Geolocation

Predicting Location via Indoor Positioning Systems

The Raw Data

```
file_offline <- file.path(data.dir, "offline.final.trace.txt")
file_online <- file.path(data.dir, "online.final.trace.txt")

raw_offline <- read_lines(file_offline)
raw_online <- read_lines(file_online)
```

Sanity Check

```
# number of comment lines in the data
sum(substr(raw_offline, 1, 1) == "#")
```

```
[1] 5312
```

```
# total number of lines in the data
length(raw_offline)
```

```
[1] 151392
```

Data Pre-processing

Generate read data function for preprocessing training and test data.

```
processLine <- function(x) {
   tokens <- strsplit(x, "[;=,]")[[1]]

   if(length(tokens) == 10)
      return(NULL)

   tmp <- matrix(tokens[ - (1:10)], ncol = 4, byrow = T)
   cbind(matrix(tokens[c(2, 4, 6:8, 10)], nrow = nrow(tmp),
```

```
                ncol = 6, byrow = T), tmp)
}

validLines <- function(data) {
   substr(data, 1, 1) != "#"
}

roundOrientation <- function(angles) {
   refs = seq(0, by = 45, length = 9)
   q <- sapply(angles, function(o) which.min(abs(o - refs)))
   c(refs[1:8], 0)[q]
}

readData <- function(file, submacs = macs) {

   lines <- read_lines(file)

   valid_lines <- lines[ validLines(lines) ]

   processed_lines <- lapply(valid_lines, processLine)

   data <- as.data.table(do.call("rbind", processed_lines),
                         stringsAsFactors = F)

   names(data) <- c("time", "scanMac", "posX", "posY", "posZ",
                    "orientation", "mac", "signal",
                    "channel", "type")

   numVars <- c("time", "posX", "posY", "posZ",
                "orientation", "signal")

   data[, (numVars) := lapply(.SD, as.numeric), .SDcols = numVars]

   data <- data[ data$type == 3, ]
   data[, type := NULL]

   data[, rawTime := time]
   data[, time := time/1000]
   class(data$time) = c("POSIXt", "POSIXct")

   # drop scanMac & posZ
   data[, `:=`(scanMac = NULL, posZ = NULL)]

   data$angle = roundOrientation(data$orientation)
```

```
   data$channel = NULL

   data$posXY <- paste(data$posX, data$posY, sep = "-")

   return(data[mac %in% submacs])
}
```

Test Pre-processor

```
lines <- processLine(raw_offline[4:20])
lines
```

```
      [,1]            [,2]                [,3]  [,4]  [,5]  [,6]
 [1,] "1139643118358" "00:02:2D:21:0F:33" "0.0" "0.0" "0.0" "0.0"
 [2,] "1139643118358" "00:02:2D:21:0F:33" "0.0" "0.0" "0.0" "0.0"
 [3,] "1139643118358" "00:02:2D:21:0F:33" "0.0" "0.0" "0.0" "0.0"
 [4,] "1139643118358" "00:02:2D:21:0F:33" "0.0" "0.0" "0.0" "0.0"
 [5,] "1139643118358" "00:02:2D:21:0F:33" "0.0" "0.0" "0.0" "0.0"
 [6,] "1139643118358" "00:02:2D:21:0F:33" "0.0" "0.0" "0.0" "0.0"
 [7,] "1139643118358" "00:02:2D:21:0F:33" "0.0" "0.0" "0.0" "0.0"
 [8,] "1139643118358" "00:02:2D:21:0F:33" "0.0" "0.0" "0.0" "0.0"
 [9,] "1139643118358" "00:02:2D:21:0F:33" "0.0" "0.0" "0.0" "0.0"
[10,] "1139643118358" "00:02:2D:21:0F:33" "0.0" "0.0" "0.0" "0.0"
[11,] "1139643118358" "00:02:2D:21:0F:33" "0.0" "0.0" "0.0" "0.0"
      [,7]                [,8]  [,9]          [,10]
 [1,] "00:14:bf:b1:97:8a" "-38" "2437000000" "3"
 [2,] "00:14:bf:b1:97:90" "-56" "2427000000" "3"
 [3,] "00:0f:a3:39:e1:c0" "-53" "2462000000" "3"
 [4,] "00:14:bf:b1:97:8d" "-65" "2442000000" "3"
 [5,] "00:14:bf:b1:97:81" "-65" "2422000000" "3"
 [6,] "00:14:bf:3b:c7:c6" "-66" "2432000000" "3"
 [7,] "00:0f:a3:39:dd:cd" "-75" "2412000000" "3"
 [8,] "00:0f:a3:39:e0:4b" "-78" "2462000000" "3"
 [9,] "00:0f:a3:39:e2:10" "-87" "2437000000" "3"
[10,] "02:64:fb:68:52:e6" "-88" "2447000000" "1"
[11,] "02:00:42:55:31:00" "-84" "2457000000" "1"
```

```
offline_valid <- raw_offline[validLines(raw_offline)]

head(offline_valid)
```

```
[1] "t=1139643118358;id=00:02:2D:21:0F:33;pos=0.0,0.0,0.0;degree=0.0;00:14:bf:b1:97:8a=-38,2437
[2] "t=1139643118744;id=00:02:2D:21:0F:33;pos=0.0,0.0,0.0;degree=0.0;00:14:bf:b1:97:8a=-38,2437
[3] "t=1139643119002;id=00:02:2D:21:0F:33;pos=0.0,0.0,0.0;degree=0.0;00:14:bf:b1:97:8a=-38,2437
```

```
[4] "t=1139643119263;id=00:02:2D:21:0F:33;pos=0.0,0.0,0.0;degree=0.0;00:14:bf:b1:97:8a=-38,2437
[5] "t=1139643119538;id=00:02:2D:21:0F:33;pos=0.0,0.0,0.0;degree=0.0;00:14:bf:b1:97:8a=-46,2437
[6] "t=1139643119818;id=00:02:2D:21:0F:33;pos=0.0,0.0,0.0;degree=0.0;00:14:bf:b1:97:8a=-37,2437
```

```
length(offline_valid)
```

```
[1] 146080
```

```
tmp <- lapply(offline_valid[1:17], processLine)

sapply(tmp, nrow)
```

```
 [1] 11 10 10 11  9 10  9  9 10 11 11  9  9  9  8 10 14
```

Dry Run

```
offline_test <- as.data.table(do.call("rbind", tmp))

offline_test
```

```
                V1                 V2  V3  V4  V5  V6                 V7  V8
  1: 1139643118358 00:02:2D:21:0F:33 0.0 0.0 0.0 0.0 00:14:bf:b1:97:8a -38
  2: 1139643118358 00:02:2D:21:0F:33 0.0 0.0 0.0 0.0 00:14:bf:b1:97:90 -56
  3: 1139643118358 00:02:2D:21:0F:33 0.0 0.0 0.0 0.0 00:0f:a3:39:e1:c0 -53
  4: 1139643118358 00:02:2D:21:0F:33 0.0 0.0 0.0 0.0 00:14:bf:b1:97:8d -65
  5: 1139643118358 00:02:2D:21:0F:33 0.0 0.0 0.0 0.0 00:14:bf:b1:97:81 -65
 ---
166: 1139643122647 00:02:2D:21:0F:33 0.0 0.0 0.0 0.0 00:0f:a3:39:e0:4b -79
167: 1139643122647 00:02:2D:21:0F:33 0.0 0.0 0.0 0.0 00:0f:a3:39:e0:4b -80
168: 1139643122647 00:02:2D:21:0F:33 0.0 0.0 0.0 0.0 00:0f:a3:39:e2:10 -83
169: 1139643122647 00:02:2D:21:0F:33 0.0 0.0 0.0 0.0 00:0f:a3:39:dd:cd -65
170: 1139643122647 00:02:2D:21:0F:33 0.0 0.0 0.0 0.0 02:00:42:55:31:00 -84
             V9 V10
  1: 2437000000   3
  2: 2427000000   3
  3: 2462000000   3
  4: 2442000000   3
  5: 2422000000   3
 ---
166: 2462000000   3
167: 2462000000   3
168: 2437000000   3
169: 2412000000   3
170: 2457000000   1
```

Process

```
# Enter Debug Context

# options(error = recover, warn = 2)

offline_data <- lapply(offline_valid, processLine)
offline <- as.data.table(do.call("rbind", offline_data),
                         stringsAsFactors = F)

names(offline) <- c("time", "scanMac", "posX", "posY", "posZ",
                    "orientation", "mac", "signal",
                    "channel", "type")

numVars <- c("time", "posX", "posY", "posZ",
             "orientation", "signal")

dim(offline)
```

```
[1] 1181628      10
```

```
offline[, (numVars) := lapply(.SD, as.numeric), .SDcols = numVars]

offline <- offline[ offline$type == 3, ]
offline[, type := NULL]

offline[, rawTime := time]
offline[, time := time/1000]
class(offline$time) = c("POSIXt", "POSIXct")

# options(error = recover, warn = 1)
```

Data Cleaning

```
summary(offline)
```

```
      time                         scanMac              posX
 Min.   :2006-02-11 02:31:58   Length:978443      Min.   : 0.00
 1st Qu.:2006-02-11 08:21:27   Class :character   1st Qu.: 2.00
 Median :2006-02-11 14:57:58   Mode  :character   Median :12.00
 Mean   :2006-02-16 09:57:37                      Mean   :13.52
 3rd Qu.:2006-02-19 09:52:40                      3rd Qu.:23.00
 Max.   :2006-03-09 15:41:10                      Max.   :33.00
      posY             posZ    orientation        mac
 Min.   : 0.000   Min.   :0   Min.   :  0.0   Length:978443
```

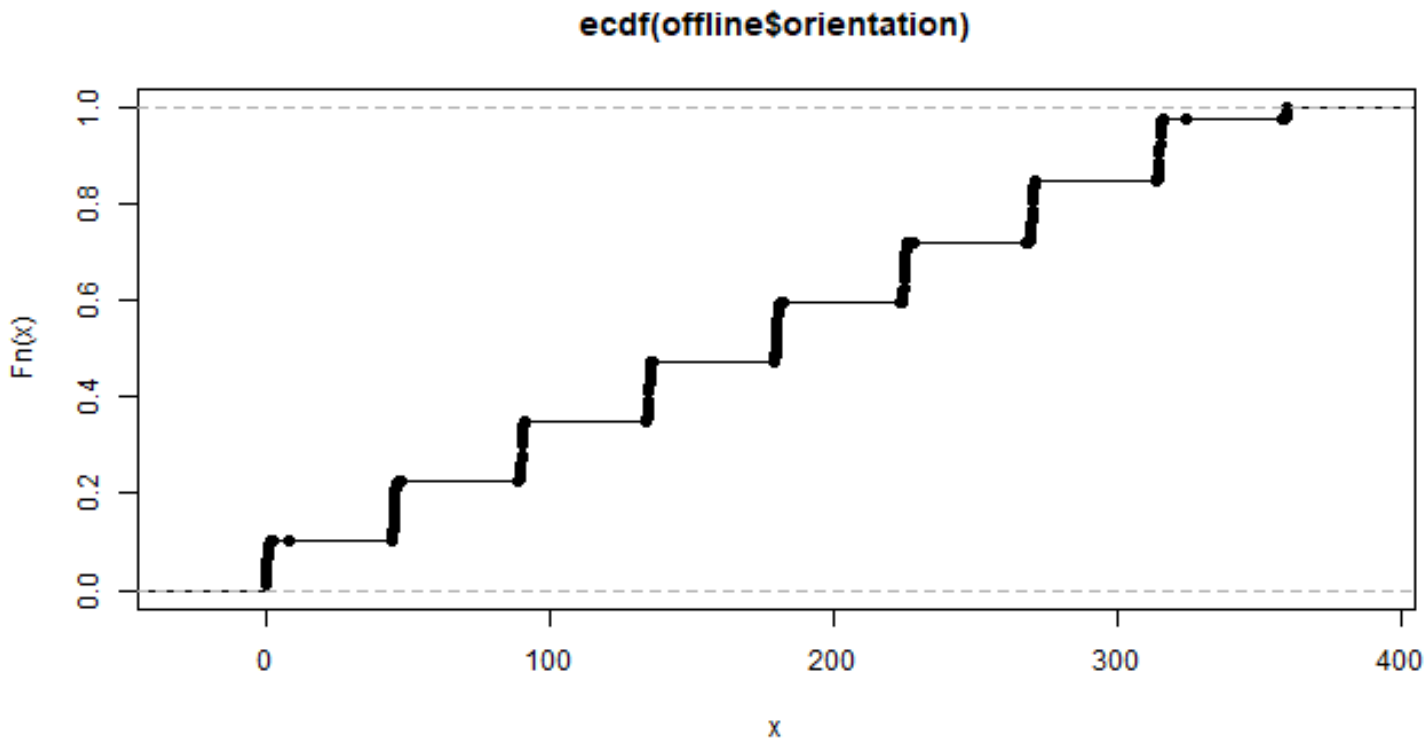
```
 1st Qu.: 3.000   1st Qu.:0   1st Qu.: 90.0   Class :character
 Median : 6.000   Median :0   Median :180.0   Mode  :character
 Mean   : 5.897   Mean   :0   Mean   :167.2
 3rd Qu.: 8.000   3rd Qu.:0   3rd Qu.:270.0
 Max.   :13.000   Max.   :0   Max.   :359.9
     signal         channel             rawTime
 Min.   :-99.0   Length:978443      Min.   :1.140e+12
 1st Qu.:-69.0   Class :character   1st Qu.:1.140e+12
 Median :-60.0   Mode  :character   Median :1.140e+12
 Mean   :-61.7                      Mean   :1.140e+12
 3rd Qu.:-53.0                      3rd Qu.:1.140e+12
 Max.   :-25.0                      Max.   :1.142e+12
```

Orientation Exploration

```
length(unique(offline$orientation))
```

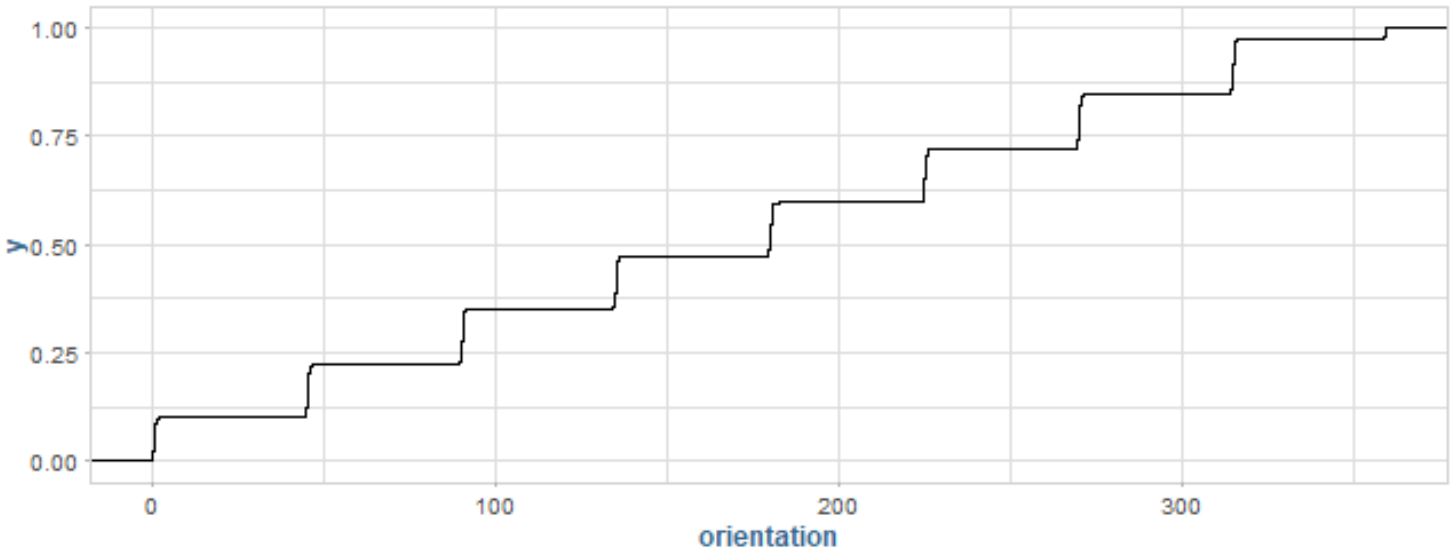
```
[1] 203
```

```
plot(ecdf(offline$orientation))
```

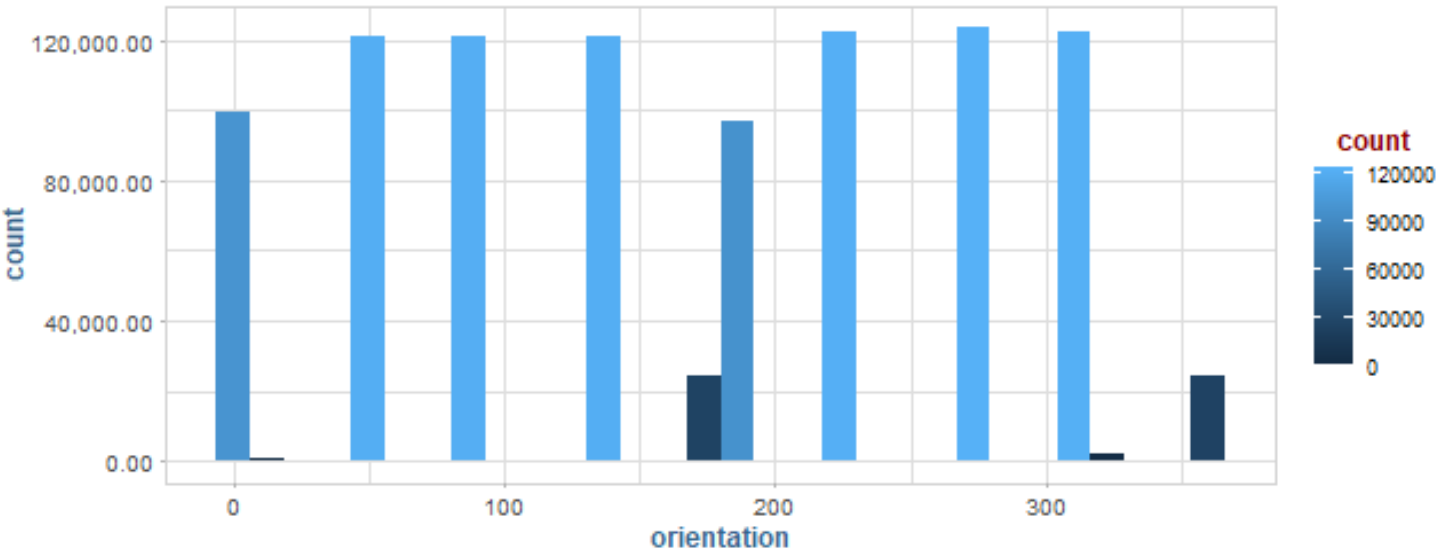



```
ggplot(offline, aes(orientation)) +
   stat_ecdf() +
```

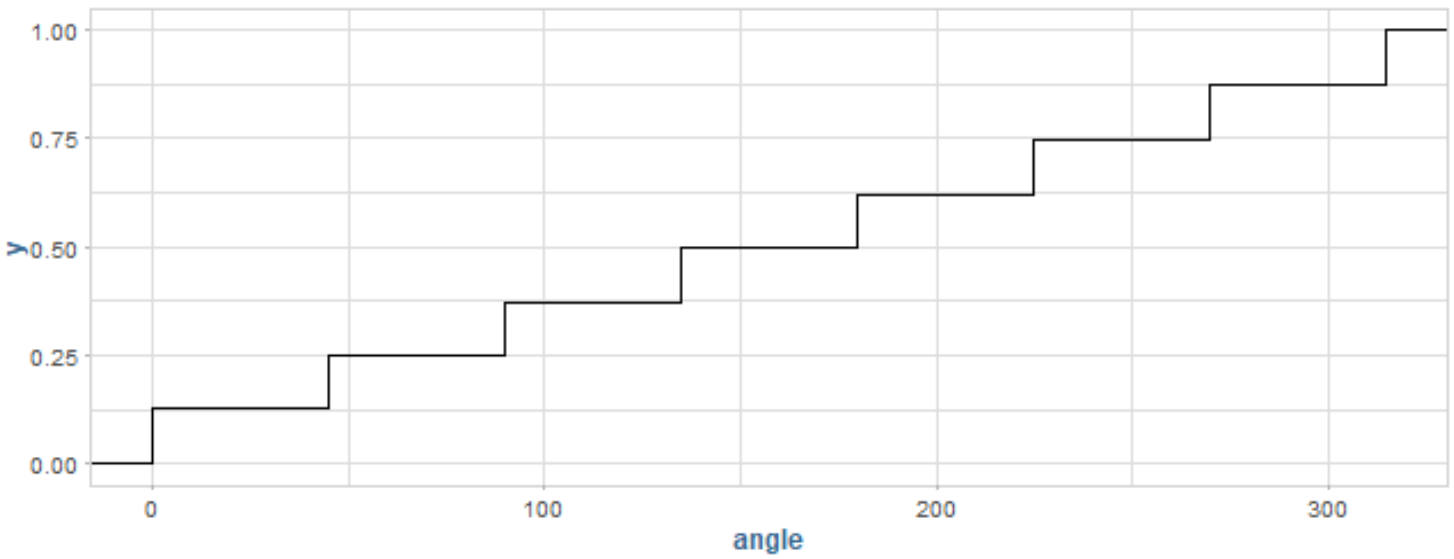
```
   labs("Orientation")
```



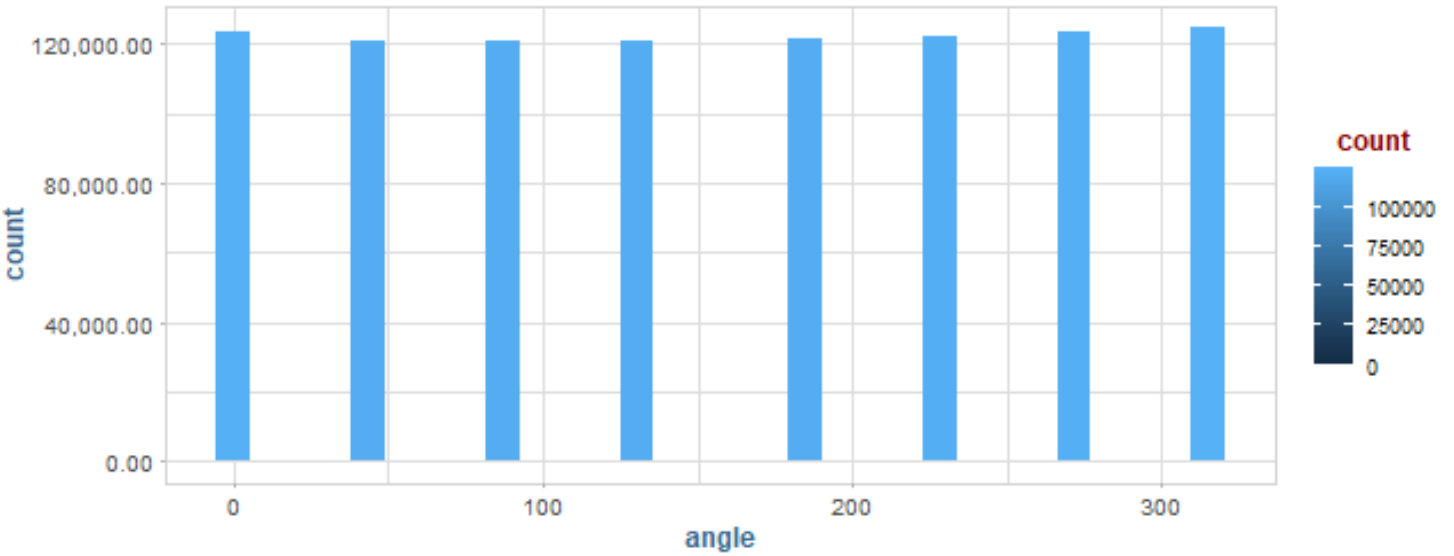

```
ggplot(offline, aes(orientation, fill = ..count..)) +
   geom_histogram(bins = 30) +
   scale_y_continuous(labels = comma) +
   labs("Orientation Value Clusters")
```



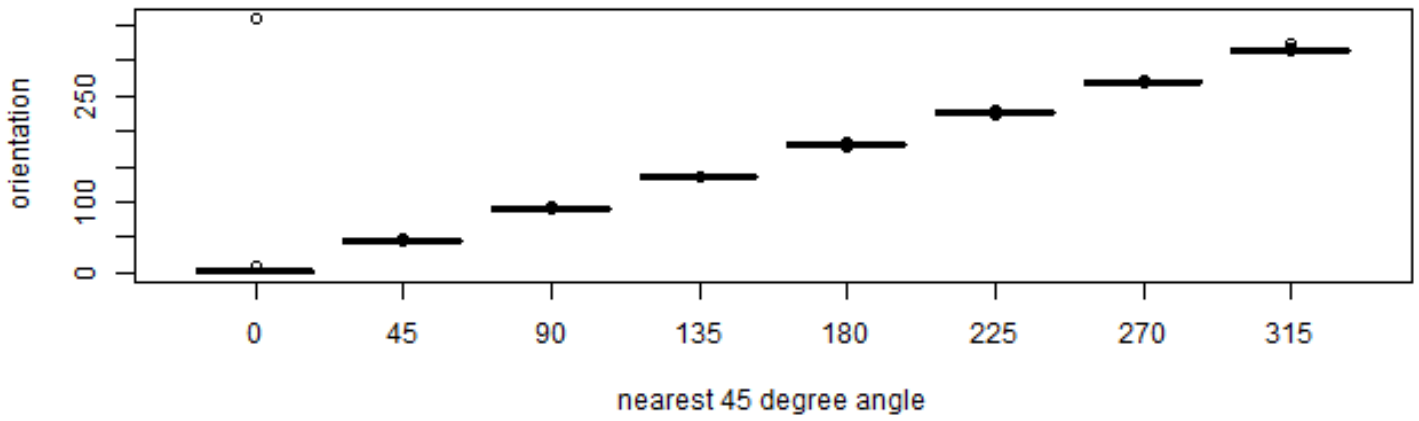

```
offline$angle <- roundOrientation(offline$orientation)

# angle = cleaned orientation column
ggplot(offline, aes(angle)) +
   stat_ecdf()
```



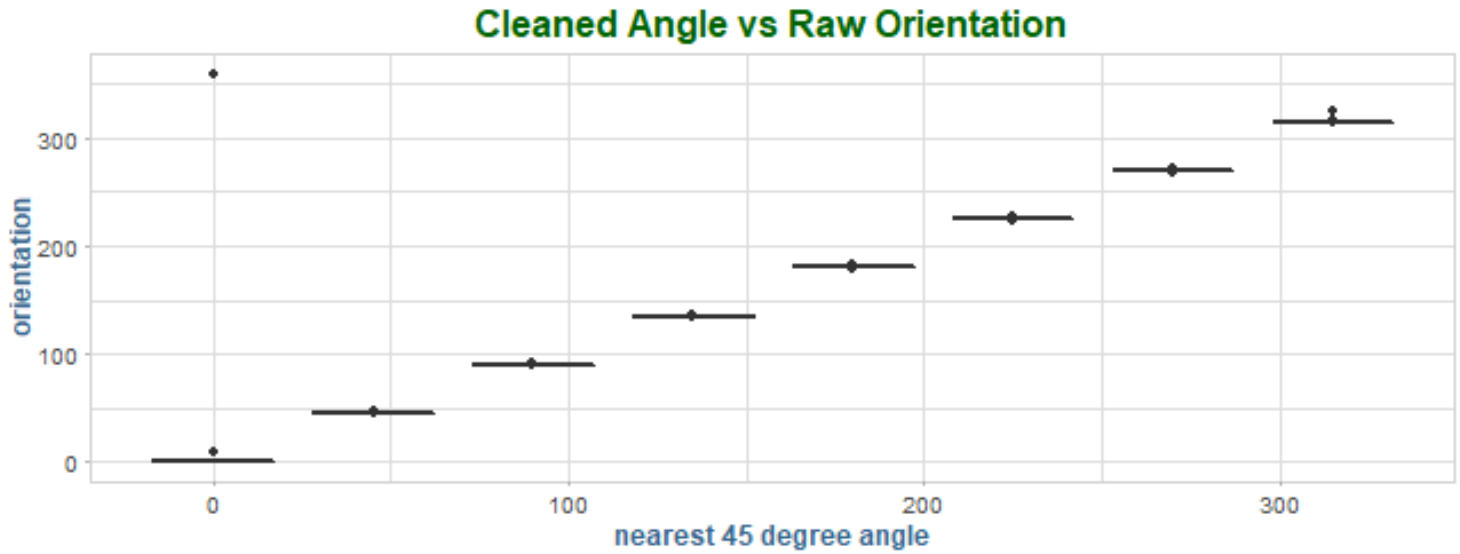

```
ggplot(offline, aes(angle, fill = ..count..)) +
   geom_histogram(bins = 30) +
   scale_y_continuous(labels = comma) +
   labs("Cleaned Up Angles")
```




```
with(offline, boxplot(orientation ~ angle,
                      xlab = "nearest 45 degree angle",
                      ylab = "orientation"))
```




```
ggplot(offline, aes(angle, orientation, group = angle)) +
   geom_boxplot() +
   labs(title = "Cleaned Angle vs Raw Orientation",
        x = "nearest 45 degree angle",
        y = "orientation")
```



MAC Address

```
c(length(unique(offline$mac)), length(unique(offline$channel)))
```

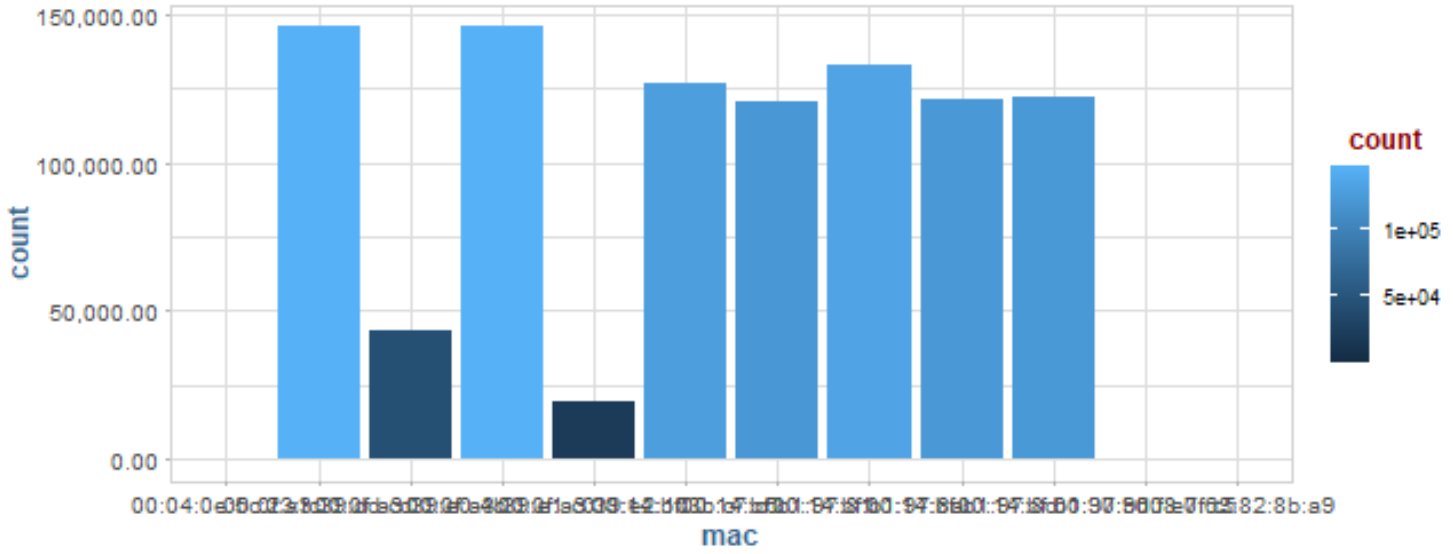
```
[1] 12  8
```

```
table(offline$mac)
```

```
00:04:0e:5c:23:fc 00:0f:a3:39:dd:cd 00:0f:a3:39:e0:4b 00:0f:a3:39:e1:c0
              418            145619             43508            145862
00:0f:a3:39:e2:10 00:14:bf:3b:c7:c6 00:14:bf:b1:97:81 00:14:bf:b1:97:8a
            19162            126529            120339            132962
00:14:bf:b1:97:8d 00:14:bf:b1:97:90 00:30:bd:f8:7f:c5 00:e0:63:82:8b:a9
           121325            122315               301               103
```

```
ggplot(offline, aes(mac, fill = ..count..)) +
   geom_histogram(stat = "count", bins = 30) +
   scale_y_continuous(labels = comma)
```

```
Warning: Ignoring unknown parameters: binwidth, bins, pad
```




```
# keep only the top 7 MAC address data points

dim(offline)
```

```
[1] 978443     11
```

```
offline_macs <- names(sort(table(offline$mac), decreasing = T)[1:7])

dim(offline)
```

```
[1] 978443     11
```

```
macChannel <- with(offline, table(mac, channel))
apply(macChannel, 1, function(x) sum(x > 0))
```

```
00:04:0e:5c:23:fc 00:0f:a3:39:dd:cd 00:0f:a3:39:e0:4b 00:0f:a3:39:e1:c0
```

```
                1                 1                 1                 1
00:0f:a3:39:e2:10 00:14:bf:3b:c7:c6 00:14:bf:b1:97:81 00:14:bf:b1:97:8a
                1                 1                 1                 1
00:14:bf:b1:97:8d 00:14:bf:b1:97:90 00:30:bd:f8:7f:c5 00:e0:63:82:8b:a9
                1                 1                 1                 1
```

```
# mac and channel are 1:1, we can remove channel
#offline$channel := NULL
```

Exploring the Position of the Hand-Held Device

```
locDF <- with(offline,
              by(offline, list(posX, posY), function(x) x))

length(locDF)
```

```
[1] 476
```

```
sum(sapply(locDF, is.null))
```

```
[1] 310
```

```
locDF <- locDF[ !sapply(locDF, is.null)]

length(locDF)
```

```
[1] 166
```

```
locCounts <- sapply(locDF, nrow)

locCounts <- sapply(locDF,
                    function(df)
                      c(df[1, c("posX", "posY")], count = nrow(df)))

class(locCounts)
```

```
[1] "matrix"
```

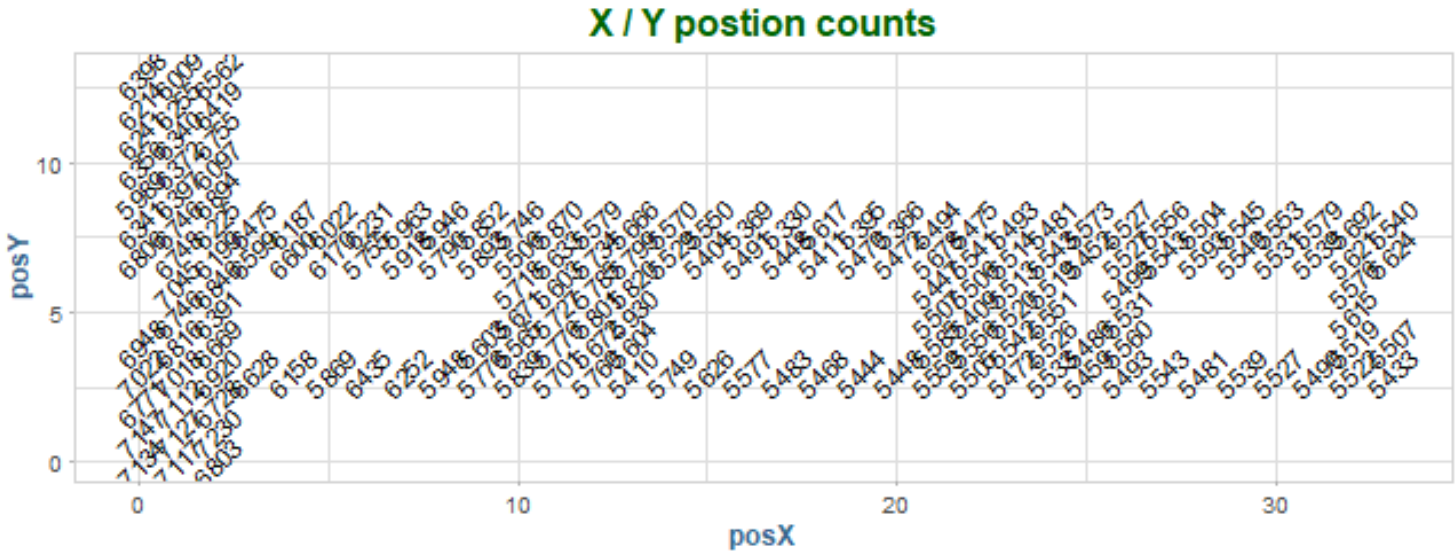
```
dim(locCounts)
```

```
[1]   3 166
```

```
locCounts[ , 1:8]
```

```
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
posX     0    1    2    0    1    2    0    1
posY     0    0    0    1    1    1    2    2
count 7134 7117 6803 7147 7127 7230 6771 7112
```

```
locCountsDF <- as.data.table(t(locCounts))

locCountsDF$posX <- unlist(locCountsDF$posX)
locCountsDF$posY <- unlist(locCountsDF$posY)

ggplot(locCountsDF, aes(posX, posY, label = count)) +
   geom_text(angle = 45) +
   labs(title = "X / Y postion counts")
```

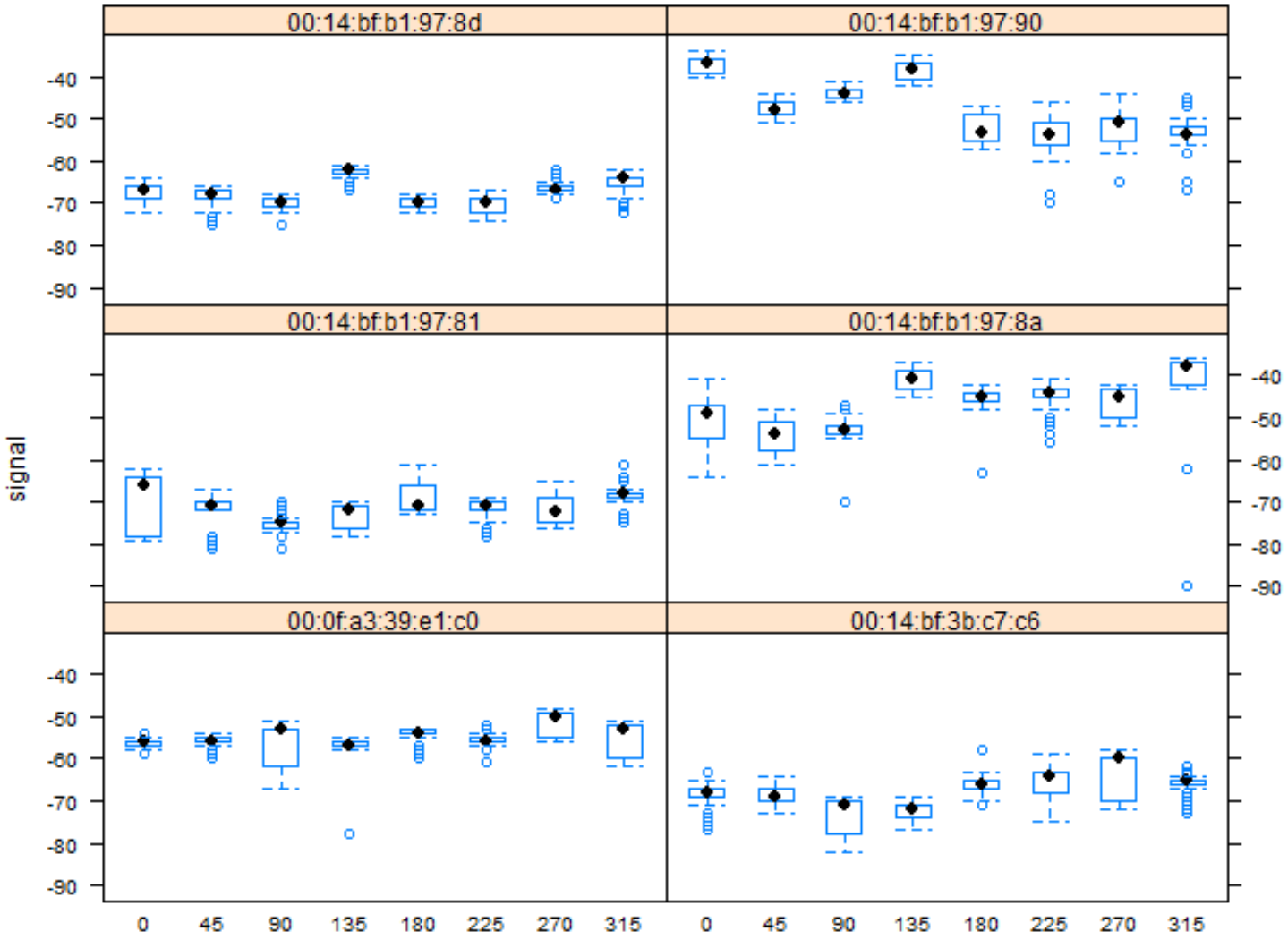


Final Data Prep

```
offline <- readData(file_offline, offline_macs)
```

Signal Strength

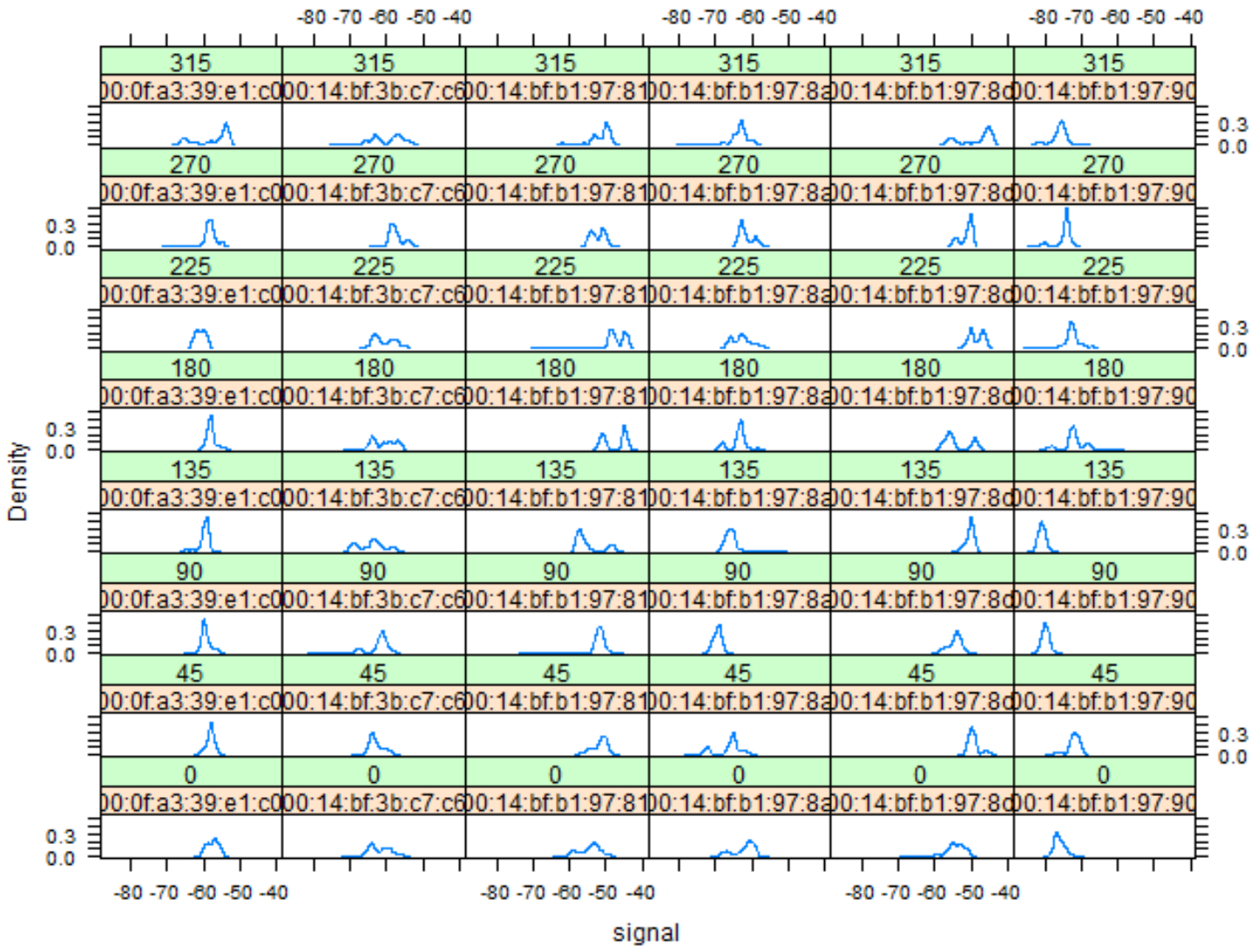
```
bwplot(signal ~ factor(angle) | mac, data = offline,
       subset = posX == 2 & posY == 12 &
          mac != "00:0f:a3:39:dd:cd",
       layout = c(2, 3))
```




```
summary(offline$signal)
```

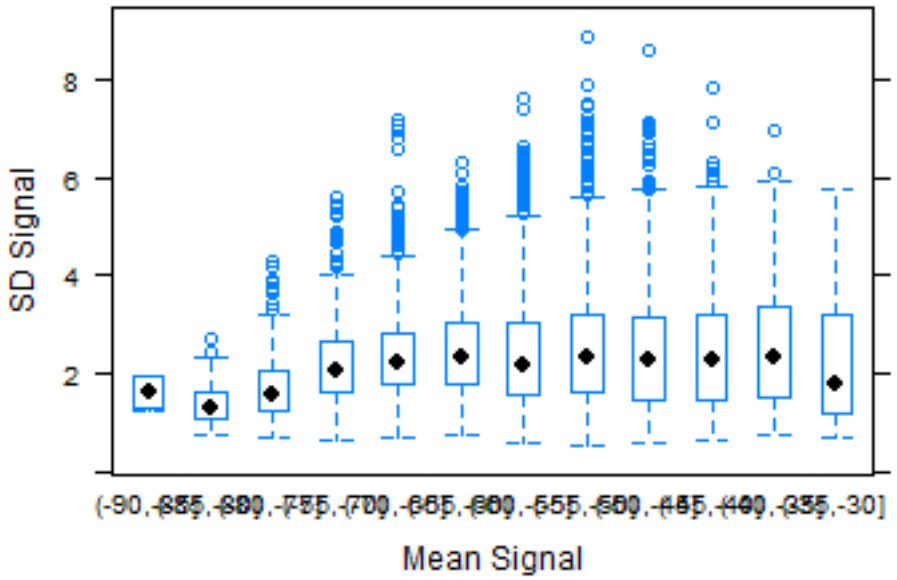
```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 -98.00  -67.00  -59.00  -59.92  -53.00  -25.00
```

```
densityplot(~ signal | mac + factor(angle), data = offline,
            subset = posX == 24 & posY == 4 &
               mac != "00:0f:a3:39:dd:cd",
            bw = 0.5, plot.points = F)
```

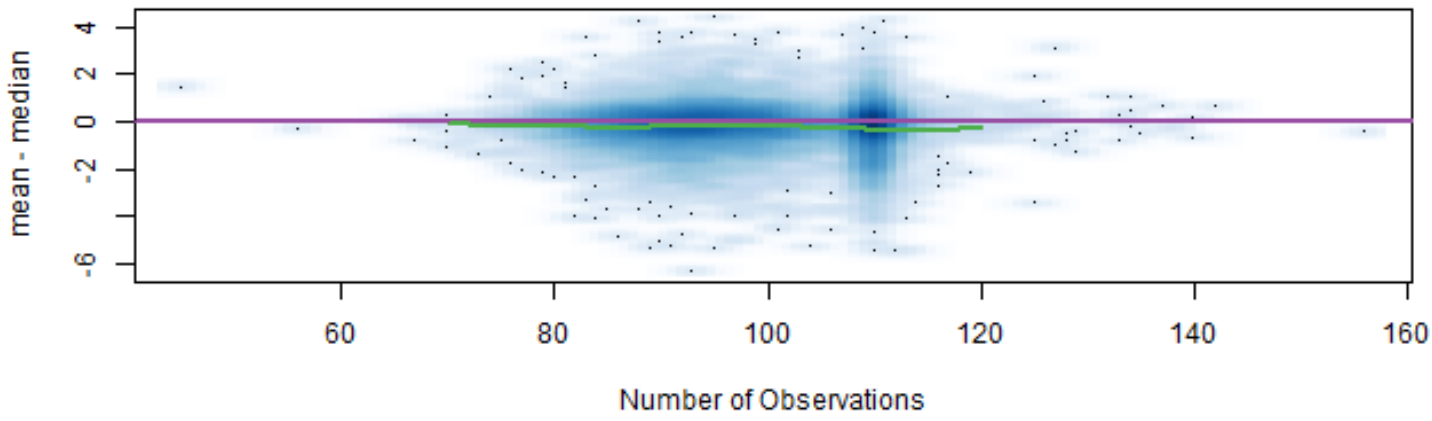



```
byLocAngleAP <- with(offline,
                     by(offline, list(posXY, angle, mac),
                        function(x) x))
signalSummary <-
   lapply(byLocAngleAP,
          function(oneLoc) {
             ans = oneLoc[1, ]
             ans$medSignal = median(oneLoc$signal)
             ans$avgSignal = mean(oneLoc$signal)
             ans$num = length(oneLoc$signal)
             ans$sdSignal = sd(oneLoc$signal)
             ans$iqrSignal = IQR(oneLoc$signal)
             ans
          })
```

```
offlineSummary <- do.call("rbind", signalSummary)

breaks <- seq(-90, -30, by = 5)

bwplot(sdSignal ~ cut(avgSignal, breaks = breaks),
       data = offlineSummary,
       subset = mac != "00:0f:a3:39:dd:cd",
       xlab = "Mean Signal", ylab = "SD Signal")
```

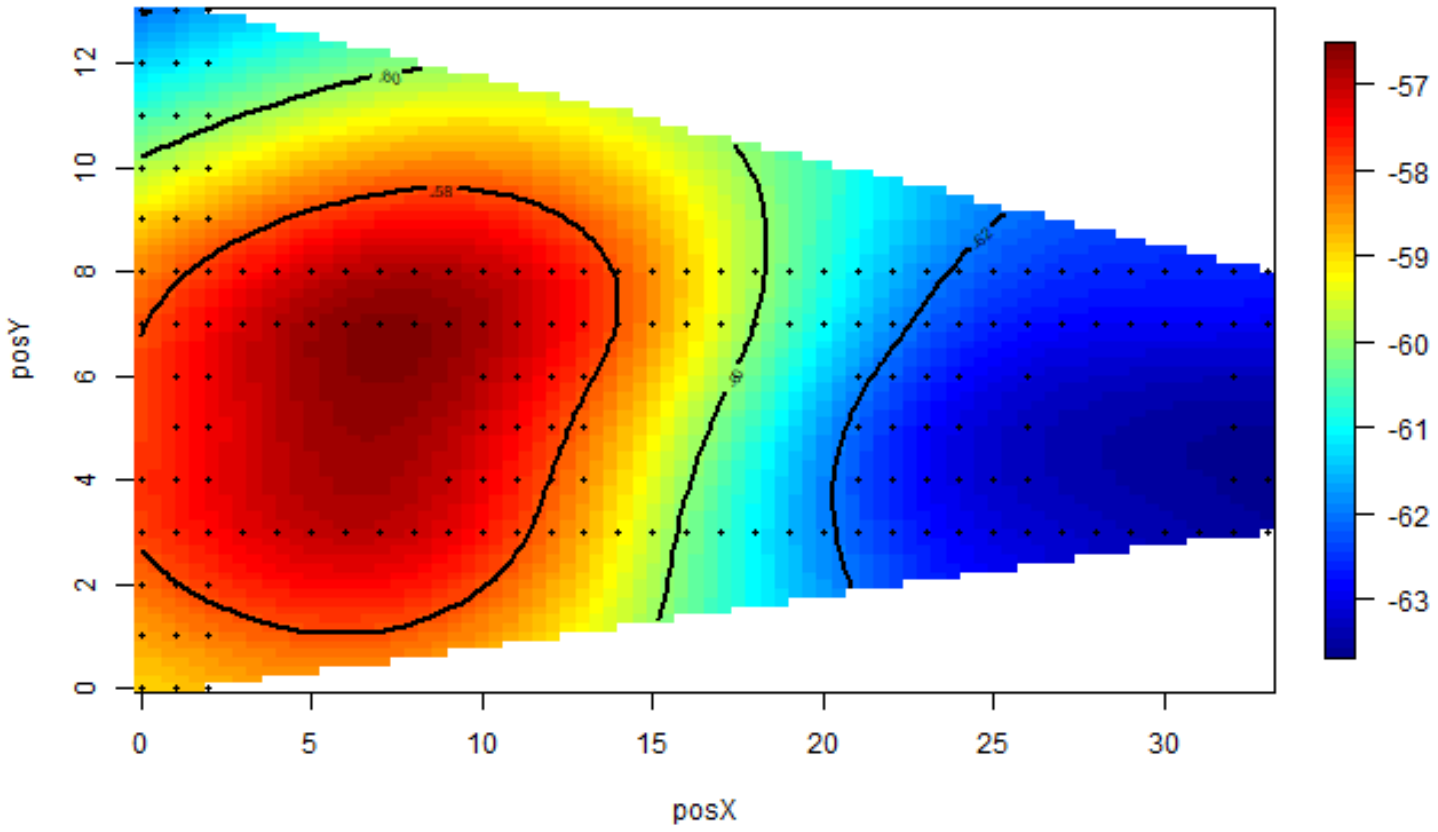



```
with(offlineSummary,
     smoothScatter((avgSignal - medSignal) ~ num,
                   xlab = "Number of Observations",
                   ylab = "mean - median"))
abline(h = 0, col = "#984ea3", lwd = 2)

lo.obj <- with(offlineSummary,
               loess(diff ~ num,
                     data = data.frame(diff = (avgSignal - medSignal),
                                       num = num)))

lo.obj.pr <- predict(lo.obj, newdata = data.frame(num = (70:120)))
lines(x = 70:120, y = lo.obj.pr, col = "#4daf4a", lwd = 2)
```

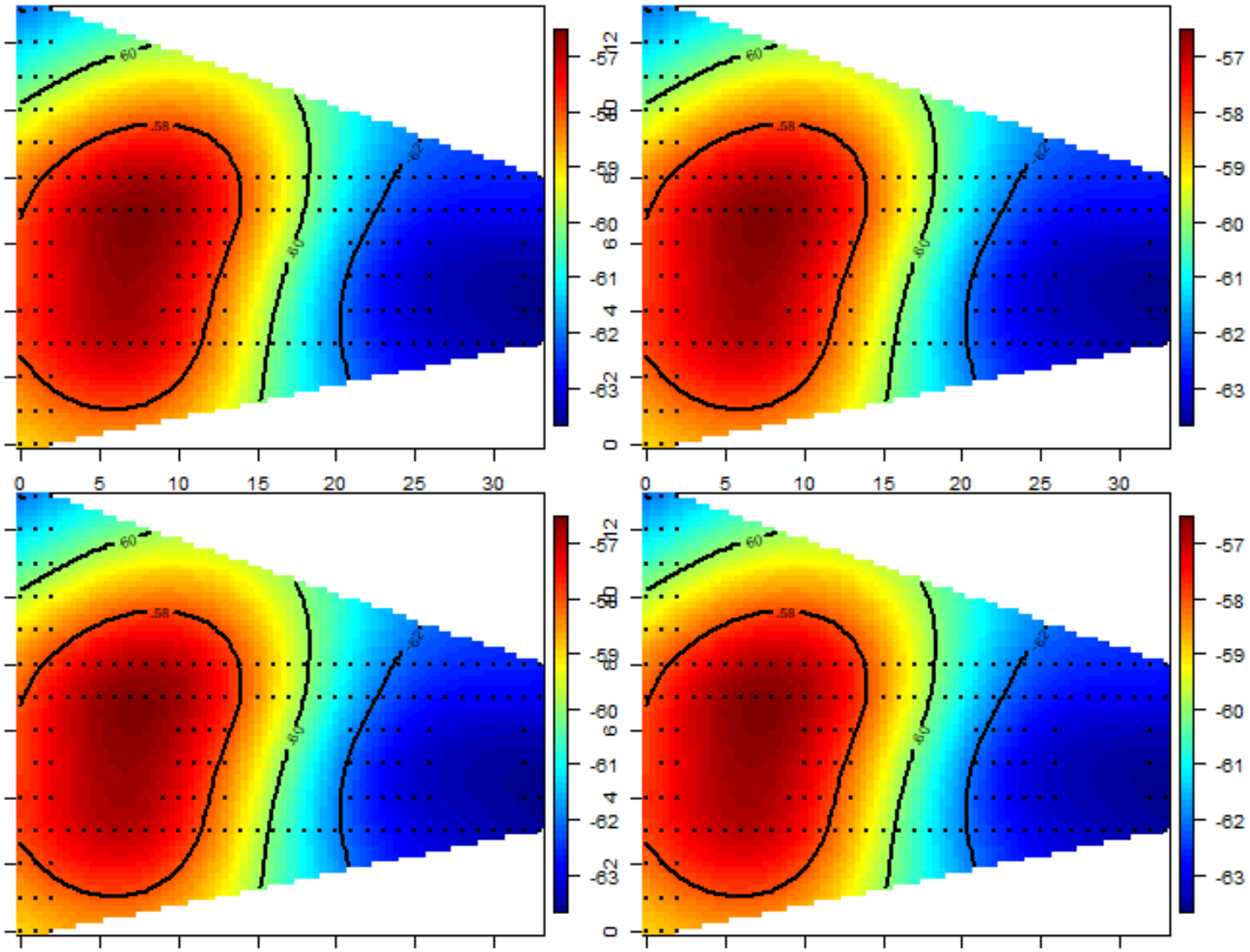


Signal and Distance

```
subMacs <- names(sort(table(offline$mac), decreasing = T))[1:7]

surfaceSS <- function(data, mac, angle) {
   oneAPAngle <- subset(offlineSummary, mac == mac & angle == angle)

   smothSS <- Tps(oneAPAngle[, c("posX", "posY")],
                  oneAPAngle$avgSignal)

   vizSmooth <- predictSurface(smothSS)

   plot.surface(vizSmooth, type = "C")
   points(oneAPAngle$posX, oneAPAngle$posY, pch=19, cex = 0.5)
}

surfaceSS(offlineSummary, subMacs[5], 0)
```




```
parCur <- par(mfrow = c(2,2), mar = rep(1, 4))
mapply(surfaceSS, mac = subMacs[ rep(c(5, 1), each = 2)],
       data = list(data = offlineSummary))
```



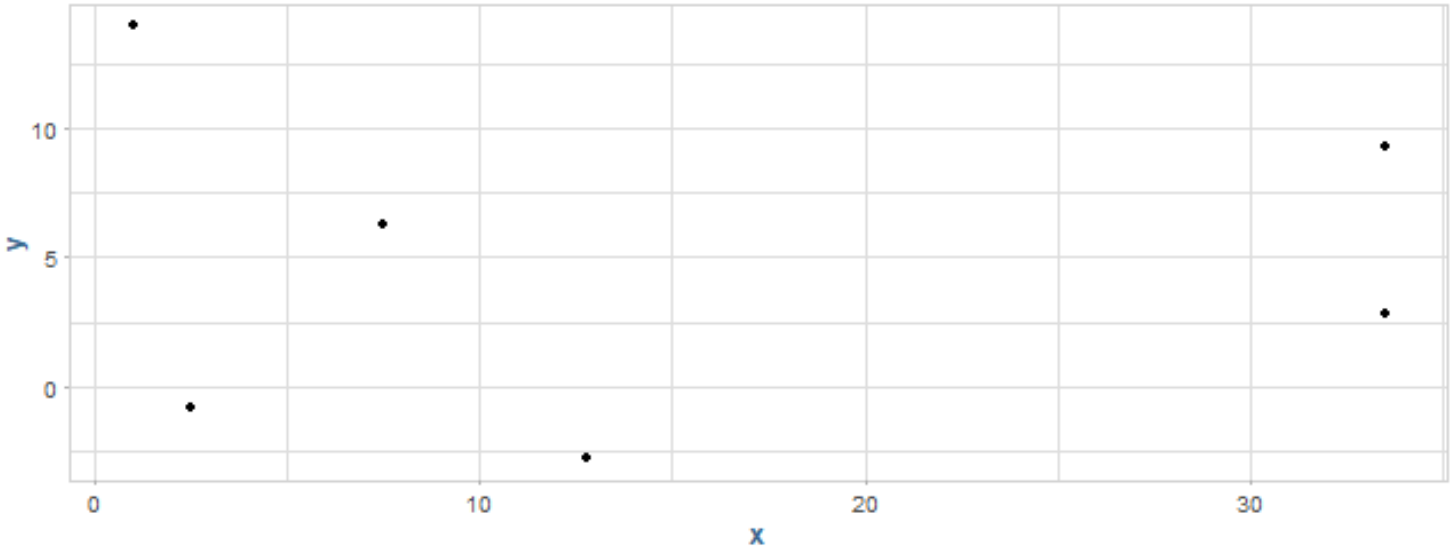

```
$`00:14:bf:b1:97:90`
NULL

$`00:14:bf:b1:97:90`
NULL

$`00:0f:a3:39:e1:c0`
NULL

$`00:0f:a3:39:e1:c0`
NULL
```

```
par(parCur)
```

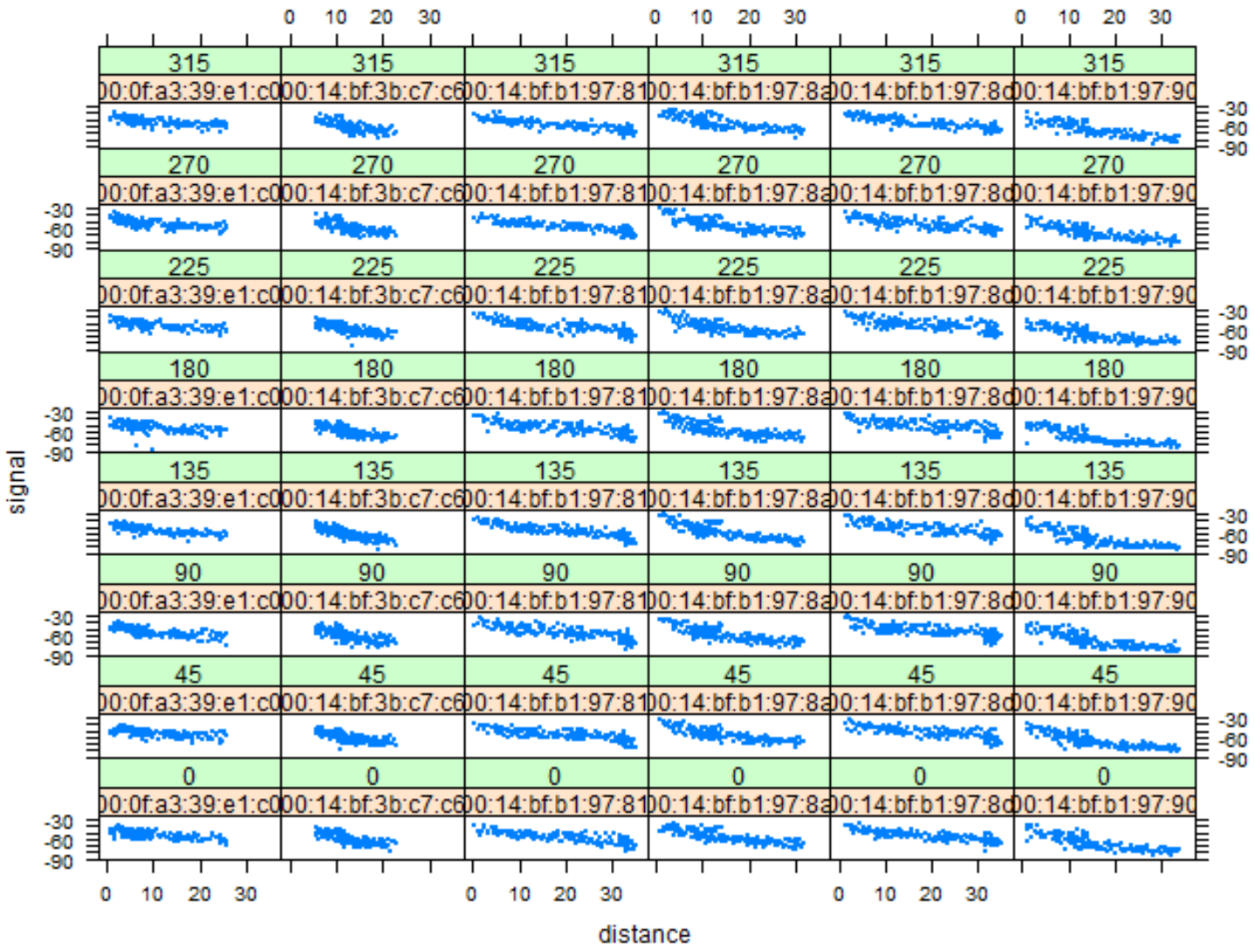
Exclude one of two similar access points

```
offlineSummary <- subset(offlineSummary, mac != subMacs[2])
```

```
AP <- matrix( c( 7.5, 6.3, 2.5, -.8, 12.8, -2.8,
                 1, 14, 33.5, 9.3, 33.5, 2.8),
              ncol = 2, byrow = T,
              dimnames = list(subMacs[ -2 ], c("x", "y") ))

ggplot(data.table(mac = rownames(AP), AP), aes(x, y)) +
   geom_point()
```




```
diffs <- offlineSummary[, c("posX", "posY")] - AP[offlineSummary$mac, ]

offlineSummary$dist <- sqrt(diffs[, 1]^2 + diffs[, 2]^2)

xyplot(signal ~ dist | factor(mac) + factor(angle),
       data = offlineSummary, pch = 19, cex = 0.3,
       xlab = "distance")
```



Nearest Neighbor Methods to Predict Location

```
macs <- unique(offlineSummary$mac)

online <- readData(file_online, submacs = macs)

length(unique(online$posXY))
```

```
[1] 60
```

```
tabonlineXYA = table(online$posXY, online$angle)
tabonlineXYA[1:6, ]
```

```
              0  45  90 135 180 225 270 315
  0-0.05      0   0   0 593   0   0   0   0
  0.15-9.42   0   0 606   0   0   0   0   0
  0.31-11.09  0   0   0   0   0 573   0   0
  0.47-8.2  590   0   0   0   0   0   0   0
  0.78-10.94 586  0   0   0   0   0   0   0
  0.93-11.69  0   0   0   0 583   0   0   0
```

```
keepVars <- c("posXY", "posX", "posY", "orientation", "angle")

byLoc <- with(online,
              by(online, list(posXY),
                 function(x) {
                    ans <- x[1, ..keepVars]
                    avgSS <- tapply(x$signal, x$mac, mean)
                    y = matrix(avgSS, nrow = 1, ncol = 6,
                               dimnames = list(ans$posXY, names(avgSS)))
                    cbind(ans, y)
                 }))

onlineSummary <- do.call("rbind", byLoc)

dim(onlineSummary)
```

```
[1] 60 11
```

```
names(onlineSummary)
```

```
 [1] "posXY"             "posX"              "posY"
 [4] "orientation"       "angle"             "00:0f:a3:39:e1:c0"
 [7] "00:14:bf:3b:c7:c6" "00:14:bf:b1:97:81" "00:14:bf:b1:97:8a"
[10] "00:14:bf:b1:97:8d" "00:14:bf:b1:97:90"
```

Choice of Orientation

```
reshapeSS <- function(data, varSignal = "signal",
                      keepVars = c("posXY", "posX", "posY")) {

   byLocation <- with(data,
                  by(data, list(posXY),
                     function(x) {
                        ans <- x[1, ..keepVars]
                        avgSS <- tapply(x$signal, x$mac, mean)
                        y = matrix(avgSS, nrow = 1, ncol = 6,
                                   dimnames = list(ans$posXY, names(avgSS)))
                        cbind(ans, y)
```

```
                    }))

   newDataSS <- do.call("rbind", byLocation)

   col_names <- colnames(newDataSS)
   n_cols <- length(col_names)

   colnames(newDataSS)[4:n_cols] <- sapply(col_names[4:n_cols], function(col) {
      n <- nchar(col)
      substr(col, n - 2, n)
   })

   newDataSS[, 4:ncol(newDataSS)] <- round(newDataSS[, 4:ncol(newDataSS)])

   return(newDataSS)
}

selectTrain <- function(angleNewObs, signals, m) {

   refs <- seq(0, by = 45, length = 8)

   nearestAngle <- roundOrientation(angleNewObs)

   if( m %% 2 == 1) {
      angles <- seq(-45 * (m - 1) / 2, 45 * (m - 1) / 2, length = m)
   } else {
      m = m + 1
      angles <- seq(-45 * (m - 1) / 2, 45 * (m - 1) / 2, length = m)

      if( sign(angleNewObs - nearestAngle) >= 1)
         angles = angles[ -1 ]
      else
         angles = angles[ -m ]
   }

   angles <- angles + nearestAngle
   angles[angles < 0] = angles[ angles < 0] + 360
   angles[angles > 360] = angles[ angles > 360 ] - 360

   signals <- signals[angle %in% angles, ]

   reshapeSS(signals, varSignal = "avgSignal")
}
```

```
train130 <- selectTrain(130, offlineSummary, m = 3)

dim(train130)
```

```
[1] 166   9
```

```
findNN <- function(newSignal, trainSubset) {
   diffs <- apply(trainSubset[ , 4:9], 1,
                  function(x) x - newSignal)
   dists <- apply(diffs, 2, function(x) sqrt(sum(x^2)) )
   closest <- order(dists)
   return(trainSubset[closest, 1:3 ])
}

predXY <- function(newSignals, newAngles, trainData,
                   numAngles = 1, k = 3) {

   closeXY <- list(length = nrow(newSignals))

   for(i in 1:nrow(newSignals)) {
      trainSS <- selectTrain(newAngles[i], trainData, m = numAngles)
      closeXY[[i]] =
         findNN(newSignal = as.numeric(newSignals[i, ]), trainSS)
   }

   estXY = lapply(closeXY,
                  function(x) sapply(x[, 2:3],
                                     function(x) mean(x[1:k])))

   estXY <- do.call("rbind", estXY)

   return(estXY)
}

estXYk1 <- predXY(newSignals = onlineSummary[, 6:11],
                  newAngles = onlineSummary[, 4],
                  offlineSummary, numAngles = 3, k = 1)

estXYk3 <- predXY(newSignals = onlineSummary[, 6:11],
                  newAngles = onlineSummary[, 4],
                  offlineSummary, numAngles = 3, k = 3)
```

```
calcError <- function(estXY, actualXY)
   sum( rowSums( (estXY - actualXY) ^ 2) )
```

```
actualXY <- onlineSummary[, c("posX", "posY")]
sapply(list(estXYk1, estXYk3), calcError, actualXY)
```

```
[1] 420.7603 388.1070
```