smmPMM_1 <- summary(hivdata$anH2[[1]]$fits$PMM)

rowNo <- 1
lm_t_tau <- hivdata$anH2[[rowNo]]$fits$PP$pp[i<j, coef(lm(t~tau))]

tMeanPOUMM <- mean(nodeTimes(hivdata$tree[[rowNo]], tipsOnly = TRUE))

coefPMM <- hivdata$anH2[[1]]$fits$PMM$spec$parMapping(coef(hivdata$anH2[[1]]$fits$PMM))
alphaPMM <- coefPMM['alpha']
sigma2PMM <- coefPMM['sigma']^2
sigmae2PMM <- coefPMM['sigmae']^2
sigma2zPMM <- POUMM::varOU(tMeanPOUMM, 0, sqrt(sigma2PMM)) + sigmae2PMM

coefPOUMM <- hivdata$anH2[[1]]$fits$POUMM$spec$parMapping(coef(hivdata$anH2[[1]]$fits$POUMM))
alphaPOUMM <- coefPOUMM['alpha']
sigma2POUMM <- coefPOUMM['sigma']^2
sigmae2POUMM <- coefPOUMM['sigmae']^2
sigma2zPOUMM <- POUMM::varOU(tMeanPOUMM, 0, sqrt(sigma2POUMM)) + sigmae2POUMM

H2POUMM <- POUMM::H2(alphaPOUMM, sqrt(sigma2POUMM), sqrt(sigmae2POUMM), t = tMeanPOUMM)
H2ePOUMM <- POUMM::H2e(hivdata$v[[rowNo]], sqrt(sigmae2POUMM), hivdata$tree[[rowNo]])

corrTable <- hivdata$corrProfile[[rowNo]]$corrTable

corrTable[, tauQuantileType:=factor(tauQuantileType,
                                    levels = rev(levels(tauQuantileType)))]

pp1 <- hivdata$anH2[[1]]$fits$PP$pp
pp1[, muz:=mean(z)]
pp1[, varz:=var(z)]
pp1.corr <- pp1[, list(tau=unique(tau), y=unique((z[1]-muz)*(z[2]-muz)/varz)), by=idPair]
lm.pp1.corr <- pp1.corr[, lm(y~tau)]
pp1[, ttip:=nodeTimes(hivdata$tree[[1]], tipsOnly = TRUE)[i]]

rSpTable <- rbindlist(lapply(
  corrTable[, levels(tauQuantileType)],
  function(tqt) {
    byCol <- paste0(tqt, "tau")
    pp1[, .SD[, list(z1=z[1], z2=z[2],
                     tau=unique(tau), t=unique(t)),
              by=idPair][, {
                rSp = cor(z1, z2, method="spearman")
                list(
                  stat="rSp", N=.N*2, K=.N,
                  est=rSp,
                  filter="all",
                  tauMean=mean(tau),
                  tauMedian=median(tau),
                  tMean = mean(t),
                  tMedian = median(t),
                  tauQuantileType = tqt,
                  CI.lower = rSp - 1.96/sqrt(.N-3),
                  CI.upper = rSp + 1.96/sqrt(.N-3),
```

```
                  Data = "Original ranks",
                  method="PP",
                  MLE = NA
                )}
                ],
       keyby=list(tauQuantile=eval(parse(text=byCol)))]
  }))

corrTable <- rbind(corrTable, rSpTable)


tqt <- "D"

linmodA <- corrTable[Data=="Original" & tauQuantileType == tqt, lm(est~tauMean)]
corrTable[Data == "Original" & tauQuantileType == tqt,
          errLin:=predict(linmodA)-est]

corrTable[Data == "Original" & tauQuantileType == tqt,
          aLinmod:=summary(linmodA)$coefficients[1, 1]]
corrTable[Data == "Original" & tauQuantileType == tqt,
          aLinmodCIlower:=confint(linmodA)[1, 1]]
corrTable[Data == "Original" & tauQuantileType == tqt,
          aLinmodCIupper:=confint(linmodA)[1, 2]]


corrTable[Data == "Original" & tauQuantileType == tqt,
          bLinmod:=summary(linmodA)$coefficients[2, 1]]
corrTable[Data == "Original" & tauQuantileType == tqt,
          bLinmodCIlower:=confint(linmodA)[2, 1]]
corrTable[Data == "Original" & tauQuantileType == tqt,
          bLinmodCIupper:=confint(linmodA)[2, 2]]
corrTable[Data == "Original" & tauQuantileType == tqt,
          pLinmod:=summary(linmodA)$coefficients[2, 4]]


linmodSp <- corrTable[Data=="Original ranks" & tauQuantileType == tqt, lm(est~tauMean)]
corrTable[Data == "Original ranks" & tauQuantileType == tqt,
          errLin:=predict(linmodSp)-est]

corrTable[Data == "Original ranks" & tauQuantileType == tqt,
          aLinmod:=summary(linmodSp)$coefficients[1, 1]]
corrTable[Data == "Original ranks" & tauQuantileType == tqt,
          aLinmodCIlower:=confint(linmodSp)[1, 1]]
corrTable[Data == "Original ranks" & tauQuantileType == tqt,
          aLinmodCIupper:=confint(linmodSp)[1, 2]]


corrTable[Data == "Original ranks" & tauQuantileType == tqt,
          bLinmod:=summary(linmodSp)$coefficients[2, 1]]
corrTable[Data == "Original ranks" & tauQuantileType == tqt,
          bLinmodCIlower:=confint(linmodSp)[2, 1]]
corrTable[Data == "Original ranks" & tauQuantileType == tqt,
          bLinmodCIupper:=confint(linmodSp)[2, 2]]
```

```
corrTable[Data == "Original ranks" & tauQuantileType == tqt,
          pLinmod:=summary(linmodSp)$coefficients[2, 4]]


rA.models.1 <- corrTable[tauQuantileType=="D"&stat=="rA"&Data=="Original"]
```

```
tabCor <- corrTable[tauQuantileType=="D" & stat=="rA"&Data=="Original"]
tabCorV <- corrTable[tauQuantileType=="V" & stat=="rA"&Data=="Original"]
tabCorSp10_4 <- pp1[tau<10^(-4), list(z1=z[1], z2=z[2]), by=idPair][
  , {
    rSp <- cor(z1, z2, method="spearman")
    list(N=2*.N, est=cor(z1, z2, method="spearman"),
         CI.lower=rSp - 1.96/sqrt(.N-3),
         CI.upper=rSp + 1.96/sqrt(.N-3))
    }]
tabCorSpA <- corrTable[tauQuantileType=="A" & stat=="rSp"&Data=="Original ranks"]
tabCorSpD <- corrTable[tauQuantileType=="D" & stat=="rSp"&Data=="Original ranks"]
tabCorSpV <- corrTable[tauQuantileType=="V" & stat=="rSp"&Data=="Original ranks"]
```