Bayesianische Statistik trifft Sport 16784

Adrian Brenner 12422308

2024-03-26

Before applying my own data I will test and update the given Appendix B code to the current R and RStudio versions. This R Code was created to calculate the Bayes factor BF_HC.

```
## requires lattice and HiddenMarkov to run
library(lattice)
library(HiddenMarkov)
```

```
## Warning: Paket 'HiddenMarkov' wurde unter R Version 4.3.1 erstellt
```

```
# Example: Sequence of Carlos Guillen's batting outcomes
# for the 2005 season (Albert, 2008), 1 is a hit and 0 is
# out.
Guillen.data <-c(
0,1,0,1,1,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1,1,1,0,0,0,0,0,0,
1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,1,0,1,0,0,1,1,0,
1,0,1,1,0,1,0,1,1,0,0,0,0,0,1,1,1,1,0,0,1,0,1,0,0,1,1,0,0,
0,1,0,1,0,0,0,1,1,1,0,1,1,1,1,0,0,1,1,1,1,0,0,1,0,0,1,0,1,
0,0,0,1,0,1,0,0,0,0,0,1,1,1,0,0,1,0,0,0,0,0,0,1,1,1,0,1,0,
0,0,0,1,1,1,1,0,1,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1,1,0,1,0,
0,0,0,0,0,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,
0,1,0,0,0,0,1,1,0,1,0,0,1,0,0,0,1,0,1,0,0,0,0,0,0,0,0,1,0,
1,0,0,1,1,1,1,0,0,0,0,0,1,1,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,1,0,0,1,0,0,
0,1,1,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,0,0,1,1,0,1,1,1,0,1,0,0,
0,0,0,1,0,1,1,0,0,0,1,0,0,0,1)
## the Bayes factor function
## use LogBF = LogBayesfactorHMM(data, gridprecision)
## input parameters are:
## dat: the data in vector form
## intprec: the precision of the integration process, higher is more precise
LogBayesfactorHMM <- function(dat,intprec=50){
grid1 <- 1 / (2 * (intprec - 1)) + seq(0, 1, length = intprec)[-intprec]
grid2 <- 1 / (2 * intprec) + seq(0, 0.5, length = intprec)[-intprec]
len1 <- length(grid1)
len2 <- length(grid2)
indices <- which(upper.tri(matrix(TRUE, len1, len1)), arr.ind = TRUE)
zzz <- matrix(0, nrow(indices), len1)
n <- length(dat)
m <- 2L
phi <- as.double(c(0.5, 0.5))
logalpha <- matrix(as.double(rep(0, m * n)), nrow = n)
lscale <- as.double(0)
memory0 <- rep(as.double(0), m)
for(k in seq_len(len2)){
```

```
xPi <- matrix(c(1 - grid2[k], grid2[k],
grid2[k], 1 - grid2[k]), 2, 2)
for(i in seq_len(nrow(zzz))){
prob <-
cbind(grid1[indices[i, 1]]^dat * (1 - grid1[indices[i, 1]])^(1 - dat),
grid1[indices[i, 2]]^dat * (1 - grid1[indices[i, 2]])^(1 - dat))
zzz[i, k] <- .Fortran("loop1", m, n, phi, prob, xPi, logalpha,
lscale, memory0, PACKAGE = "HiddenMarkov")[[7]]
}
}
f <- 699 - max(zzz, na.rm = TRUE)
zzz <- exp(zzz + f)
kMarginalHMM <- log(mean(zzz,na.rm=T)) - f
kMarginalCPM <- lgamma(length(dat[dat == 1]) + 1) + lgamma(length(dat)
- length(dat[dat == 1]) + 1) - lgamma(length(dat) + 2)
BF <- kMarginalHMM - kMarginalCPM
return(BF)
}
# compute log Bayes factor HMM/CPM
# positive log(BF) indicates evidence in favor of HMM
LogBayesfactorHMM(Guillen.data) ## should be approx 0.45
```

```
## [1] 0.4521182
```

Output of Dummy-Baseball HiddenMarkov code successful. Now I will reapply the same procedure for a multitude of games in the 2005/2006 season, again for both Kobe Bryant and Shaquille O'Neal.

Note here that it is not necessary to compute both for the HMM and the CPM. The code is programmed and adjusted for to output a factor suitable for an interpretation based on Jefferys tabular mentioned in the term paper (See: 2.1 Existing Theory).

```
library(lattice)
library(HiddenMarkov)

KobeBEST.data <-c(1,1,0,0,0,1,1,0,0,1,1,1,1,1,1,0,0,0,1,1,1,0,1,1,0,0,1,1,1,1,1,1,1,
1,1,0,1,1,1,0,0,1,1,1,1,0,0,1,1,1,1,1,1,1,1,0,1,0,1,1,0,1,1,1,1,1)

LogBayesfactorHMM <- function(dat,intprec=50){
grid1 <- 1 / (2 * (intprec - 1)) + seq(0, 1, length = intprec)[-intprec]
grid2 <- 1 / (2 * intprec) + seq(0, 0.5, length = intprec)[-intprec]
len1 <- length(grid1)
len2 <- length(grid2)
indices <- which(upper.tri(matrix(TRUE, len1, len1)), arr.ind = TRUE)
zzz <- matrix(0, nrow(indices), len1)
n <- length(dat)
m <- 2L
phi <- as.double(c(0.5, 0.5))
logalpha <- matrix(as.double(rep(0, m * n)), nrow = n)
lscale <- as.double(0)
memory0 <- rep(as.double(0), m)
for(k in seq_len(len2)){
xPi <- matrix(c(1 - grid2[k], grid2[k],
grid2[k], 1 - grid2[k]), 2, 2)
for(i in seq_len(nrow(zzz))){
prob <-
```

```
cbind(grid1[indices[i, 1]]^dat * (1 - grid1[indices[i, 1]])^(1 - dat),
grid1[indices[i, 2]]^dat * (1 - grid1[indices[i, 2]])^(1 - dat))
zzz[i, k] <- .Fortran("loop1", m, n, phi, prob, xPi, logalpha,
lscale, memory0, PACKAGE = "HiddenMarkov")[[7]]
}
}
f <- 699 - max(zzz, na.rm = TRUE)
zzz <- exp(zzz + f)
kMarginalHMM <- log(mean(zzz,na.rm=T)) - f
kMarginalCPM <- lgamma(length(dat[dat == 1]) + 1) + lgamma(length(dat)
- length(dat[dat == 1]) + 1) - lgamma(length(dat) + 2)
BF <- kMarginalHMM - kMarginalCPM
return(BF)
}
LogBayesfactorHMM(KobeBEST.data)
```

```
## [1] 0.19158
```

```
library(lattice)
library(HiddenMarkov)

KobeWORST.data <-c(0,1,1,0,0,0,1,0,0,1,1,0,0,0)

LogBayesfactorHMM <- function(dat,intprec=50){
grid1 <- 1 / (2 * (intprec - 1)) + seq(0, 1, length = intprec)[-intprec]
grid2 <- 1 / (2 * intprec) + seq(0, 0.5, length = intprec)[-intprec]
len1 <- length(grid1)
len2 <- length(grid2)
indices <- which(upper.tri(matrix(TRUE, len1, len1)), arr.ind = TRUE)
zzz <- matrix(0, nrow(indices), len1)
n <- length(dat)
m <- 2L
phi <- as.double(c(0.5, 0.5))
logalpha <- matrix(as.double(rep(0, m * n)), nrow = n)
lscale <- as.double(0)
memory0 <- rep(as.double(0), m)
for(k in seq_len(len2)){
xPi <- matrix(c(1 - grid2[k], grid2[k],
grid2[k], 1 - grid2[k]), 2, 2)
for(i in seq_len(nrow(zzz))){
prob <-
cbind(grid1[indices[i, 1]]^dat * (1 - grid1[indices[i, 1]])^(1 - dat),
grid1[indices[i, 2]]^dat * (1 - grid1[indices[i, 2]])^(1 - dat))
zzz[i, k] <- .Fortran("loop1", m, n, phi, prob, xPi, logalpha,
lscale, memory0, PACKAGE = "HiddenMarkov")[[7]]
}
}
f <- 699 - max(zzz, na.rm = TRUE)
zzz <- exp(zzz + f)
kMarginalHMM <- log(mean(zzz,na.rm=T)) - f
kMarginalCPM <- lgamma(length(dat[dat == 1]) + 1) + lgamma(length(dat)
- length(dat[dat == 1]) + 1) - lgamma(length(dat) + 2)
BF <- kMarginalHMM - kMarginalCPM
return(BF)
```

```
}
LogBayesfactorHMM(KobeWORST.data)
```

```
## [1] 0.1563497
```

```
library(lattice)
library(HiddenMarkov)

Kobe1.data <-c(0,1,1,1,1,0,1,1,0,1,0,0,0,0,0,0,1,1,0,1,0,0,1,1,1,0,1,1,0,0,1,1,1,1,1)

LogBayesfactorHMM <- function(dat,intprec=50){
grid1 <- 1 / (2 * (intprec - 1)) + seq(0, 1, length = intprec)[-intprec]
grid2 <- 1 / (2 * intprec) + seq(0, 0.5, length = intprec)[-intprec]
len1 <- length(grid1)
len2 <- length(grid2)
indices <- which(upper.tri(matrix(TRUE, len1, len1)), arr.ind = TRUE)
zzz <- matrix(0, nrow(indices), len1)
n <- length(dat)
m <- 2L
phi <- as.double(c(0.5, 0.5))
logalpha <- matrix(as.double(rep(0, m * n)), nrow = n)
lscale <- as.double(0)
memory0 <- rep(as.double(0), m)
for(k in seq_len(len2)){
xPi <- matrix(c(1 - grid2[k], grid2[k],
grid2[k], 1 - grid2[k]), 2, 2)
for(i in seq_len(nrow(zzz))){
prob <-
cbind(grid1[indices[i, 1]]^dat * (1 - grid1[indices[i, 1]])^(1 - dat),
grid1[indices[i, 2]]^dat * (1 - grid1[indices[i, 2]])^(1 - dat))
zzz[i, k] <- .Fortran("loop1", m, n, phi, prob, xPi, logalpha,
lscale, memory0, PACKAGE = "HiddenMarkov")[[7]]
}
}
f <- 699 - max(zzz, na.rm = TRUE)
zzz <- exp(zzz + f)
kMarginalHMM <- log(mean(zzz,na.rm=T)) - f
kMarginalCPM <- lgamma(length(dat[dat == 1]) + 1) + lgamma(length(dat)
- length(dat[dat == 1]) + 1) - lgamma(length(dat) + 2)
BF <- kMarginalHMM - kMarginalCPM
return(BF)
}
LogBayesfactorHMM(Kobe1.data)
```

```
## [1] 0.4952662
```

```
library(lattice)
library(HiddenMarkov)

Kobe2.data <-c(0,1,1,0,1,1,1,1,1,1,1,1,1,0,0,1,1,0,0,0,1,1,0,1,0,1,0,1,1,1,0,1,1,1,1,1,1,0,0,0)

LogBayesfactorHMM <- function(dat,intprec=50){
grid1 <- 1 / (2 * (intprec - 1)) + seq(0, 1, length = intprec)[-intprec]
grid2 <- 1 / (2 * intprec) + seq(0, 0.5, length = intprec)[-intprec]
len1 <- length(grid1)
```

```
len2 <- length(grid2)
indices <- which(upper.tri(matrix(TRUE, len1, len1)), arr.ind = TRUE)
zzz <- matrix(0, nrow(indices), len1)
n <- length(dat)
m <- 2L
phi <- as.double(c(0.5, 0.5))
logalpha <- matrix(as.double(rep(0, m * n)), nrow = n)
lscale <- as.double(0)
memory0 <- rep(as.double(0), m)
for(k in seq_len(len2)){
xPi <- matrix(c(1 - grid2[k], grid2[k],
grid2[k], 1 - grid2[k]), 2, 2)
for(i in seq_len(nrow(zzz))){
prob <-
cbind(grid1[indices[i, 1]]^dat * (1 - grid1[indices[i, 1]])^(1 - dat),
grid1[indices[i, 2]]^dat * (1 - grid1[indices[i, 2]])^(1 - dat))
zzz[i, k] <- .Fortran("loop1", m, n, phi, prob, xPi, logalpha,
lscale, memory0, PACKAGE = "HiddenMarkov")[[7]]
}
}
f <- 699 - max(zzz, na.rm = TRUE)
zzz <- exp(zzz + f)
kMarginalHMM <- log(mean(zzz,na.rm=T)) - f
kMarginalCPM <- lgamma(length(dat[dat == 1]) + 1) + lgamma(length(dat)
- length(dat[dat == 1]) + 1) - lgamma(length(dat) + 2)
BF <- kMarginalHMM - kMarginalCPM
return(BF)
}
LogBayesfactorHMM(Kobe2.data)
```

```
## [1] 0.3560458
```

```
library(lattice)
library(HiddenMarkov)

Kobe3.data <-c(1,1,1,1,1,0,1,0,0,1,0,1,0,1,1,1,0,1,0,0,1,0,0,1,0,1,0,0,1,0,1,1,1,1,1)

LogBayesfactorHMM <- function(dat,intprec=50){
grid1 <- 1 / (2 * (intprec - 1)) + seq(0, 1, length = intprec)[-intprec]
grid2 <- 1 / (2 * intprec) + seq(0, 0.5, length = intprec)[-intprec]
len1 <- length(grid1)
len2 <- length(grid2)
indices <- which(upper.tri(matrix(TRUE, len1, len1)), arr.ind = TRUE)
zzz <- matrix(0, nrow(indices), len1)
n <- length(dat)
m <- 2L
phi <- as.double(c(0.5, 0.5))
logalpha <- matrix(as.double(rep(0, m * n)), nrow = n)
lscale <- as.double(0)
memory0 <- rep(as.double(0), m)
for(k in seq_len(len2)){
xPi <- matrix(c(1 - grid2[k], grid2[k],
grid2[k], 1 - grid2[k]), 2, 2)
for(i in seq_len(nrow(zzz))){
```

```
prob <-
cbind(grid1[indices[i, 1]]^dat * (1 - grid1[indices[i, 1]])^(1 - dat),
grid1[indices[i, 2]]^dat * (1 - grid1[indices[i, 2]])^(1 - dat))
zzz[i, k] <- .Fortran("loop1", m, n, phi, prob, xPi, logalpha,
lscale, memory0, PACKAGE = "HiddenMarkov")[[7]]
}
}
f <- 699 - max(zzz, na.rm = TRUE)
zzz <- exp(zzz + f)
kMarginalHMM <- log(mean(zzz,na.rm=T)) - f
kMarginalCPM <- lgamma(length(dat[dat == 1]) + 1) + lgamma(length(dat)
- length(dat[dat == 1]) + 1) - lgamma(length(dat) + 2)
BF <- kMarginalHMM - kMarginalCPM
return(BF)
}
LogBayesfactorHMM(Kobe3.data)
```

```
## [1] 0.1817988
```

Log Result for Bryant BEST game: 0.19158 -> 1.211162 (HMM - not worth more than a bare mention)

```
exp(0.19158)
```

```
## [1] 1.211162
```

Log Result for Bryant WORST game: 0.1563497 -> 1.169235 (HMM - not worth more than a bare mention)

```
exp(0.1563497)
```

```
## [1] 1.169235
```

Log Result for Bryant 1st game: 0.4952662 -> 1.640935 (HMM - not worth more than a bare mention)

```
exp(0.4952662)
```

```
## [1] 1.640935
```

Log Result for Bryant 2nd game: 0.3560458 -> 1.427673 (HMM - not worth more than a bare mention)

```
exp(0.3560458)
```

```
## [1] 1.427673
```

Log Result for Bryant 3rd game: 0.1817988 -> 1.199373 (HMM - not worth more than a bare mention)

```
exp(0.1817988)
```

```
## [1] 1.199373
```

And now for the 5 games for Shaquille O'Neal.

```
library(lattice)
library(HiddenMarkov)

ShaqBEST.data <-c(0,1,1,0,0,0,0,1,1,0,1,0,1,0,1,0,1,1,0,1,1,1,1,0,0,1,0,0,1,1,1,1,1)

LogBayesfactorHMM <- function(dat,intprec=50){
grid1 <- 1 / (2 * (intprec - 1)) + seq(0, 1, length = intprec)[-intprec]
grid2 <- 1 / (2 * intprec) + seq(0, 0.5, length = intprec)[-intprec]
len1 <- length(grid1)
len2 <- length(grid2)
```

```
indices <- which(upper.tri(matrix(TRUE, len1, len1)), arr.ind = TRUE)
zzz <- matrix(0, nrow(indices), len1)
n <- length(dat)
m <- 2L
phi <- as.double(c(0.5, 0.5))
logalpha <- matrix(as.double(rep(0, m * n)), nrow = n)
lscale <- as.double(0)
memory0 <- rep(as.double(0), m)
for(k in seq_len(len2)){
xPi <- matrix(c(1 - grid2[k], grid2[k],
grid2[k], 1 - grid2[k]), 2, 2)
for(i in seq_len(nrow(zzz))){
prob <-
cbind(grid1[indices[i, 1]]^dat * (1 - grid1[indices[i, 1]])^(1 - dat),
grid1[indices[i, 2]]^dat * (1 - grid1[indices[i, 2]])^(1 - dat))
zzz[i, k] <- .Fortran("loop1", m, n, phi, prob, xPi, logalpha,
lscale, memory0, PACKAGE = "HiddenMarkov")[[7]]
}
}
f <- 699 - max(zzz, na.rm = TRUE)
zzz <- exp(zzz + f)
kMarginalHMM <- log(mean(zzz,na.rm=T)) - f
kMarginalCPM <- lgamma(length(dat[dat == 1]) + 1) + lgamma(length(dat)
- length(dat[dat == 1]) + 1) - lgamma(length(dat) + 2)
BF <- kMarginalHMM - kMarginalCPM
return(BF)
}
LogBayesfactorHMM(ShaqBEST.data)
```

```
## [1] 0.1646427
```

```
library(lattice)
library(HiddenMarkov)

ShaqWORST.data <-c(1,0,0,0,0,1,0,1,0,0,0,0)

LogBayesfactorHMM <- function(dat,intprec=50){
grid1 <- 1 / (2 * (intprec - 1)) + seq(0, 1, length = intprec)[-intprec]
grid2 <- 1 / (2 * intprec) + seq(0, 0.5, length = intprec)[-intprec]
len1 <- length(grid1)
len2 <- length(grid2)
indices <- which(upper.tri(matrix(TRUE, len1, len1)), arr.ind = TRUE)
zzz <- matrix(0, nrow(indices), len1)
n <- length(dat)
m <- 2L
phi <- as.double(c(0.5, 0.5))
logalpha <- matrix(as.double(rep(0, m * n)), nrow = n)
lscale <- as.double(0)
memory0 <- rep(as.double(0), m)
for(k in seq_len(len2)){
xPi <- matrix(c(1 - grid2[k], grid2[k],
grid2[k], 1 - grid2[k]), 2, 2)
for(i in seq_len(nrow(zzz))){
prob <-
```

```
cbind(grid1[indices[i, 1]]^dat * (1 - grid1[indices[i, 1]])^(1 - dat),
grid1[indices[i, 2]]^dat * (1 - grid1[indices[i, 2]])^(1 - dat))
zzz[i, k] <- .Fortran("loop1", m, n, phi, prob, xPi, logalpha,
lscale, memory0, PACKAGE = "HiddenMarkov")[[7]]
}
}
f <- 699 - max(zzz, na.rm = TRUE)
zzz <- exp(zzz + f)
kMarginalHMM <- log(mean(zzz,na.rm=T)) - f
kMarginalCPM <- lgamma(length(dat[dat == 1]) + 1) + lgamma(length(dat)
- length(dat[dat == 1]) + 1) - lgamma(length(dat) + 2)
BF <- kMarginalHMM - kMarginalCPM
return(BF)
}
LogBayesfactorHMM(ShaqWORST.data)
```

```
## [1] 0.01519907
```

```
library(lattice)
library(HiddenMarkov)

Shaq1.data <-c(0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,1,1)

LogBayesfactorHMM <- function(dat,intprec=50){
grid1 <- 1 / (2 * (intprec - 1)) + seq(0, 1, length = intprec)[-intprec]
grid2 <- 1 / (2 * intprec) + seq(0, 0.5, length = intprec)[-intprec]
len1 <- length(grid1)
len2 <- length(grid2)
indices <- which(upper.tri(matrix(TRUE, len1, len1)), arr.ind = TRUE)
zzz <- matrix(0, nrow(indices), len1)
n <- length(dat)
m <- 2L
phi <- as.double(c(0.5, 0.5))
logalpha <- matrix(as.double(rep(0, m * n)), nrow = n)
lscale <- as.double(0)
memory0 <- rep(as.double(0), m)
for(k in seq_len(len2)){
xPi <- matrix(c(1 - grid2[k], grid2[k],
grid2[k], 1 - grid2[k]), 2, 2)
for(i in seq_len(nrow(zzz))){
prob <-
cbind(grid1[indices[i, 1]]^dat * (1 - grid1[indices[i, 1]])^(1 - dat),
grid1[indices[i, 2]]^dat * (1 - grid1[indices[i, 2]])^(1 - dat))
zzz[i, k] <- .Fortran("loop1", m, n, phi, prob, xPi, logalpha,
lscale, memory0, PACKAGE = "HiddenMarkov")[[7]]
}
}
f <- 699 - max(zzz, na.rm = TRUE)
zzz <- exp(zzz + f)
kMarginalHMM <- log(mean(zzz,na.rm=T)) - f
kMarginalCPM <- lgamma(length(dat[dat == 1]) + 1) + lgamma(length(dat)
- length(dat[dat == 1]) + 1) - lgamma(length(dat) + 2)
BF <- kMarginalHMM - kMarginalCPM
return(BF)
```

```
}
LogBayesfactorHMM(Shaq1.data)
```

```
## [1] 0.4760496
```

```
library(lattice)
library(HiddenMarkov)

Shaq2.data <-c(1,1,0,1,1,0,1,0,1,1,0,1,1,0,0,0,1,1,1,0)

LogBayesfactorHMM <- function(dat,intprec=50){
grid1 <- 1 / (2 * (intprec - 1)) + seq(0, 1, length = intprec)[-intprec]
grid2 <- 1 / (2 * intprec) + seq(0, 0.5, length = intprec)[-intprec]
len1 <- length(grid1)
len2 <- length(grid2)
indices <- which(upper.tri(matrix(TRUE, len1, len1)), arr.ind = TRUE)
zzz <- matrix(0, nrow(indices), len1)
n <- length(dat)
m <- 2L
phi <- as.double(c(0.5, 0.5))
logalpha <- matrix(as.double(rep(0, m * n)), nrow = n)
lscale <- as.double(0)
memory0 <- rep(as.double(0), m)
for(k in seq_len(len2)){
xPi <- matrix(c(1 - grid2[k], grid2[k],
grid2[k], 1 - grid2[k]), 2, 2)
for(i in seq_len(nrow(zzz))){
prob <-
cbind(grid1[indices[i, 1]]^dat * (1 - grid1[indices[i, 1]])^(1 - dat),
grid1[indices[i, 2]]^dat * (1 - grid1[indices[i, 2]])^(1 - dat))
zzz[i, k] <- .Fortran("loop1", m, n, phi, prob, xPi, logalpha,
lscale, memory0, PACKAGE = "HiddenMarkov")[[7]]
}
}
f <- 699 - max(zzz, na.rm = TRUE)
zzz <- exp(zzz + f)
kMarginalHMM <- log(mean(zzz,na.rm=T)) - f
kMarginalCPM <- lgamma(length(dat[dat == 1]) + 1) + lgamma(length(dat)
- length(dat[dat == 1]) + 1) - lgamma(length(dat) + 2)
BF <- kMarginalHMM - kMarginalCPM
return(BF)
}
LogBayesfactorHMM(Shaq2.data)
```

```
## [1] 0.04519474
```

```
library(lattice)
library(HiddenMarkov)

Shaq3.data <-c(0,0,1,0,0,0,1,0,0,1,1,0,0,0,1,0,0,0,1,0,1,1,0)

LogBayesfactorHMM <- function(dat,intprec=50){
grid1 <- 1 / (2 * (intprec - 1)) + seq(0, 1, length = intprec)[-intprec]
grid2 <- 1 / (2 * intprec) + seq(0, 0.5, length = intprec)[-intprec]
len1 <- length(grid1)
```

```
len2 <- length(grid2)
indices <- which(upper.tri(matrix(TRUE, len1, len1)), arr.ind = TRUE)
zzz <- matrix(0, nrow(indices), len1)
n <- length(dat)
m <- 2L
phi <- as.double(c(0.5, 0.5))
logalpha <- matrix(as.double(rep(0, m * n)), nrow = n)
lscale <- as.double(0)
memory0 <- rep(as.double(0), m)
for(k in seq_len(len2)){
xPi <- matrix(c(1 - grid2[k], grid2[k],
grid2[k], 1 - grid2[k]), 2, 2)
for(i in seq_len(nrow(zzz))){
prob <-
cbind(grid1[indices[i, 1]]^dat * (1 - grid1[indices[i, 1]])^(1 - dat),
grid1[indices[i, 2]]^dat * (1 - grid1[indices[i, 2]])^(1 - dat))
zzz[i, k] <- .Fortran("loop1", m, n, phi, prob, xPi, logalpha,
lscale, memory0, PACKAGE = "HiddenMarkov")[[7]]
}
}
f <- 699 - max(zzz, na.rm = TRUE)
zzz <- exp(zzz + f)
kMarginalHMM <- log(mean(zzz,na.rm=T)) - f
kMarginalCPM <- lgamma(length(dat[dat == 1]) + 1) + lgamma(length(dat)
- length(dat[dat == 1]) + 1) - lgamma(length(dat) + 2)
BF <- kMarginalHMM - kMarginalCPM
return(BF)
}
LogBayesfactorHMM(Shaq3.data)
```

```
## [1] -0.03387582
```

Log Result for O'Neal BEST game 0.1646427 -> 1.178972 (HMM - not worth more than a bare mention)

```
exp(0.1646427)
```

```
## [1] 1.178972
```

Log Result for O'Neal WORST game: 0.01519907 -> 1.015315 (HMM - not worth more than a bare mention)

```
exp(0.01519907)
```

```
## [1] 1.015315
```

Log Result for O'Neal 1st game: 0.4760496 -> 1.609703 (HMM - not worth more than a bare mention)

```
exp(0.4760496)
```

```
## [1] 1.609703
```

Log Result for O'Neal 2nd game: 0.04519474 -> 1.046232 (HMM - not worth more than a bare mention)

```
exp(0.04519474)
```

```
## [1] 1.046232
```

Log Result for O'Neal 3rd game: -0.03387582 -> 1.034456 (CPM)

```r
1/exp(-0.03387582)
```

```
## [1] 1.034456
```

Adrian Brenner for the Institute of Statistic at LMU - under observation of Dr. Volker Schmid.