ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC - KỸ THUẬT MÁY TÍNH

TOÁN RỜI RẠC 1

Bài tập lớn đề tài MÃ ĐỀ

Thống kê & phân tích dữ liệu bằng R

GVHD: NNN
Mã nhóm: N
Nhóm SV: Nguyen Van A- 22102134
Tran Van B - 88471334

TP. HỒ CHÍ MINH, THÁNG 1/2015

Mục lục

Nhật ký làm việc nhóm

Nhật ký làm việc nhóm

Bài báo cáo này trình bày về thống kê và phân tích dữ liệu chiều cao của ca sĩ ở New York Choral Society năm 1979, được chia thành 4 cột lần lượt theo giọng nữ cao, nữ trầm, nam cao và nam trầm.

1 Giới thiệu bài toán

Ta cần phải phân tích dữ liệu để cung cấp các thông tin xác thực, trực quan, mô tả cụ thể, dễ hiểu vấn đề đang phân tích để phục vụ nghiên cứu khoa học. Đặc biệt trong các vấn đề kinh tế-xã hội và khi nghiên cứu số lớn chúng ta cần phải quan tâm đến các công cụ kỹ thuật về phân tích số liệu và biểu đồ.

Phân tích số liệu và biểu đồ thường được tiến hành bằng các phần mềm thông dụng như SAS, SPSS, Stata, Statistica, và S-Plus. Đây là những phần mềm được các công ti phần mềm phát triển và giới thiệu trên thị trường khoảng ba thập niên qua, và đã được các trường đại học, các trung tâm nghiên cứu và công ti kĩ nghệ trên toàn thếgiới sửdụng cho giảng dạy và nghiên cứu. Nhưng vì chi phí để sửdụng các phần mềm này tương đối đắt tiền (có khi lên đến hàng trăm ngàn đô-la mỗi năm). Do đó, các nhà nghiên cứu thống kê trên thế giới đã hợp tác với nhau để phát triển một phần mềm mới, với chủ trương mã nguồn mở, sao cho tất cảcác thành viên trong ngành thống kê học và toán học trên thếgiới có thể sử dụng một cách thống nhất và hoàn toàn miễn phí.

Năm 1996, trong một bài báo quan trọng vềtính toán thống kê, hai nhà thống kê học Ross Ihaka và Robert Gentleman [lúc đó] thuộc Trường đại học Auckland, New Zealand phát hoạmột ngôn ngữmới cho phân tích thống kê mà họ đặt tên là R. Nói một cách ngắn gọn, R là một phần mềm sử dụng cho phân tích thống kê và vẽ biểu đồ. Thật ra, về bản chất, R là ngôn ngữ máy tính đa năng, có thể sửdụng cho nhiều mục tiêu khác nhau, từ tính toán đơn giản, toán học giải trí (recreational mathematics), tính toán ma trận (matrix), đến các phân tích thống kê phức tạp. Vì là một ngôn ngữ, cho nên người ta có thể sử dụng R để phát triển thành các phần mềm chuyên môn cho một vấn đề tính toán cá biệt.

Sơ lược về đề tài : Phân tích đề tài chiều cao của các ca sĩ trong các hội hợp xướng New York vào năm 1979. Với giọng hát từ cao nhất đến thấp nhất với thứ tự Soprano, Alto, Tenor, Bass. Trong đó hai cột đầu tiên là giọng nữ còn hai cột sau là giọng nam. Các dữ liệu ban đầu bao gồm hai bộ phận cho từng phần. Bộ dữ liệu này chỉ báo cáo 1 Soprano, 1 Alto, 1 Tenor, 1 Bass. Với số trường hợp ban đầu là 39.

2 Cơ sở lý thuyết

2.1 Thống kê mô tả

Nói đến thống kê mô tả là nói đến việc mô tả dữ liệu bằng các phép tính và chỉ số thống kê thông thường mà chúng ta đã làm quen qua từ thuở trung học như số trung bình (mean), số trung vị (median), số lớn nhất (max), số nhỏ nhất (min), phương sai (variance), độ lệch chuẩn (standard deviation)...

Trong đó ta làm quen các định nghĩa chưa biết :

- Phương sai của một biến ngẫu nhiên là một độ đo sự phân tán thống kê của biến đó, nó hàm ý các giá trị của biến đó thường ở cách giá trị kỳ vọng bao xa.

- Độ lệch chuẩn, hay độ lệch tiêu chuẩn, là một đại lượng thống kê mô tả dùng để đo mức độ phân tán của một tập dữ liệu đã được lập thành bảng tần số. Có thể tính ra độ lệch chuẩn bằng cách lấy căn bậc hai của phương sai.

- số trung vị (tiếng Anh: median) là một số tách giữa nửa lớn hơn và nửa bé hơn của một mẫu, một quần thể, hay một phân bố xác suất. Nó là giá trị giữa trong một phân bố, mà số số nằm trên hay dưới con số đó là bằng nhau. Điều đó có nghĩa rằng 1/2 quần thể sẽ có các giá trị nhỏ hơn hay bằng số trung vị, và một nửa quần thể sẽ có giá trị bằng hoặc lớn hơn số trung vị.

2.2 Công cụ R

Như đã nói ở trên, R là một công cụ miễn phí dùng để phân tích dữ liệu. Chúng ta có thể sử dụng R để thực hiện các phép toán từ đơn giản đến phức tạp. Những bài toán tiêu biểu: các phép kiểm định thống kê, tính toán trên ma trận, hồi quy tuyến tính, gom cụm dữ liệu, bài toán phân lớp... Và vì R là một ngôn ngữ nên chúng ta có thể viết ứng dụng trên R để giải quyết các vấn đề cụ thể.

- Các hàm của R để tính toán thống kê mô tả:

```
> option (width=100)
# chuyển directory
> setwd ("C:/works/stats")

# đọc dữ liệu vào R
> igfdata <- read.table ("igf.txt", hearder = TRUE, na.string = ".")
> attach (igfdata)

# xem xét các cột số trong dữ liệu
> names (igfdata)
hoặc
> igfdata

# tính trung bình
> mean (age)

# phương sai và độ lệch chuẩn
> var (age)
> sd (age)
```

3 Kết quả phân tích dữ liệu

3.1 Problem 1

- Để đọc được các tệp ".xlsx" và lưu lại dưới dạng R cho việc xử lý sau này, cần cài đặt và sử dụng package **readxl**

```
> install.packages("readxl")
> library(readxl)
```

3.1.1 Đọc các tệp Route*

1. **Đọc dữ liệu**
   - Trước hết cần lấy danh sách tên của các tệp cần đọc trong thư mục và đọc các tệp theo danh sách tên đó
   - Đối với các tệp Route, RouteId được tính là số ở hàng đầu tiên trong tệp đó, ta chỉ lấy

các tệp có RouteId nằm trong khoảng phù hợp theo yêu cầu của đề để làm dataset
- Lấy lại danh sách tên các tệp có RouteId phù hợp và đọc tất cả các tệp trong các thư mục Route* theo danh sách tên này đồng thời xóa các hàng chứa thiếu dữ liệu trong tệp

```
> setwd("D:/BTL_RR/data/RouteCell60")
> RC60.names <- list.files(pattern = ".xlsx")
> rc60 <- lapply(RC60.names,read_excel)
> R.names <- {}
> for (i in 1:length(RC60.names)){
+   if ((as.integer(rc60[[i]][2,1]) <= 117) & (as.integer(rc60[[i]][2,1]) >= 57)){
+     R.names<- c(R.names, RC60.names[i])
+   }
+ }
> RC60 <- lapply(R.names,read_excel)
> for (i in 1:length(RC60)){RC60[[i]]<- na.omit(RC60[[i]])}
> setwd("D:/BTL_RR/data/RouteGPS")
> RGPS <- lapply(R.names,read_excel)
> for (i in 1:length(RGPS)){RGPS[[i]]<- na.omit(RGPS[[i]])}
```

2. **Xử lý dữ liệu***(chuyển polyline của các tệp RouteCell từ tọa độ GPS về tọa độ cell)*
- Cài đặt và sử dụng package **stringr** để tiện lợi trong việc đọc và xử lý các dãy String (do polyline được lưu dưới dạng String)
- Từ các dãy String trong Polyline ta tách ra các tọa độ cần thiết và sử dụng công thức (phụ thuộc vào cell 40 hay cell 60) để chuyển từ tọa độ GPS thành tọa độ cell
- Chú ý rằng trong Polyline có một số điểm bị ngược thứ tự Lat và Long, cần phát hiện và đổi lại thứ tự các điểm này

```
> install.packages("stringr")
> library("stringr")
> for (i in 1:length(RC60)){
+   RC60[[i]]$Polyline
<-sapply(str_extract_all(RC60[[i]]$Polyline,"\\d+\\.*\\d*"),as.numeric)
+   for (k in 1:nrow(RC60[[i]])){
+     a<-{}
+     b<-{}
+     for (j in 1:length(RC60[[i]]$Polyline[[k]])){
+       if (RC60[[i]]$Polyline[[k]][j] > 90)
+         {a<- c(a,ceiling((((RC60[[i]]$Polyline[[k]][j]-106.309795))*10^5)/60))}
+       else {b<- c(b,ceiling((((11.175186-RC60[[i]]$Polyline[[k]][j]))*10^5)/60)))
+     }
+     RC60[[i]]$Polyline[[k]]<-NA
+     for (l in 1:length(a)){
+       RC60[[i]]$Polyline[[k]]
<-c(RC60[[i]]$Polyline[[k]],paste(b[l],a[l],sep = ","))
+     }
+     RC60[[i]]$Polyline[[k]]<-na.omit(RC60[[i]]$Polyline[[k]])
+   }
+ }
> for (i in 1:length(RC60)){
+   for (k in 1:nrow(RC60[[i]])){
```

```
+     RC60[[i]]$Polyline[[k]]
<-c(RC60[[i]]$Polyline[[k]],paste(RC60[[i]]$Lat[k],RC60[[i]]$Lng[k],sep = ","))
+   }
+ }
```

3.1.2 Đọc các tệp Journey*

- Trước hết cần lấy danh sách tên các tệp cần đọc trong thư mục
- Đối với các tệp Journey*, JourneyId được tính là số trong tên tệp ta chỉ lấy danh sách tên các tệp có JourneyId nằm trong khoảng phù hợp theo yêu cầu của đề
- Đọc tất cả các tệp trong các thư mục Journey* theo danh sách tên này để làm dataset

```
> setwd("D:/BTL_RR/data/JourneyCell40")
> Jn40.names <- list.files(pattern = ".xlsx")
> Jn.names <- {}
> for (i in 1:length(Jn40.names)){
+     if ((as.integer(gsub(".xlsx","",Jn40.names[i]))>=201)
&(as.integer(gsub(".xlsx","",Jn40.names[i])) <=670)){
+         Jn.names <- c(Jn.names, Jn40.names[i])
+     }
+ }
> Jn40 <- lapply(Jn.names, read_excel)
> for (i in 1:length(Jn40)){Jn40[[i]]<- na.omit(Jn40[[i]])}
> setwd("D:/BTL_RR/data/JourneyGPS")
> JnGPS <- lapply(Jn.names, read_excel)
> for (i in 1:length(JnGPS)){JnGPS[[i]]<- na.omit(JnGPS[[i]])}
> setwd("D:/BTL_RR/data/JourneyCell60")
> Jn60 <- lapply(Jn.names,read_excel)
> for (i in 1:length(Jn60)){Jn60[[i]]<- na.omit(Jn60[[i]])}
```

3.1.3 Lưu dữ liệu

- Lưu lại các dữ liệu đã đọc dưới dang R để tiện cho việc xử lý về sau

```
> setwd("D:/BTL_RR/Data")
> save(RGPS, file="RGPS.rda")
> save(Jn40, file="Jn40.rda")
> save(JnGPS, file="JnGPS.rda")
> save(Jn60,file="Jn60.rda")
> save(RC60,file="RC60.rda")
```

- Kết quả:

Jn40	Large list (470 elements, 47.4 Mb)
Jn60	Large list (470 elements, 47.4 Mb)
JnGPS	Large list (470 elements, 63.2 Mb)
RC60	Large list (37 elements, 2.3 Mb)
RGPS	Large list (37 elements, 2.3 Mb)

3.2 Problem 2

- Sử dụng dữ liệu có sẵn trong tệp Jn40.rda
- Mỗi data.frame trong Jn40 là 1 xe buýt nên độ dài vector Jn40 là số lượng xe

```
> load("D:/BTL_RR/data/Jn40.rda")
> length(Jn40)
[1] 470
```

- Vậy có 470 xe buýt trong tập mẫu

3.3 Problem 3

- Sử dụng dữ liệu có sẵn trong tệp RC60.rda
- Mỗi data.frame trong RC60 là 1 tuyến xe buýt nên độ dài vector RC60 là số lượng tuyến

```
> load("D:/BTL_RR/data/RC60.rda")
> length(RC60)
[1] 37
```

- Vậy có 37 tuyến xe buýt trong tập mẫu

3.4 Problem 4

3.4.1 Trích xuất tập dữ liệu

- Sử dụng dữ liệu có sẵn trong tệp RC60.rda
- Cần xác định data.frame có RouteId thỏa mãn yêu cầu của đề (số gần với số MD nhất, nếu có nhiều hơn một số có cùng giá trị chênh lệch, trả về Id nhỏ nhất) - Trích xuất data.frame đã được xác định ra để xử lý

```
> load("D:/BTL_RR/data/RC60.rda")
> RouteId<- {}
> for (i in 1:length(RC60))
+     {RouteId<- c(RouteId, as.integer(RC60[[i]][2,1]))}
> nearestId<- min(RouteId[which(abs(RouteId-67)==min(abs(RouteId-67)))])
> for (i in 1:length(RC60)){
+   if (as.integer(RC60[[i]][2,1])==nearestId)
+     (Route<- RC60[[i]])
+ }
```

- Kết quả:

```
> nearestId
[1] 67
```

- Vậy ta chọn data.frame có RouteId là 67

3.4.2 Xử lý dữ liệu

a) - Lấy danh sách tất cả các cell trong Polyline và loại bỏ các cell trùng lặp - Đếm số lượng các cell khác nhau còn lại chính là số cell mà tuyến đó đi qua

```
> a<-do.call(c,Route$Polyline)
> length(unique(a))
[1] 267
```

- Vậy tuyến này đi qua tổng cộng 267 cell khác nhau

b) - Tổng quãng đường di chuyển của tuyến xe có thể tính xấp xỉ bằng tổng của tất cả các giá trị trong cột Distance

```
> sum(as.numeric(Route$Distance))/1000   #km
[1] 36.833
```

- Vậy tổng quãng đường di chuyển của tuyến này là 36,833 km

c) - Mỗi lần xe buýt đi qua một cell ta đếm cell đó thêm 1 lần
- Lấy danh sách tất cả các cell trong Polyline và loại bỏ các cell liên tiếp mà trùng lặp
- Đếm trong danh sách trên mỗi cell xuất hiện bao nhiêu lần và đưa ra danh sách các cell có mặt nhiều hơn một lần

```
> a<-do.call(c,Route$Polyline)
> a<-rle(a)$values
> names(which(table(a)>1))
[1] "676,644" "677,642" "679,632" "679,633" "680,632" "680,644" "681,631"
[8] "682,631" "683,631" "685,630" "686,629" "686,648" "687,629" "687,649"
[15] "688,628" "688,629" "689,650" "689,651" "690,646" "691,647" "691,652"
[22] "692,626" "692,645" "692,647" "692,648" "692,649" "692,650" "693,624"
[29] "693,625" "693,626" "693,644" "693,648" "693,649" "694,622" "695,622"
[36] "695,640" "696,622" "696,639" "697,623" "700,624" "701,641" "701,642"
[43] "702,625" "703,625" "703,643" "704,643" "704,645" "704,649" "708,618"
[50] "708,620" "708,622" "708,624" "709,614" "709,615" "709,617" "710,613"
[57] "710,624" "710,625" "711,612" "711,623" "711,626" "712,610" "712,611"
[64] "713,608" "713,609" "714,604" "714,605" "714,607" "714,608" "715,585"
[71] "715,600" "715,602" "716,582" "716,586" "716,595" "716,597" "717,588"
[78] "717,589" "717,591" "718,572" "718,578" "718,579" "719,570" "719,573"
[85] "719,578" "724,562" "724,563" "726,559" "727,557" "728,556" "729,554"
[92] "729,555" "730,552" "731,551" "731,552" "732,549" "732,550" "734,550"
[99] "737,552" "738,551" "739,549" "744,546" "747,537" "747,538" "748,537"
[106] "749,536" "749,543" "750,536" "751,535" "751,536" "751,537" "751,539"
[113] "752,536" "752,537" "753,535" "753,538" "754,534" "754,540" "755,534"
[120] "761,537" "763,536" "764,536" "765,535" "769,531" "771,528"
```

d) - Tính khoảng cách trung bình đơn giản bằng hàm **mean**, tuy nhiên cần chú ý định dạng của dữ liệu của cột Distance trong data.frame là String, cần chuyển sang định dạng Numeric để tính toán

```
> mean(as.numeric(Route$Distance))
[1] 454.7284   #m
```

- Vậy khoảng cách trung bình giữa các trạm liên tiếp trên hành trình của xe là 454,7 m

3.5 Problem 5

3.5.1 Trích xuất tập dữ liệu

- Sử dụng dữ liệu có sẵn từ tệp RC60.rda
- Ta tạo một data.frame tên Route.all tổng hợp tất cả các tuyến xe trong tập mẫu với các thông số cần thiết bao gồm: Id, s (quãng đường), Celpass (những cell khác nhau mà tuyến đi qua), Celnum (số lượng cell khác nhau mà tuyến đi qua)
- Quãng đường của mỗi tuyến được tính bằng tổng các khoảng cách giữa 2 trạm liên tiếp trên hành trình của tuyến
- Cột Celpass được tạo bằng cách nhập tất cả các dòng polyline của tuyến lại thành 1 vector và khử các giá trị trùng lặp. Độ dài của vector này chính là số lượng cell khác nhau mà tuyến đi qua
- Lưu data.frame này dưới dạng R để xử lý về sau

```
> load("D:/BTL_RR/data/RC60.rda")
> Route.all <- data.frame
(RouteId=rep(0,length(RC60)),s=rep(0,length(RC60)),Celnum=rep(0,length(RC60)))
> Celpass <- {}
> for (n in 1:length(RC60)) {
+   Route.all$RouteId[n]<-RC60[[n]]$Route_Id[1]
+   Route.all$s[n]<-sum(as.numeric(RC60[[n]]$Distance))/1000
+   Celpass[[n]]<-unique(unlist(RC60[[n]]$Polyline))
+   Route.all$Celnum[n]=length(Celpass[[n]])
+ }
> Route.all <- data.frame(Route.all,I(Celpass))
```

- Kết quả:

```
> Route.all
   RouteId        s Celnum      Celpass
1       57 38.49100    218 495,564,....
2       58 19.69300    126 861,635,....
3       59 31.91000    216 681,552,....
4       60 32.86957    195 724,515,....
5       61 23.47300    156 523,615,....
6       62 41.38200    277 553,506,....
7       63 31.43000    227 511,700,....
8       64 32.03500    213 689,789,....
9       65 31.27700    254 570,724,....
10      66 35.45200    252 601,670,....
[ reached getOption("max.print") -- omitted 27 rows ]
```

- Sử dụng data.frame này để tạo một data.frame mới là Routecels.all thống kê theo các cell với 2 thông số: RouteId (liệt kê các tuyến đi qua cell) và Routenum (Số lượng tuyến đi qua cell)

```
> Routecels<-{}
> for (n in 1:nrow(Route.all)){
+   Routecels[[n]]<-data.frame(Cells=Route.all$Celpass[[n]],
RouteId=rep(Route.all$RouteId[n],length(Route.all$Celpass[[n]])))
+ }
> Routecels.all <- do.call(rbind,Routecels)
```

```
> Routecels.all$RouteId <- as.character(Routecels.all$RouteId)
> Routecels.all <- aggregate(RouteId~Cells,Routecels.all,c)
> Routecels.all <- data.frame(Routecels.all,Routenum=rep(0,nrow(Routecels.all)))
> for (n in 1:nrow(Routecels.all)){
+   Routecels.all$Routenum[n]<-length(Routecels.all$RouteId[[n]])
+ }
```

- Lưu các data.frame vừa tạo dưới dạng R để xử lý về sau

```
> setwd("D:/BTL_RR/data")
> save(Route.all,file="Route.all.rda")
> save(Routecels.all,file="Routecels.all.rda")
```

3.5.2 Xủ lý dữ liệu

a) - Sử dụng dữ liệu có sẵn từ tệp Route.all.rda
- Số lượng cell mà một tuyến cho trước đi qua đã được cho trong cột Celnum

```
> load("D:/BTL_RR/data/Route.all.rda")
> RouId=67  #RouteId in dataset
> Route.all[Route.all$RouteId==RouId,c(1,3)]
   RouteId Celnum
11      67    267
```

- Như trên ta thấy tuyến có RouteId bằng 67 đi qua 267 cell khác nhau

b) - Sử dụng dữ liệu có sẵn từ tệp Routecels.all.rda
- Mở data.frame Routecels.all và hiển thị dòng có chứa cell phù hợp với Lat, Long cho trước

```
> load("D:/BTL_RR/data/Routecels.all.rda")
> Lt=599    #Nhap Lat
> Lng=669   #Nhap Long
> Lat_Lng<-paste(Lt,Lng,sep = ",")
> Routecels.all[Routecels.all$Cells==Lat_Lng,]
      Cells                     RouteId Routenum
820 599,669 61, 63, 64, 66, 71, 74, 101        7
```

- Như vậy đối với cell cho trước có Lat và Long tương ứng bằng 599 và 669 có tổng cộng 7 tuyến đi qua là: 61, 63, 64, 66, 71, 74, 101

c,d,e) - Sử dụng dữ liệu có sẵn từ tệp Route.all.rda
- Sắp xếp data.frame Route.all theo thứ tự giảm dần của cột s (quãng đường) và hiển thị các tuyến theo yêu cầu đề bài

```
> load("D:/BTL_RR/data/Route.all.rda")
> Route.all[order(-Route.all$s),][c(1,2),c(1,2)]
   RouteId      s
17      73 65.954
20      76 58.962
```

- Như vậy tuyến số 73 dài nhất, tuyến số 76 dài nhì
- Danh sách các tuyến thuộc một phần ba đầu theo thứ tự chiều dài tuyến giảm dần:

```
> Route.all[order(-Route.all$s),][c(1:round(nrow(Route.all)/3)),c(1,2)]
       RouteId      s
17      73 65.954
20      76 58.962
19      75 58.402
35     113 56.525
14      70 54.140
32     106 50.870
37     116 49.628
33     111 46.856
26      91 46.455
23      79 46.444
16      72 45.147
22      78 42.667
```

f,g,h) - Sử dụng dữ liệu có sẵn từ tệp Route.all.rda
- Sắp xếp data.frame Route.all theo thứ tự giảm dần cột Celnum (số lượng cell chứa trong một tuyến) và hiển thị các tuyến theo yêu cầu đề bài

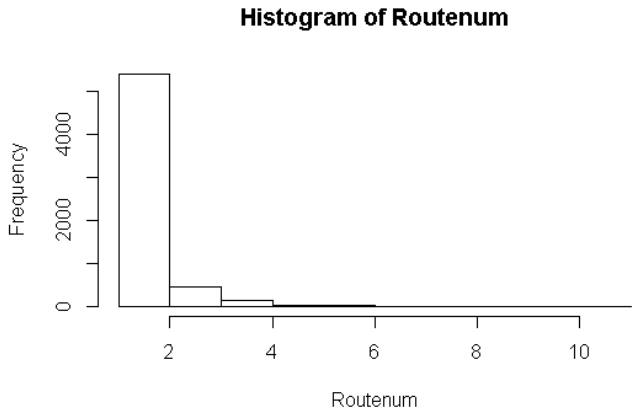
```
> load("D:/BTL_RR/data/Route.all.rda")
> Route.all[order(-Route.all$Celnum),][c(1,2),c(1,3)]
   RouteId Celnum
17      73    455
19      75    392
```

- Như vậy tuyến số 73 chứa nhiều cell nhất, tuyến số 75 chứa nhiều cell nhì
- Danh sách các tuyến thuộc một phần ba đầu theo thứ tự số lượng cell đi qua giảm dần:

```
> Route.all[order(-Route.all$Celnum),][c(1:round(nrow(Route.all)/3)),c(1,3)]
   RouteId Celnum
17      73    455
19      75    392
20      76    343
16      72    333
23      79    298
31     105    297
26      91    290
14      70    283
6       62    277
37     116    271
11      67    267
22      78    266
```

i) - Sử dụng dữ liệu có sẵn từ tệp Routecels.all.rda
- Lập biểu đồ phân bố đơn giản bằng hàm **hist** đối với cột Routenum (số xe buýt chứa cell), tuy nhiên cần sử dụng hàm **table** để có số liệu phân bố cụ thể

```
> load("D:/BTL_RR/data/Routecels.all.rda")
> table(Routecels.all$Routenum)
   1    2    3    4    5    6    7    8    9   10   11
4519  884  460  144   34   37    9    4    1    2    2
> hist(Routecels.all$Routenum, xlab="Routenum",main="Histogram of Routenum")
```



j) - Sử dụng dữ liệu có sẵn từ tệp Routecels.all.rda
- Trước hết cần tìm ra số lượng xe buýt đi qua một cell có tần suất xuất hiện nhiều nhất bằng cách sử dụng hàm **table** kết hợp với hàm **sort** đối với cột Routenum
- Hiện thị các dòng chứa cell có giá trị ở cột Routenum (số lượng xe buýt đi qua cell) bằng với giá trị vừa tìm được

```
> load("D:/BTL_RR/data/Routecels.all.rda")
> Routecels.all[Routecels.all$Routenum
==as.numeric(names(sort(table(Routecels.all$Routenum),T)[1])),]
        Cells RouteId Routenum
7     509,566      57        1
10    516,567      57        1
11    518,567      57        1
12    519,567      57        1
15    524,567      57        1
16    525,567      57        1
17    526,568      57        1
18    528,571      57        1
19    530,573      57        1
20    531,574      57        1
[ reached getOption("max.print") -- omitted 4509 rows ]
```

k)

3.6 Problem 6

3.6.1 Trích xuất tập dữ liệu

- Sử dụng dữ liệu có sẵn từ tệp "Jn40.rda"
- Nhập Journey.Id(JnId) cần khảo sát
- Trích xuất data.frame của Journey.Id từ dataset đồng thời xóa các dòng liên tiếp có cùng Lat và Long (do khi Lat và Long của xe không đổi tức xe vẫn đang di chuyển trong 1 cell, đếm là một lần đi qua cell đó)
- Làm gọn data.frame bằng cách nhóm các dòng có cùng Lat và Long và chuyển cột 3 (Sendtime) thành biến đếm số lần lặp lại của một tọa độ cũng chính là số lần xe đi qua cell đó.

```
> load("D:/BTL_RR/data/Jn40.rda")
> JnId = 201          #201-670
> n = JnId-200
> Celpass
<- Jn40[[n]][c(TRUE,diff(as.numeric(interaction(Jn40[[n]][,c(1,2)]))) != 0),]
> Celpass <- aggregate(list(Times=Celpass[[3]]),Celpass[,c(1,2)],length)
```

- Kết quả:

```
> Celpass
    Lat  Long  Times
1   745  670     1
2   746  670     1
3   745  671     4
4   746  671     4
5   746  673     1
6   746  674     1
7   742  677     1
8   739  681     1
9   736  683     1
10  732  686     1
[reached getOption("max.print") -- omitted 619 rows]
```

3.6.2 Xử lý dữ liệu

a) - Số lượng cell mà xe đó đi qua chính là số hàng của data.frame Celpass vừa trích xuất

```
> Celpass.num=nrow(Celpass)
```

- Kết quả:

```
> Celpass.num
[1] 629
```

b,c) - Sử dụng dữ liệu có sẵn từ tệp JnGPS.rda
- Ta tính quãng đường di chuyển của xe bằng cách tính tổng khoảng cách giữa các tọa độ GPS liên tiếp
- Cần cài đặt package **geosphere** và sử dụng hàm **distVincentyEllipsoid** để tính khoảng cách giữa 2 điểm trên Trái Đất theo tọa độ GPS
- Thời gian di chuyển của xe sẽ được tính là tổng các hiệu của 2 Sendtime liên tiếp nếu có sự

thay đổi về tọa độ giữa 2 hàng liên tiếp đó
- Vận tốc trung bình được tính bằng tổng quãng đường đã di chuyển chia cho tổng thời gian chuyển động

```
> load("D:/BTL_RR/data/JnGPS.rda")
> JnId = 201  #201-670
> n = JnId -200
> library(geosphere)
> s <- sum(distVincentyEllipsoid(JnGPS[[n]][,c(2,1)]))/1000  #km
> t <- 0
> for (i in 1:(nrow(JnGPS[[n]])-1)) {
+     if (all(JnGPS[[n]][i,c(1,2)]==JnGPS[[n]][(i+1),c(1,2)])==F){
+        t <- t+JnGPS[[n]][[3]][i+1]-JnGPS[[n]][[3]][i]
+     }
+ }
> v=s/t*3600    #km/h
```

- Kết quả:

```
> s
[1] 206.1909
> v
[1] 10.13029
```

d) - Ta lập bảng phân bố theo số lần xe buýt đi qua các cell bằng hàm **table** và vẽ Histogram bằng hàm **hist**

```
> table(Celpass$Times)

  1   2   3   4   5   6   7   8   9  13  16
290 156  82  57  25   7   5   3   2   1   1
> hist(Celpass$Times,xlab="Times",main="Times passed of cells")
```

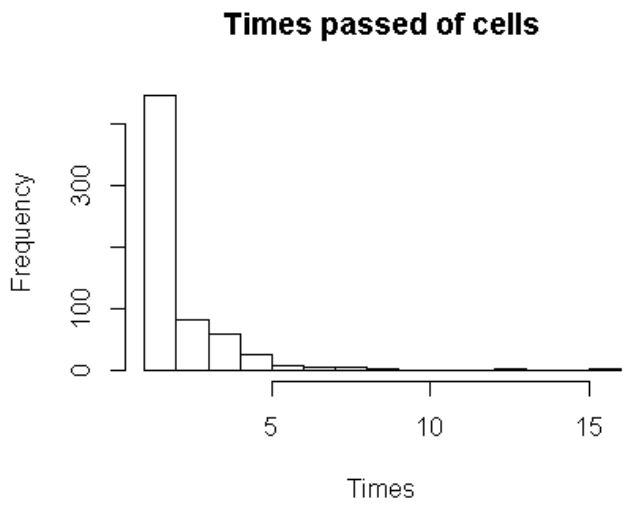
- Kết quả:
- Nhận xét: Số lần xe buýt đi qua một cell chủ yếu là 1 và 2, số lần đi qua một cell nhiều nhất là 16 và chỉ có duy nhất một cell như thế,...v.v...

e) - Ta tìm ra số lần đi qua một cell của xe mà xuất hiện nhiều nhất bằng cách sử dụng kết hợp 2 hàm **sort** và **table**
- Xuất ra các hàng chứa cell mà số lần xe buýt đi qua cell đó có bằng giá trị vừa tìm được

```
> Times.common <- subset(Celpass,Times==as.numeric(names(sort(table(Celpass$Times),T)[1])))
```

- Kết quả:

```
     Lat Long Times
1    745  670     1
2    746  670     1
5    746  673     1
```




```
6    746  674     1
7    742  677     1
8    739  681     1
9    736  683     1
10   732  686     1
11   729  693     1
12   729  694     1
[reached getOption("max.print") -- omitted 280 rows]
```

3.7 Problem 7

3.7.1 Trích xuất tập dữ liệu

- Sử dụng dữ liệu có sẵn từ tệp Jn40.rda - Xử lý dữ liệu của tất cả các xe trong dataset giống như câu 6a đồng thời đối với mỗi data.frame tương ứng với từng xe thêm vào một cột *Bus* để chỉ JourneyId của xe đó
- Kết hợp data.frame của tất cả các xe thành 1 data.frame lớn **Celpass.all**
- Rút gọn data.frame này bằng cách gộp các cell có cùng tọa độ lại, số lần đi qua cell đó bằng tổng số lần ở các data.frame thành phần cộng lại, JourneyId của các xe đi qua cell đó sẽ được liệt kê ở cột *Bus*
- Thêm một cột vào data.frame này để đếm số lượng xe buýt đi qua một cell
- Lưu lại các data.frame thành phần và data.frame tổng dưới dang R để xử lý về sau

```
> load("D:/BTL_RR/data/Jn40.rda")
> Celpass <- {}
> for (n in 1:length(Jn40)) {
+   Celpass[[n]] <- Jn40[[n]][c(TRUE,diff(as.numeric(interaction(Jn40[[n]][,c(1,2)]))) != 0),]
+   Celpass[[n]] <- aggregate(list(Times=Celpass[[n]][[3]]),Celpass[[n]][,c(1,2)],length)
+   Celpass[[n]] <- data.frame(Celpass[[n]],Bus = rep(n+200,nrow(Celpass[[n]])))
```

```
+ }
> Celpass.all <- do.call(rbind,Celpass)
> Celpass.all <- merge(aggregate(Times~Lat+Long,Celpass.all,sum),aggregate(Bus~Lat+Long,Celpass.a
> Bus.num <- {}
> for (i in 1:nrow(Celpass.all)) {
+   Bus.num[i]=length(Celpass.all[[i,4]])}
> Celpass.all <- data.frame(Celpass.all,Bus.num)
> setwd("D:/BTL_RR/data")
> save(Celpass,file="Celpass.rda")
> save(Celpass.all,file="Celpass.all.rda")
```

- Kết quả:

```
          Lat  Long Times  Bus                   Bus.num
1          -1  -277     3  609,613                     2
2          -1  -281     1  660                         1
3          -1    10     1  359                         1
4          -1  2547     1  353                         1
5          -1   307     5  445,475,513,542,543         5
6          -1   308     3  475,513,543                 3
7          -1  4903     1  298                         1
8         -10  -292     1  575                         1
9         -10  -293     4  483,609,660                 3
10        -10  -294     5  483,500,575,609             4
[ reached getOption("max.print") -- omitted 74085 rows ]
```

3.7.2 Xử lý dữ liệu

a) - Sử dụng dữ liệu có sẵn từ tệp Celpass.all.rda
- Nhập vào Lat, Long của cell cho trước, kết quả sẽ đưa ra tổng số lần di chuyển qua cell đó

```
> load("D:/BTL_RR/data/Celpass.all.rda")
> Lt = -1
> Lng = -277
> Celpass.all[Celpass.all$Lat==Lt & Celpass.all$Long==Lng,3]
[1] 3
```

b,c) - Sử dụng dữ liệu có sẵn từ tệp JnGPS.rda
- Thực hiện các bước giống câu 6b,c tuy nhiên cần thêm vòng lặp để tính toán cho tất cả các xe trong dataset. - Sử dụng hàm **mean** để tính giá trị trung bình của tất cả các quãng đường và vận tốc

```
> load("D:/BTL_RR/data/JnGPS.rda")
> s <- {}
> t <- {}
> v <- {}
> library(geosphere)
> for (n in 1:length(JnGPS)) {
```

```
+     k = nrow(JnGPS[[n]])
+     t[n] <- 0
+     s[n] <- sum(distVincentyEllipsoid(JnGPS[[n]][,c(2,1)]))/1000
+     for (i in 1:(k-1)) {
+         if (all(JnGPS[[n]][i,c(1,2)]==JnGPS[[n]][(i+1),c(1,2)])==FALSE){
+             t[n] <- t[n]+JnGPS[[n]][[3]][i+1]-JnGPS[[n]][[3]][i]
+         }
+     }
+     v[n]=s[n]/t[n]*3600
+     t[n]=t[n]/3600
+ }
> Aver.v = mean(na.omit(v))
> Aver.s = mean(s)
```

- Kết quả:

```
> Aver.s
[1] 219.3662
> Aver.v
[1] 17.84721
```

d) - Tính số cell đi qua của tất cả các xe trong dataset và lấy giá trị trung bình cộng, làm tròn về số nguyên dương để được kết quả

```
> load("D:/BTL_RR/data/Celpass.rda")
> Celnum <- {}
> for (n in 1:length(Celpass)){
+   Celnum[n]=nrow(Celpass[[n]])
+ }
> Aver.Celnum = round(mean(Celnum))
```

- Kết quả:

```
> Aver.Celnum
[1] 748
```

- Đến đây, ta tổng hợp lại các dự liệu đã tính toán từ các câu b,c,d vào data.frame Bus.all để tiện cho việc xử lý các câu sau

```
> Bus.all <-data.frame(Bus=1:length(JnGPS),s,t,v,Celnum)
> setwd("D:/BTL_RR/data")
> save(Bus.all,file="Bus.all.rda")
```

e,f) - Sử dụng dữ liệu có sẵn từ tệp Bus.all.rda
- Sắp xếp data.frame Bus.all theo thứ tự giảm dần của cột s (quãng đường) và hiển thị 2 hàng đầu tiên chính là 2 xe buýt có quãng đường di chuyển dài nhất và dài nhì

```
> load("D:/BTL_RR/data/Bus.all.rda")
> Bus.all[order(-Bus.all$s),][c(1,2),]
    Bus        s        t        v Celnum
121 321 624.0570 19.74222 31.61027   3066
153 353 622.7715 16.63556 37.43617   3046
```

g) - Sử dụng dữ liệu có sẵn từ tệp Bus.all.rda
- Sắp xếp data.frame Bus.all theo thứ tự giảm dần của cột s (quãng đường) và hiển thị 1/3 số hàng đầu tiên (đã làm tròn)

```
> load("D:/BTL_RR/data/Bus.all.rda")
> Bus.all[order(-Bus.all$s),][c(1:round(nrow(Bus.all)/3)),]
Bus        s         t        v Celnum
121 321 624.0570 19.742222 31.61027   3066
153 353 622.7715 16.635556 37.43617   3046
312 512 562.2572 19.559722 28.74567   1690
65  265 559.5562 14.111944 39.65125   2128
115 315 496.3697 12.626111 39.31296   2003
113 313 480.6213 11.238056 42.76730   1927
192 392 439.9032 18.776389 23.42853   1276
427 627 399.2530 18.249167 21.87788   1242
348 548 385.4045 20.399722 18.89264   1160
159 359 378.7065 10.061667 37.63854   2777
[ reached getOption("max.print") -- omitted 143 rows ]
```

h,i,j) - Sử dụng dữ liệu có sẵn từ tệp Celpass.all.rda
- Cả 3 câu này ta đều dùng chung một phương pháp tổng quát là sắp xếp các giá trị ở cột Bus.num (số xe buýt đi qua một cell) của data.frame Celpass.all thành 1 dãy và loại đi các giá trị trùng nhau. Từ đó ta dễ dàng tìm ra được giá trị lớn nhất, lớn thứ hai,... của cột Bus.num và trích xuất ra các hàng có giá trị tương ứng

```
> load("D:/BTL_RR/data/Celpass.all.rda")
> Celpass.all[Celpass.all$Bus.num==unique(sort(Celpass.all$Bus.num,T))[1],c(1,2,3,5)]
       Lat Long Times Bus.num
14653 1061  853   407     108
> Celpass.all[Celpass.all$Bus.num==unique(sort(Celpass.all$Bus.num,T))[2],c(1,2,3,5)]
           Lat Long Times Bus.num
14654 1061  854   359      94
14655 1061  855   265      94
> Celpass.all[Celpass.all$Bus.num%in%unique(sort(Celpass.all$Bus.num,T))[c(1,2)],c(1,2,3,5)]
       Lat Long Times Bus.num
14653 1061  853   407     108
14654 1061  854   359      94
14655 1061  855   265      94
```

k) - Sử dụng dữ liệu có sẵn từ tệp Celpass.all.rda
- Sắp xếp data.frame Celpass.all theo thứ tự giảm dần của cột Bus.num và hiển thị 1/3 số hàng đầu tiên (đã làm tròn)

```
> load("D:/BTL_RR/data/Celpass.all.rda")
> Celpass.all[order(-Celpass.all$Bus.num),c(1,2,3,5)][c(1:round(nrow(Celpass.all)/3)),]
       Lat  Long Times Bus.num
14653 1061   853   407     108
14654 1061   854   359      94
14655 1061   855   265      94
61304  827   761   336      93
```

```
61698  832   765   261      93
61386  828   762   312      92
14652 1061   852   184      91
60991  823   759   231      91
60929  822   758   277      90
61462  829   763   196      90
[ reached getOption("max.print") -- omitted 24688 rows ]
```

l) - Sử dụng dữ liệu có sẵn từ tệp Celpass.all.rda
- Ta đếm tần suất của các giá trị trong cột Times (số lần đi qua một cell) của data.frame Celpass.all bằng hàm **table** và lấy giá trị có tần suất lớn nhất - Trích xuất từ data.frame Celpass.all ra các hàng có giá trị ở cột Times tương ứng giá trị vừa tìm được

```
> load("D:/BTL_RR/data/Celpass.all.rda")
> Celpass.all[Celpass.all$Times==as.numeric(names(sort(table(Celpass.all$Times),T)[1])),c(1,
         Lat  Long Times Bus.num
2         -1  -281     1       1
3         -1    10     1       1
4         -1  2547     1       1
7         -1  4903     1       1
8        -10  -292     1       1
11       -10    19     1       1
19      -100   113     1       1
22      -100  2650     1       1
23      -100  5233     1       1
24      -100  5234     1       1
[ reached getOption("max.print") -- omitted 29357 rows ]
```

3.8 Kết quả phân tích

- Thuộc tính thứ 1 - Soprano:

```
> min(Soprano)
[1] 60
> max(Soprano)
[1] 68
> mean(Soprano)
[1] 64.2
> median(Soprano)
[1] 65
> var(Soprano)
[1] 4.168421
> sd(Soprano)
[1] 2.041671
```

Nhận xét: Qua số liệu được phân tích ở trên ta thấy: chiều cao thấp nhất của đối tượng alto là 60 inch, chiều cao cao nhất là 68 inch,phương sai của Soprano thấp (4.7) cho thấy khoảng cách để đạt đến chiều cao kì vọng gần,ở đây số trung vị cho thấy chiếu cao của đối tượng này nằm chủ yếu ở 65 inch,độ lệch chuẩn cho thấy các đối tượng có chênh lệch chiều cao so với chiều cao

trung bình khoảng hơn 2 inch.
- Thuộc tính thứ 2 - Alto:

```
> min(Alto)
[1] 60
> max(Alto)
[1] 72
> mean(Alto)
[1] 64.7
> median(Alto)
[1] 65.5
> var(Alto)
[1] 8.747368
> sd(Alto)
[1] 2.957595
```

Nhận xét: Qua số liệu được phân tích ở trên ta thấy: chiều cao thấp nhất của đối tượng alto là 60 inch, chiều cao cao nhất là 72 inch,phương sai của Alto khá lớn (8.7) cho thấy khoảng cách để đạt đến chiều cao kì vọng khá xa,ở đây số trung vị cho thấy chiếu cao của đối tượng này nằm chủ yếu ở 65.5 inch,độ lệch chuẩn cho thấy các đối tượng có chênh lệch chiều cao so với chiều cao trung bình khoảng 3 inch.
- Thuộc tính thứ 3 - Tenor:

```
> min(Tenor)
[1] 64
> max(Tenor)
[1] 76
> mean(Tenor)
[1] 69.15
> median(Tenor)
[1] 68.5
> var(Tenor)
[1] 10.34474
> sd(Tenor)
[1] 3.216323
```

Nhận xét: Qua số liệu được phân tích ở trên ta thấy: chiều cao thấp nhất của đối tượng alto là 64 inch, chiều cao cao nhất là 76 inch,phương sai của Tenor lớn (10.3) cho thấy khoảng cách để đạt đến chiều cao kì vọng rất xa,ở đây số trung vị cho thấy chiếu cao của đối tượng này nằm chủ yếu ở 69.15 inch,độ lệch chuẩn cho thấy các đối tượng có chênh lệch chiều cao o với chiều cao trung bình khoảng hơn 3 inch.
- Thuộc tính thứ 4 - Bass:

```
>  min(Bass)
[1] 66
> max(Bass)
[1] 75
> mean(Bass)
[1] 70.4
> median(Bass)
[1] 70.5
```

```
> var(Bass)
[1] 5.305263
> sd(Bass)
[1] 2.303316
```

Nhận xét: Qua số liệu được phân tích ở trên ta thấy: chiều cao thấp nhất của đối tượng alto là 66 inch, chiều cao cao nhất là 75 inch,phương sai của Bass mức trung bình (5.3) cho thấy khoảng cách để đạt đến chiều cao kì vong,ở đây số trung vị cho thấy chiếu cao của đối tượng này nằm chủ yếu ở 70.4 inch,độ lệch chuẩn cho thấy các đối tượng có chênh lệch chiều cao so với chiều cao trung bình khoảng hơn 2 inch.

4 Kết luận

Trong báo cáo này chúng tôi đã trình bày về R với định nghĩa, ứng dụng về R. Sử dụng các hàm của R để thực hiện việc thống kê mô tả tập dữ liệu là phân tích chiều cao của nam và nữ trong dàn hợp xướng New York vào năm 1979. Qua đó đã làm rõ được các thông số về chiều cao min, max, phương sai, độ lệch chuẩn... Và cũng đã chỉ ra được ý nghĩa tầm quan trọng của ngôn ngữ R va ứng dụng của nó để phân tích dữ liệu.

Tài liệu

[1] Giáo sư Nguyễn Văn Tuấn "<**http://www.nguyenvantuan.net/**>", *xem ngày :* 24-29/05/2012.

[2] wikipedia. "**link: http://vi.wikipedia.org/**", *phương sai,độ lệch chuẩn, số trung vị*, lần truy cập cuối: 29/05/2012.