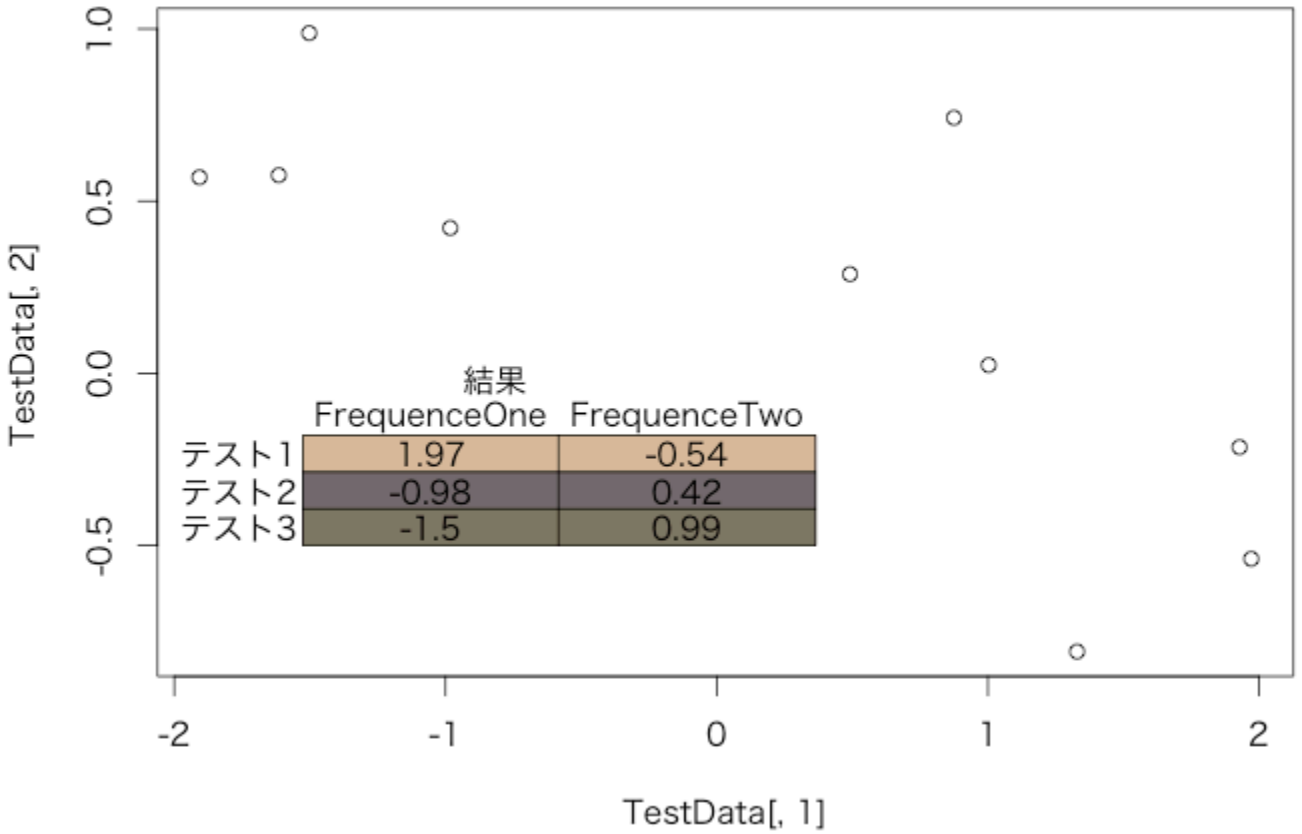
```
#パッケージの読み込み
library("plotrix")
par(family = "HiraKakuProN-W3")
#データ例を作成
TestData <- data.frame(row.names = paste("テスト", 1:10, sep = ""),
                       FrequenceOne = runif(10, min = -3, max = 3),
                       FrequenceTwo = runif(10, min = -1, max = 1),
                       ColData = c("#d9bb9c", "#756c6d", "#807765",
                                   "#ad8a80", "#685432", "#464031",
                                   "#28231e", "#563c22", "#deb7a0", "#505457"))
```

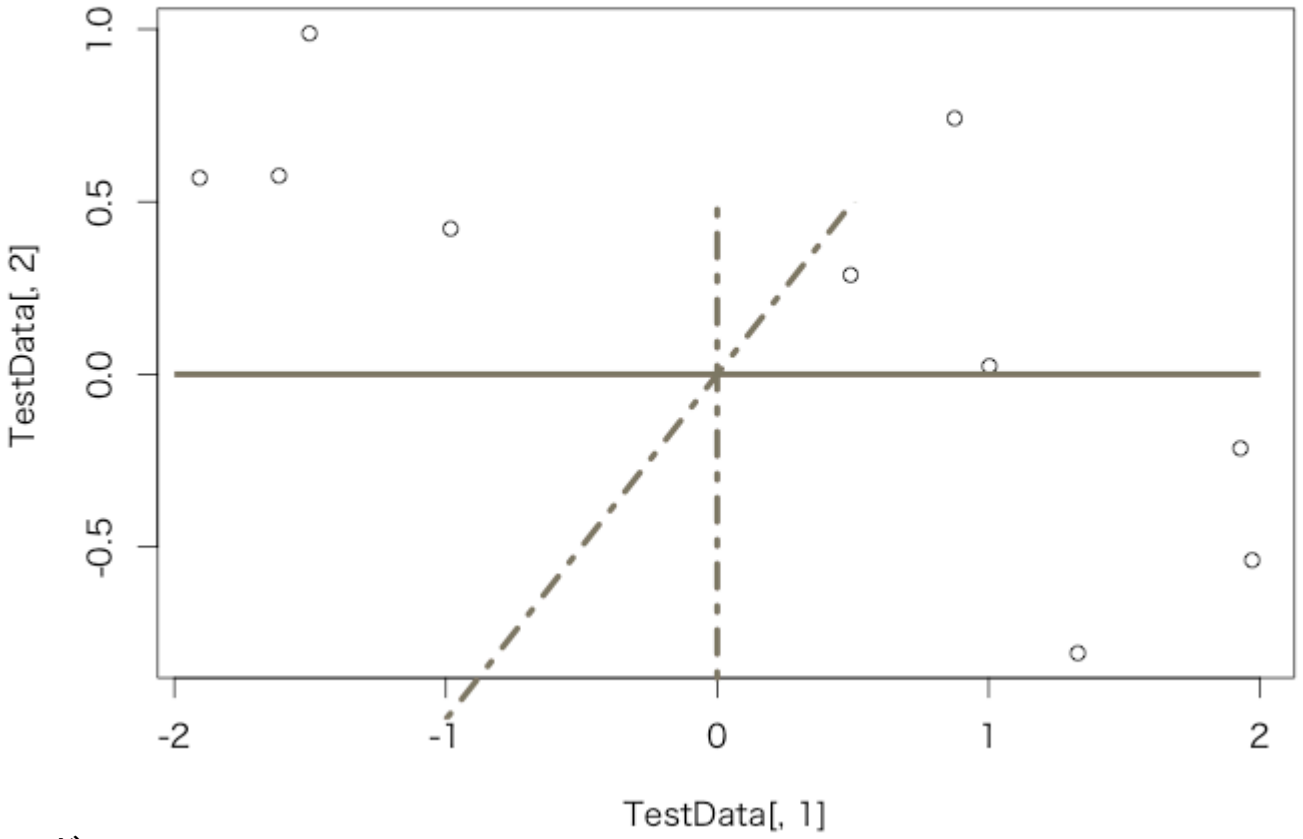
addtable2plotコマンド



コマンド:

```
plot(TestData[, 1], TestData[, 2])
addtable2plot(-2, -0.5, table = round(TestData[1:3, 1:2], 2), bty = "o",
              display.rownames = TRUE, hlines = TRUE,
              vlines = TRUE, title = "結果", bg = as.character(TestData[1:3, 3]))
```

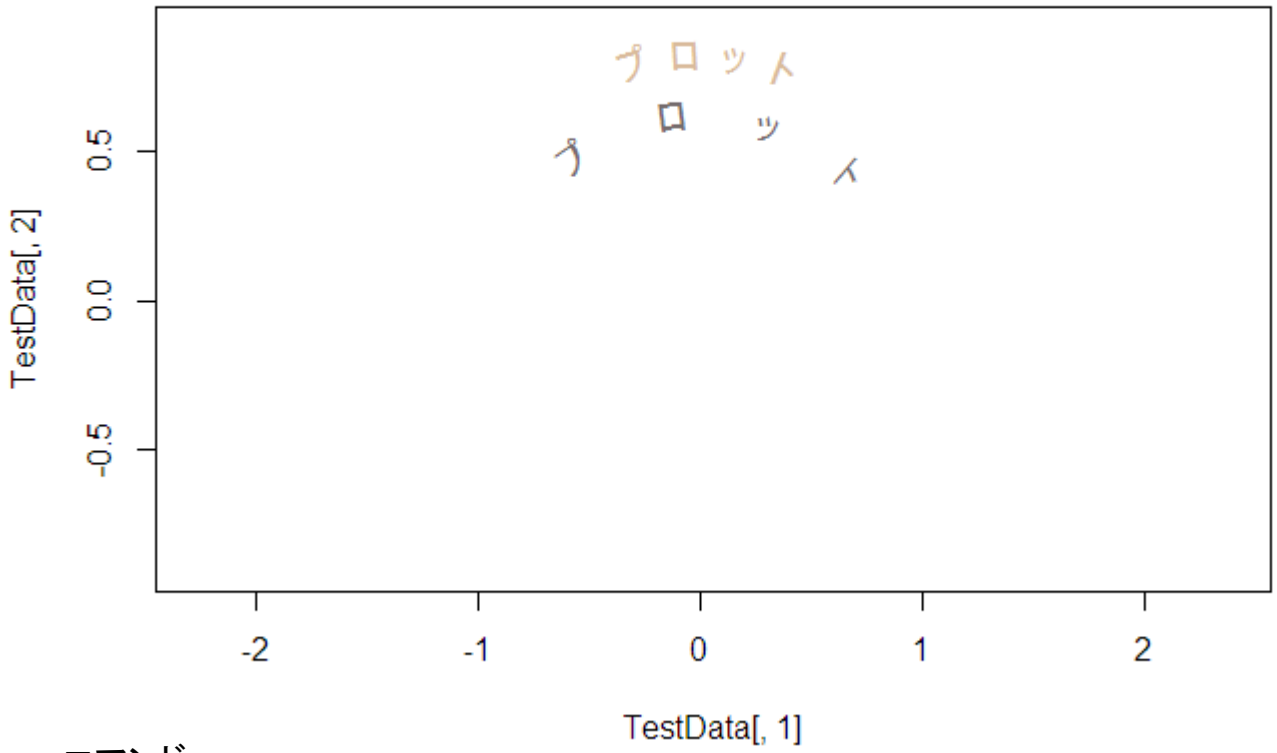
ablineclipコマンド



コマンド:

```
plot(TestData[, 1], TestData[, 2])
#X軸方向に線を追加
ablineclip(h = 0, x1 = −2, x2 = 2, lty = 1, lwd = 4, col = as.character(TestData[3, 3]))
#Y軸方向に線を追加
ablineclip(v = 0, y1 = −1.0, y2 = 0.5, lty = 4, lwd = 4, col = as.character(TestData[3, 3]))
#傾き線を追加（a = intercept, b = slop オプション）
ablineclip(a = 0, b = 1, y1 = −1.0, y2 = 0.5, lty = 4, lwd = 4, col = as.character(TestData[3, 3]))
```

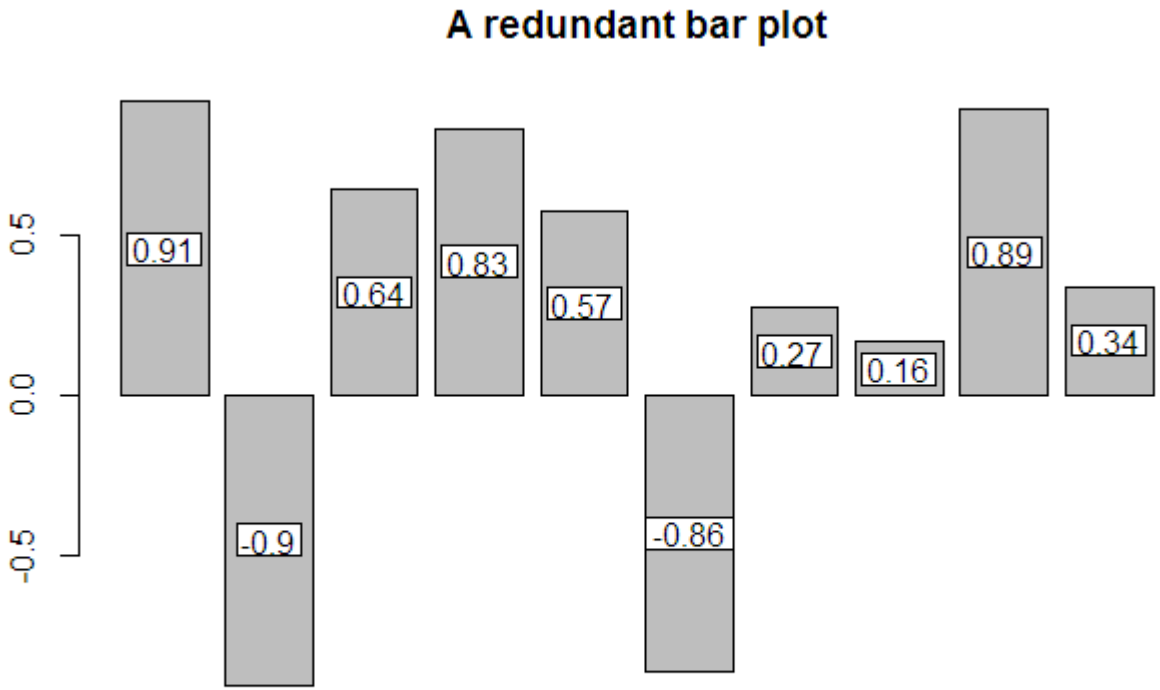
arctextコマンド



コマンド:

```
plot(TestData[, 1], TestData[, 2], type = "n")
#普通にプロット
arctext("プロット", center = c(0, 0), col = as.character(TestData[1, 3]))
#プロット範囲を広げることもできます
arctext("プロット", center = c(0, -0.2), stretch = 2, col = as.character(TestData[2, 3]))
```

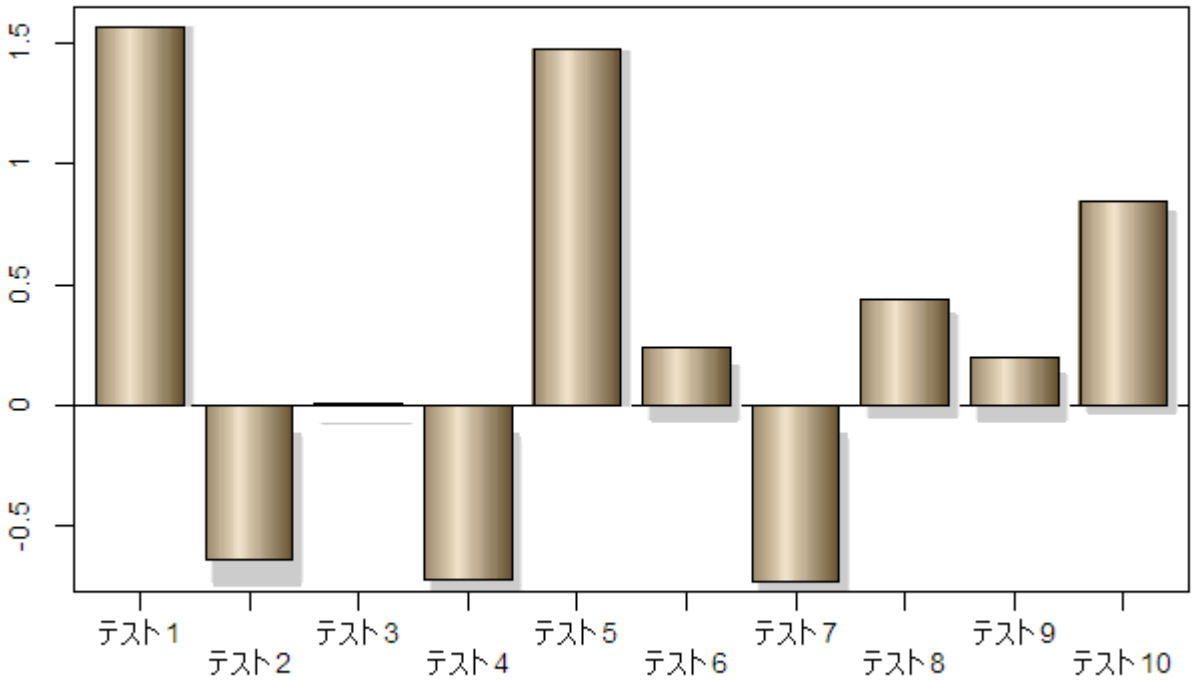
barlabelsコマンド



コマンド:

```
barpos <- barplot(TestData[, 2], main = "A redundant bar plot")
#minyでラベルを表示する最小値を設定します
barlabels(barpos, round(TestData[, 2], 2), miny = -1)
```

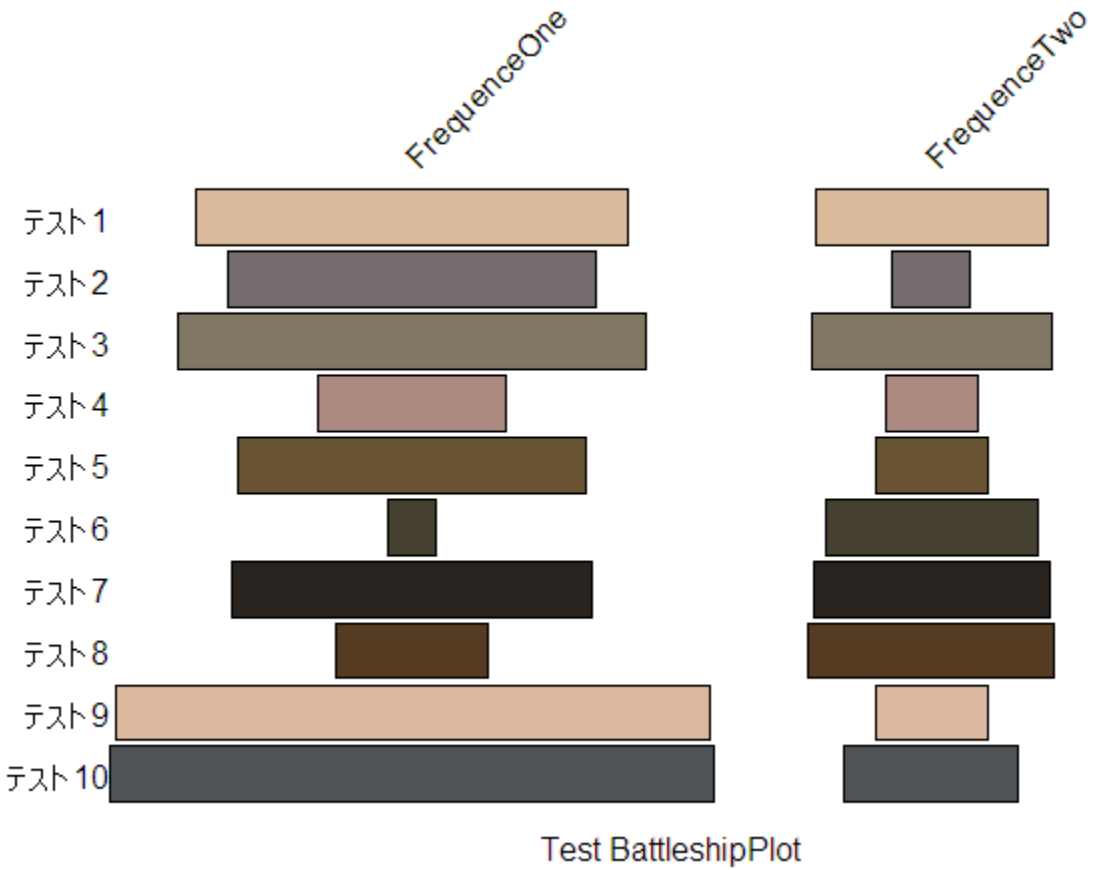
barpコマンド



コマンド:

```
barp(apply(TestData[, 1:2], 1, mean), names.arg = row.names(TestData[, 1:2]),
        col = as.character(TestData[5, 3]), staxx = TRUE, staxy = FALSE,
        cex.axis = 0.9, cylindrical = TRUE, shadow=TRUE)
```

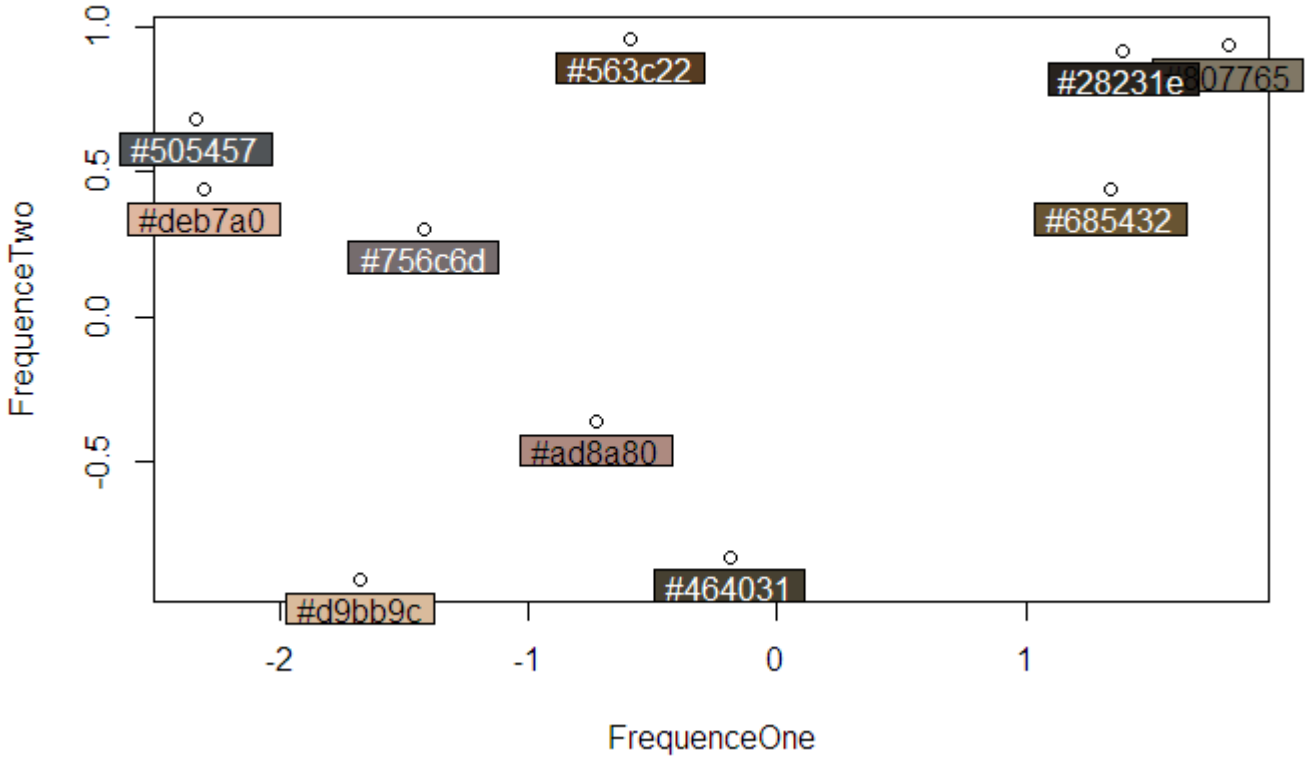
battleship.plotコマンド



コマンド:

```
battleship.plot(TestData[, 1:2], xlab = "Test BattleshipPlot",
                col = as.character(TestData[, 3]),
                xaxlab = colnames(TestData[, 1:2]), yaxlab = row.names(TestData))
```

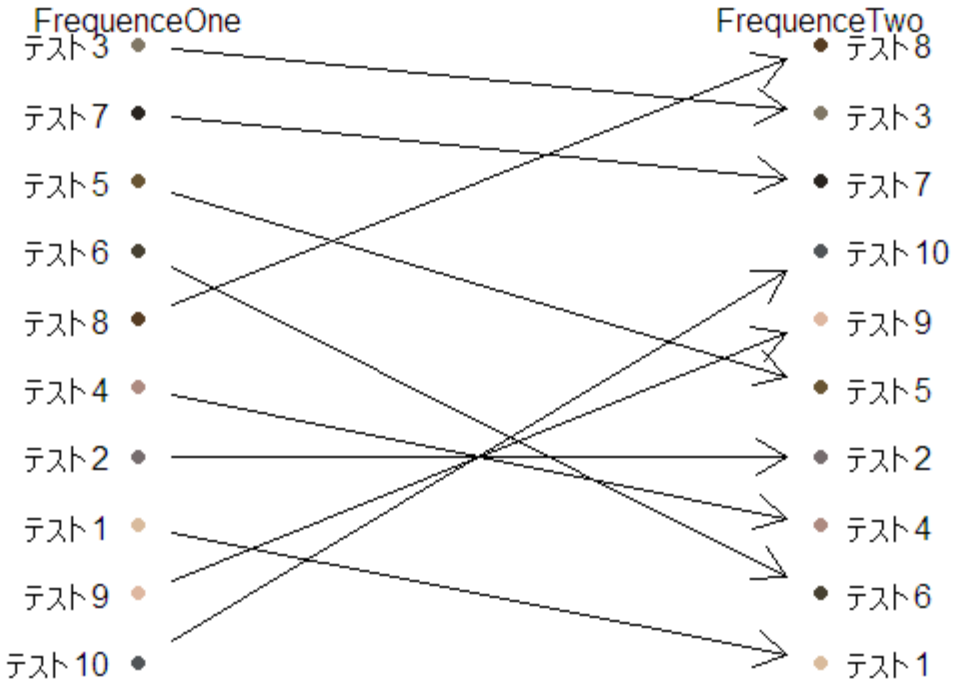
boxed.labelsコマンド



コマンド:

```
plot(TestData[, 1:2], type = "p")
boxed.labels(TestData[, 1], TestData[, 2] - 0.1, bg = as.character(TestData[, 3]),
             labels = TestData[, 3])
```

bumpchartコマンド



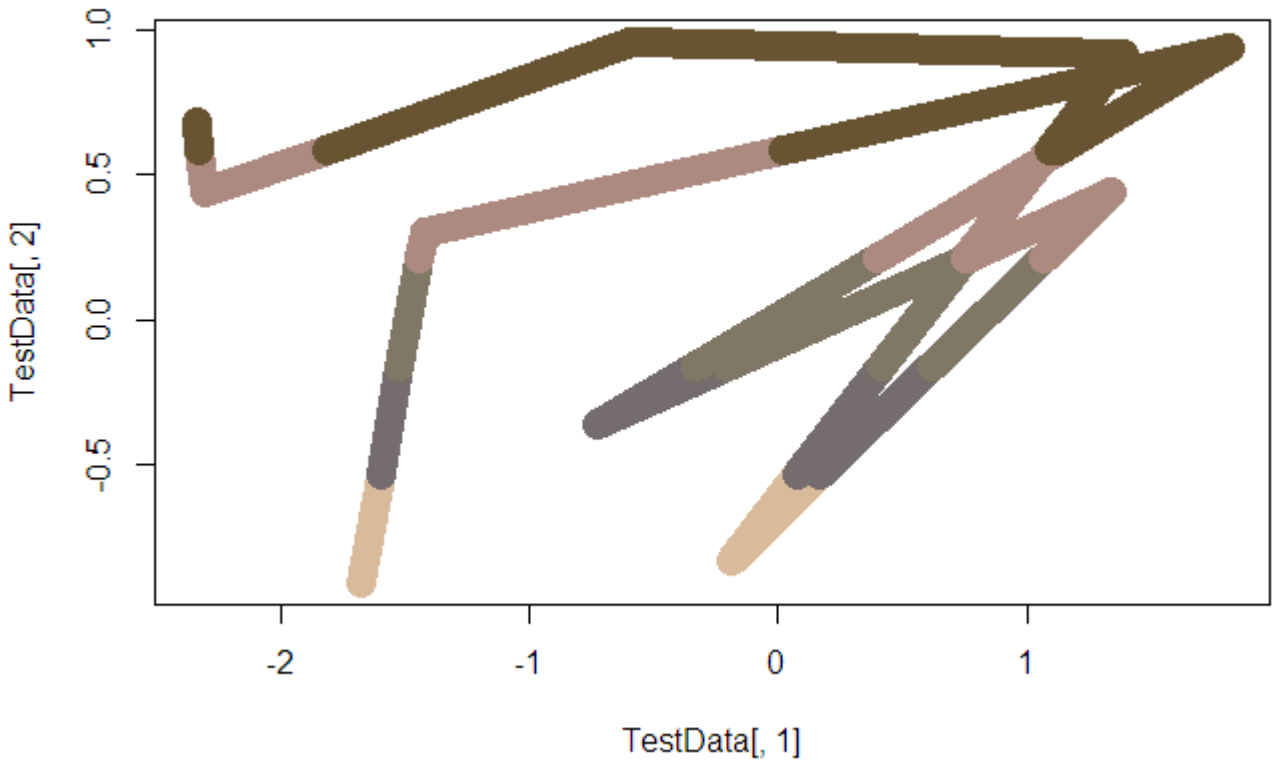
コマンド：bumpchart(TestData[, 1:2], arrows = TRUE, col = as.character(TestData[, 3]))

clock24.plotコマンド

コマンド:

```
clock24.plot(TestData[, 1], TestData[, 2], main = " TEST clock24.plot",
             show.grid = FALSE, line.col = as.character(TestData[, 3]), lwd = 3)
```

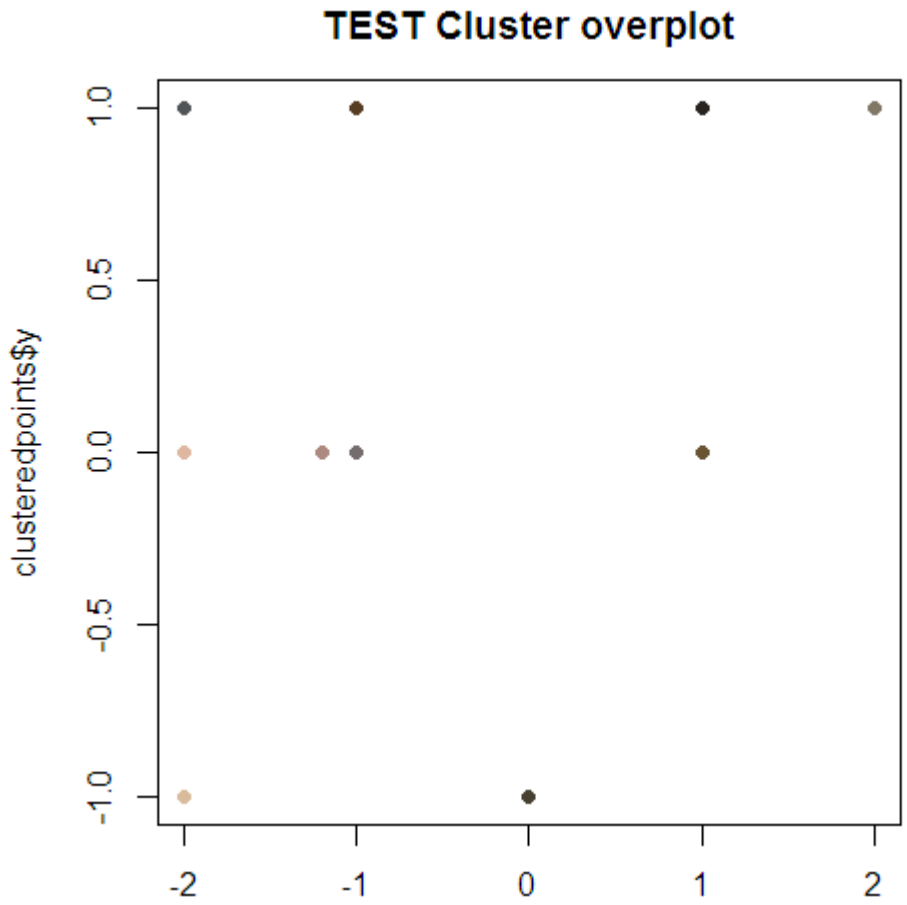
clplotコマンド



コマンド:

```
clplot(TestData[, 1], TestData[, 2], cols = as.character(TestData[, 3]), lwd = 15)
```

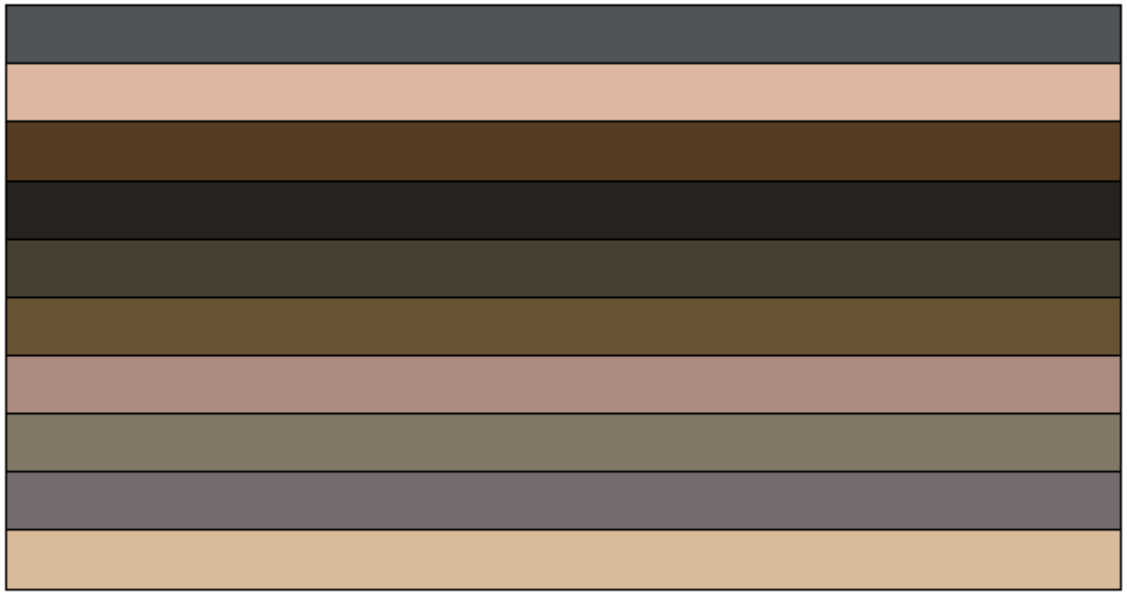
clusteredpointsコマンド



コマンド:

```
clusteredpoints <- cluster.overplot(round(TestData[, 1:2], 0),
                                    col = as.character(TestData[, 3]),
                                    away = rep(0.2,2))
plot(clusteredpoints, col = clusteredpoints$col, pch = 19, main = "TEST Cluster overplot")
```

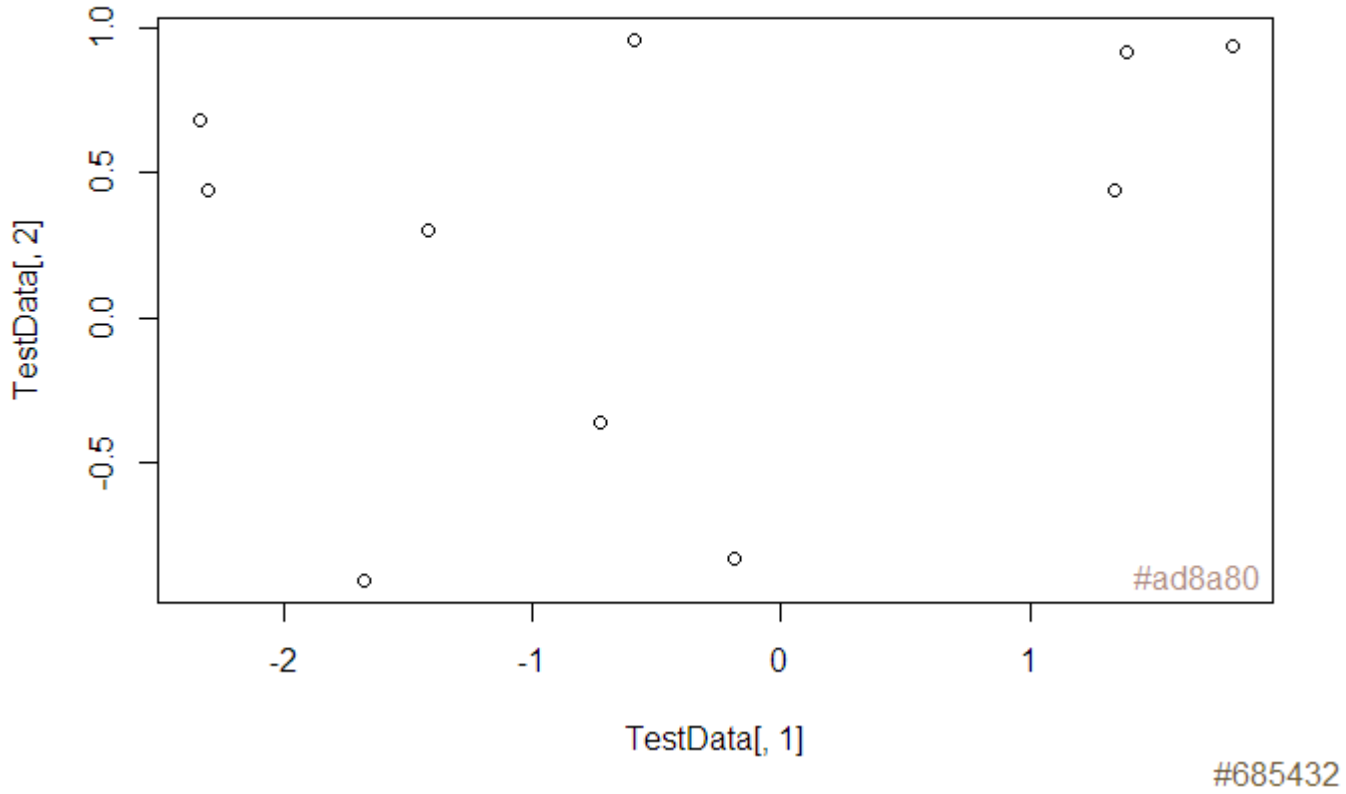
color2D.matplotコマンド



コマンド:

```
color2D.matplot(matrix(1:10, nrow = 10), axes = FALSE, yrev = FALSE,
                cellcolors = as.character(TestData[, 3]), xlab = "", ylab = "")
```

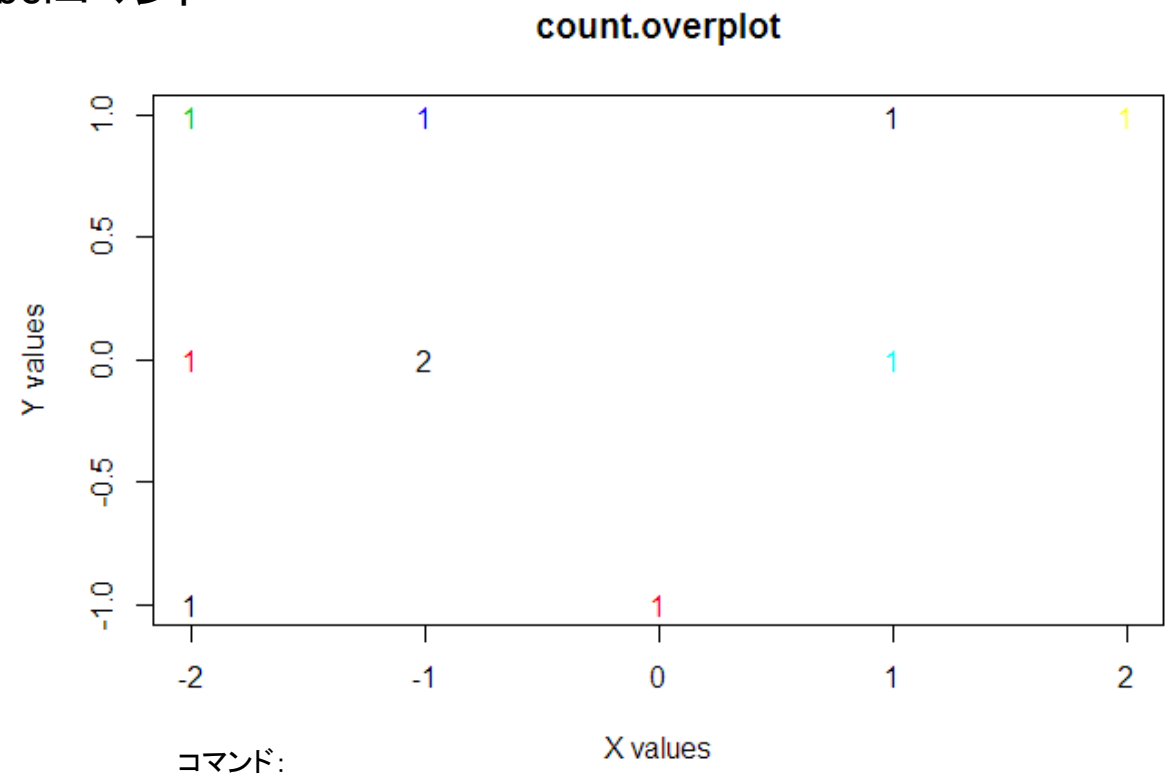
corner.labelコマンド



コマンド:

```
plot(TestData[, 1], TestData[, 2])
corner.label(TestData[4, 3], y = -1, x = 1, col = as.character(TestData[4, 3]))
corner.label(TestData[5, 3], y = -1, x = 1, figcorner = TRUE, col = as.character(TestData[5, 3]))
```

corner.labelコマンド



コマンド：

```
Test <- round(TestData[, 1:2], 0)
Test
```

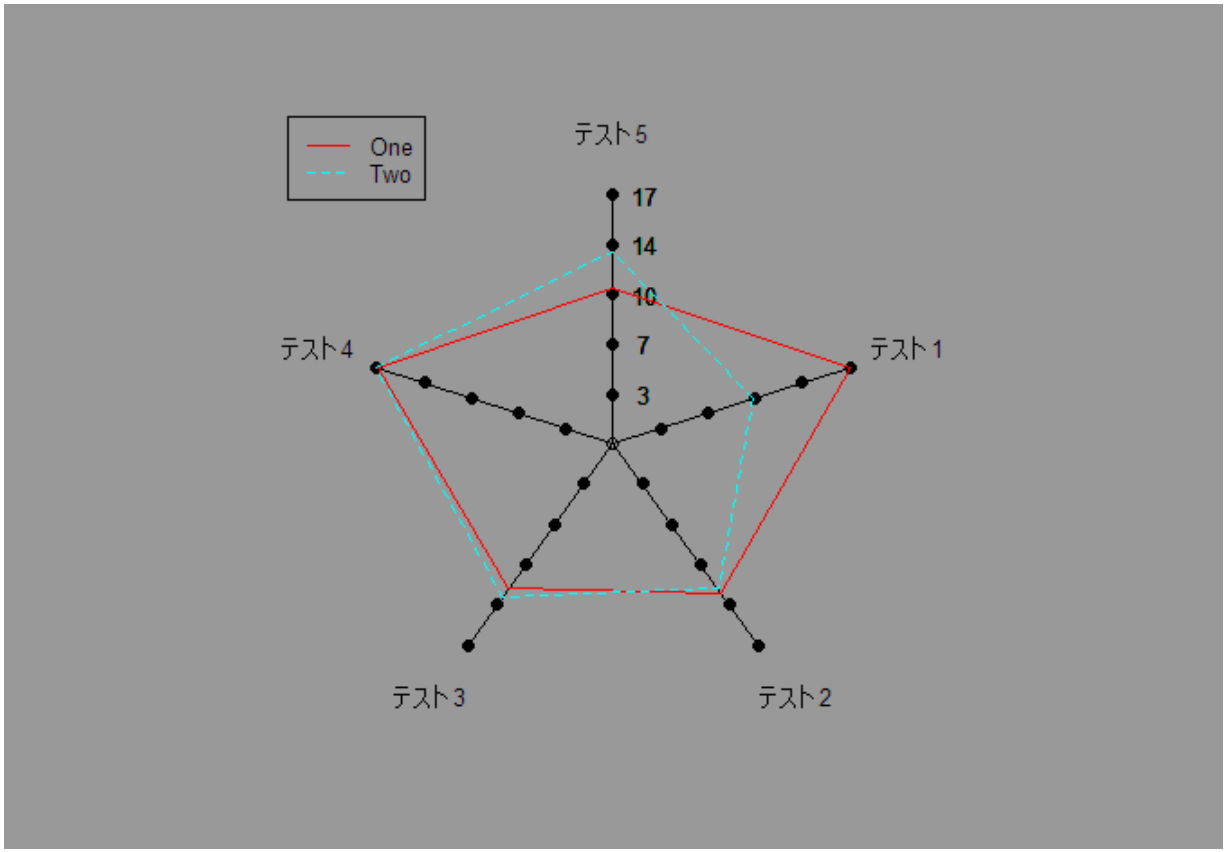
	FrequenceOne	FrequenceTwo
テスト1	−2	−1
テスト2	−1	0
テスト3	2	1
テスト4	−1	0
テスト5	1	0
テスト6	0	−1
テスト7	1	1
テスト8	−1	1
テスト9	−2	0
テスト10	−2	1

```
count.overplot(Test, main = "count.overplot", col = TestData[, 3],
                    xlab = "X values", ylab = "Y values")
```

dendroPlotコマンド

コマンド:
dendroPlot(TestData[, 1:2], xlab = "Groups", ylab = "Value of x", main = "Test dendroPlot")

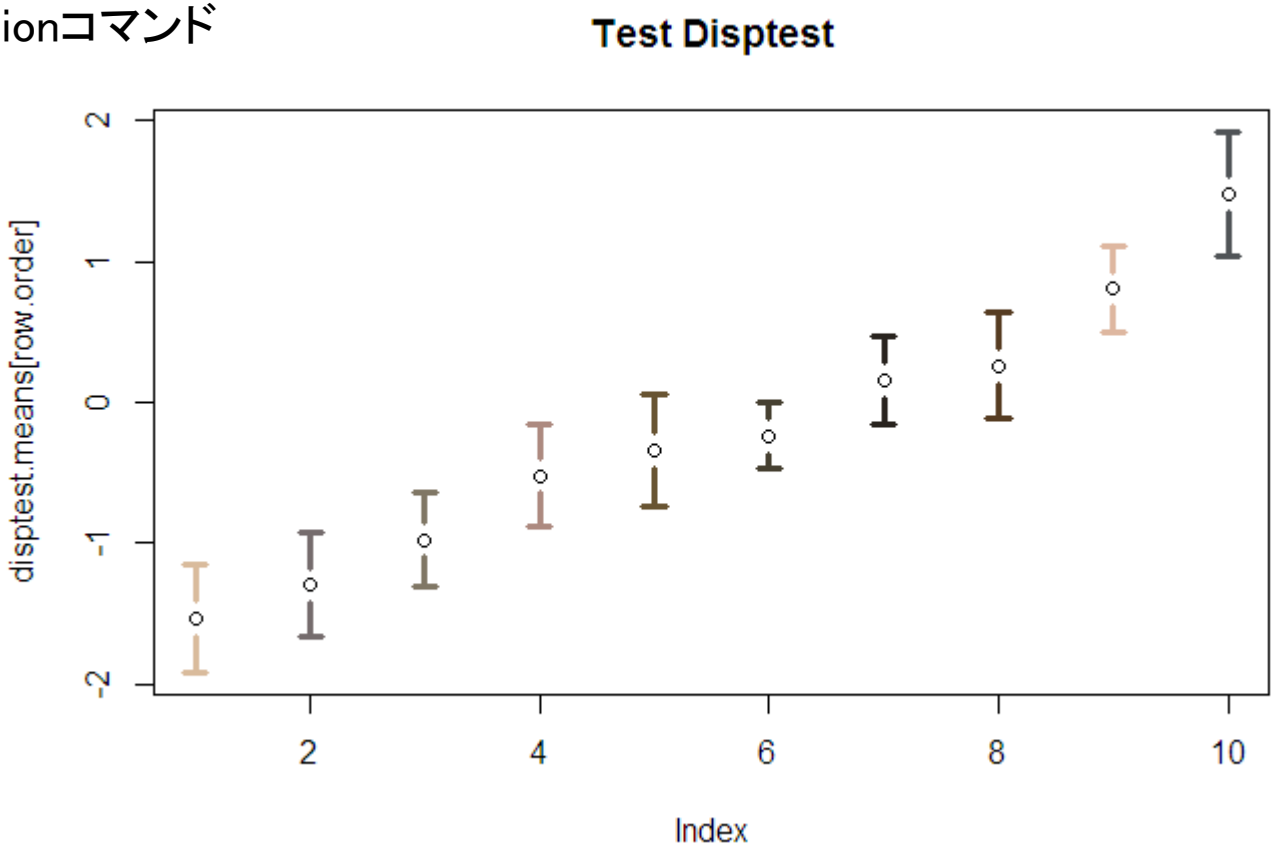
diamondplotコマンド



コマンド:

```
TestData <- data.frame(row.names = paste("テスト", 1:5, sep = ""),
                       One = runif(5, min = 1, max = 2),
                       Two = runif(5, min = 1, max = 2),
                       ColData = c("#d9bb9c", "#756c6d", "#807765",
                                   "#ad8a80", "#685432"))
diamondplot(TestData[, 1:2])
```

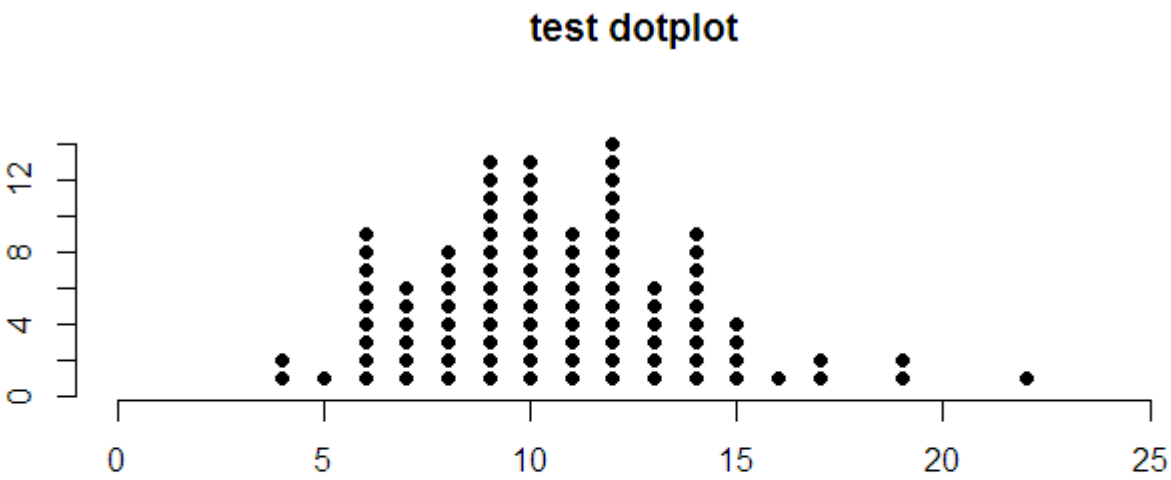
dispersionコマンド



コマンド:

```
disptest.means <- rowMeans(TestData[, 1:2])
row.order <- order(disptest.means)
se.disptest <- unlist(apply(disptest, 1, std.error))
plot(disptest.means[row.order], main="Test Disptest",
     ylim = c(min(disptest.means-se.disptest), max(disptest.means
+se.disptest)))
dispersion(1:10, disptest.means[row.order], se.disptest[row.order],
                lwd = 3, col = as.character(TestData[, 3]))
```

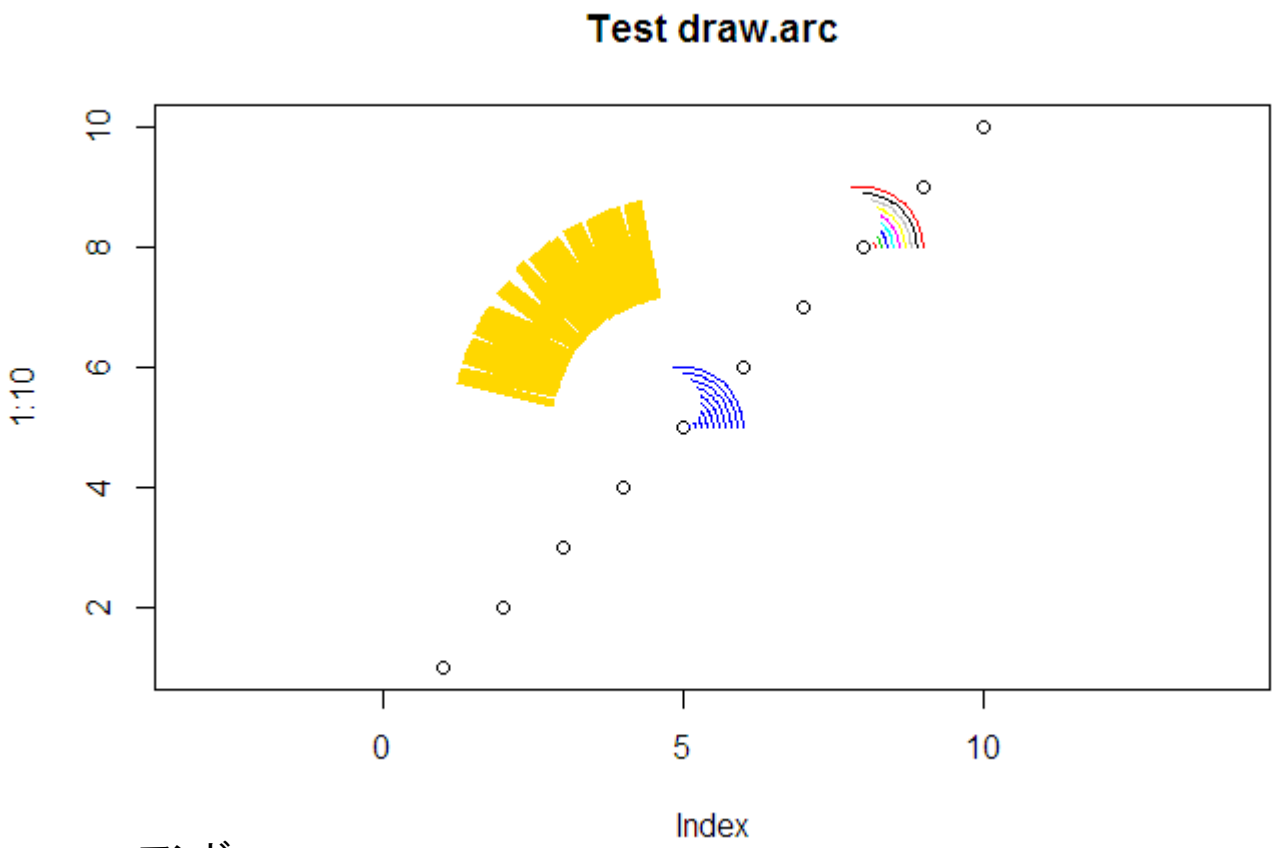
dotplot.mtbコマンド



コマンド:

```
set.seed(42)
x <- rpois(100,10)
dotplot.mtb(x, yaxis = TRUE, main = "test dotplot")
```

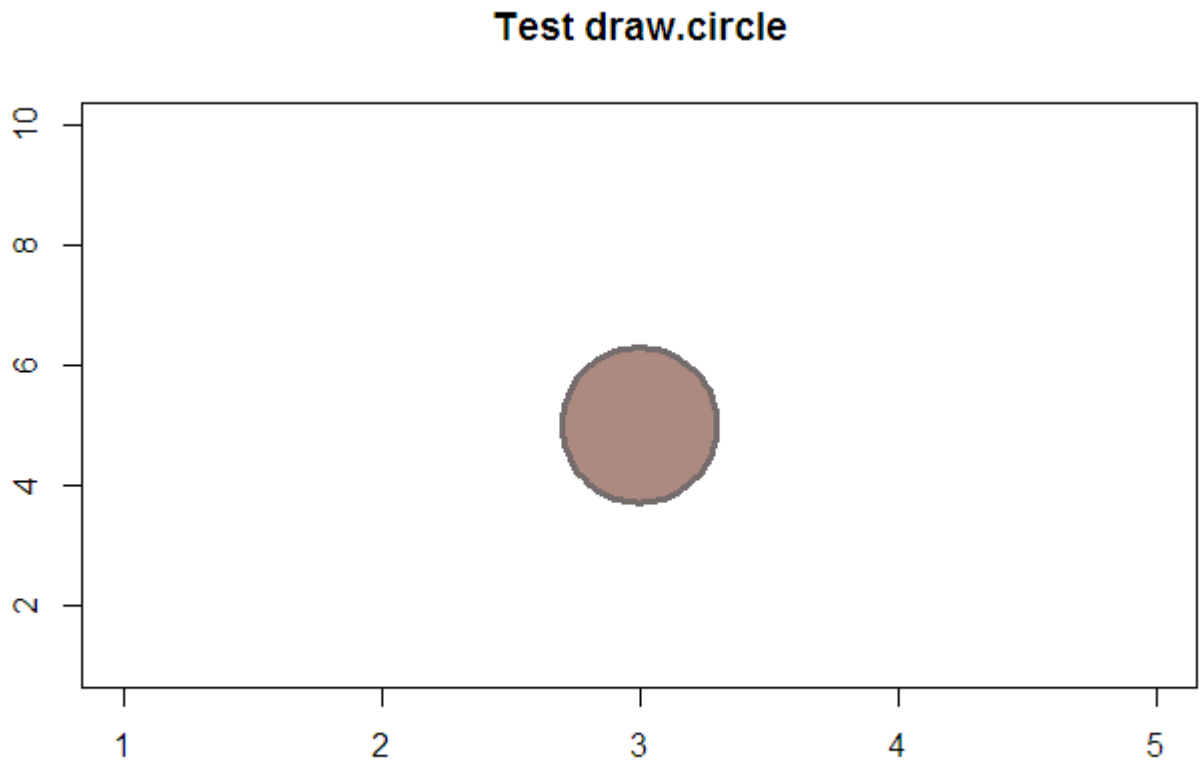
draw.arcコマンド



コマンド:

```
plot(1:10, asp = 1,  main="Test draw.arc")
draw.arc(5, 5, 1:10/10, deg2 = 1:10*10, col = "blue")
draw.arc(8, 8, 1:10/10, deg2 = 1:10*10, col = 1:10)
draw.arc(5, 5, 3, deg1 = 100, deg2 = 170, col = "gold", lwd = 50, lend = 1)
```

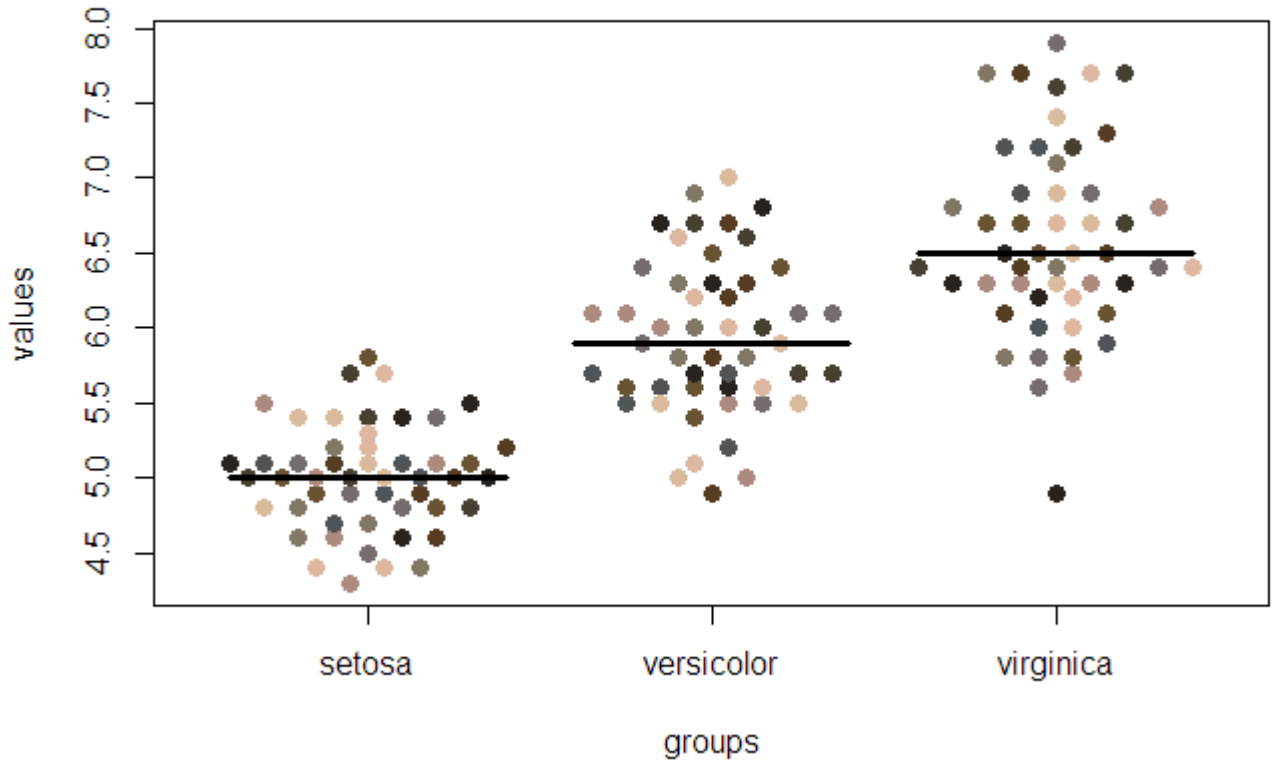
draw.circleコマンド



コマンド:

```
plot(1:5, seq(1, 10, length = 5), type = "n",
     xlab = "", ylab = "", main = "Test draw.circle")
#radiusで半径を指定
draw.circle(3, 5, radius = 0.3, border = as.character(TestData[2, 3]),
            col = as.character(TestData[4, 3]), lty = 1, lwd = 3)
```

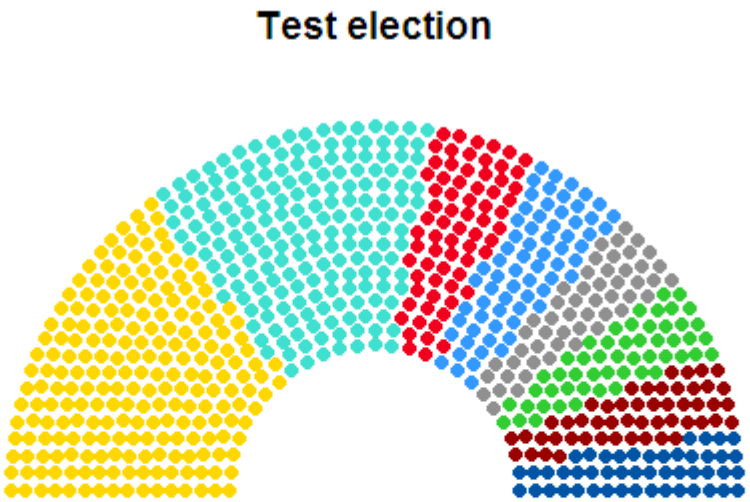
ehplotコマンド



コマンド:

```
data(iris)
ehplot(iris$Sepal.Length, iris$Species, intervals = 20,
        cex  =1.8, pch = 20, col = as.character(TestData[, 3]))
```

electionコマンド



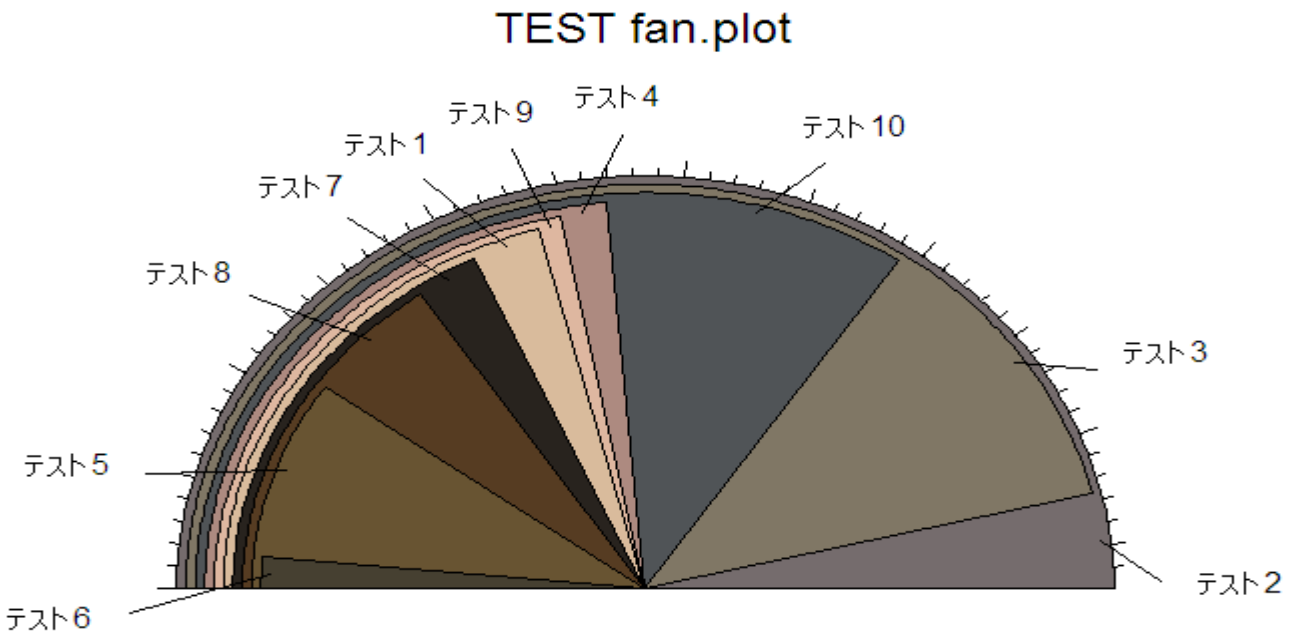
コマンド:

```
#データ例作成
eu = structure(list(colour = c("#3399FF", "#F0001C", "#0054A5", "#FFD700",
                               "#990000", "#909090", "#32CD32", "#40E0D0"),
                    party = c("EPP", "S and D", "ECR", "ALDE", "GUE-NGL",
                              "Non-Inscrits", "Greens-EFA", "EFDD"),
                    members = c(220L, 191L, 70L, 68L, 52L, 52L, 50L, 48L)),
                    .Names = c("colour", "party", "members"), row.names = c(NA, -8L),
                    class = "data.frame")
strasbourg = seats(751, 16)

#electionコマンド
eugov = election(strasbourg, eu, party~members, colours = eu$colour)
oldmar<-par(mar=c(2,4,4,2))

#図のプロット
plot(eugov$x, eugov$y, col = eugov$colour, asp = 1, pch = 19,
     ylim = c(-2, 2.5), xlab = "", ylab = "", main = "Test election", axes=FALSE)
```

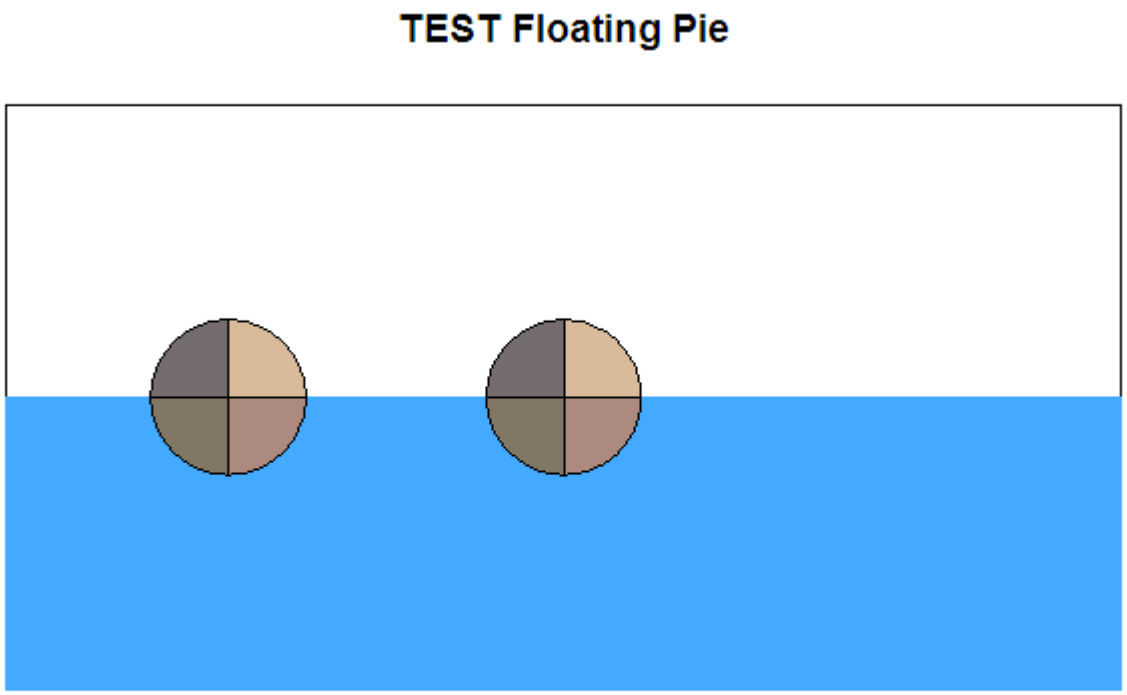
fan.plotコマンド



コマンド:

```
#データ例を作成
TestData <- data.frame(row.names = paste("テスト", 1:10, sep = ""),
                       FrequenceOne = runif(10, min = 0, max = 1),
                       FrequenceTwo = runif(10, min = -1, max = 1),
                       ColData = c("#d9bb9c", "#756c6d", "#807765",
                                   "#ad8a80", "#685432", "#464031",
                                   "#28231e", "#563c22", "#deb7a0", "#505457"))
#プロット
fan.plot(TestData[, 1], max.span = pi,
         labels = row.names(TestData),
         main = "TEST fan.plot", ticks = 276,
         col = as.character(TestData[, 3]))
```

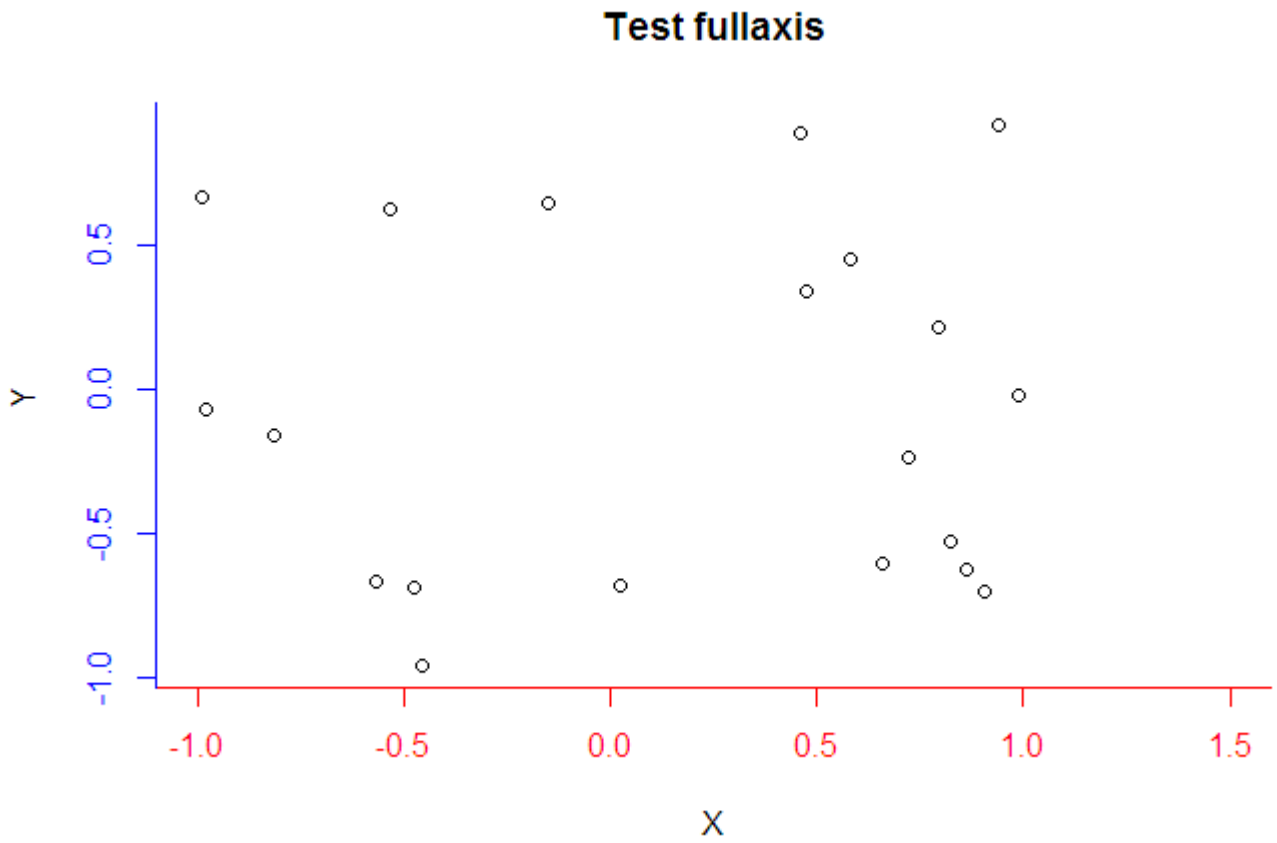
floating.pieコマンド



コマンド:

```
plot(1:5, type = "n", main = "TEST Floating Pie", xlab = "", ylab = "", axes = FALSE)
box()
polygon(c(0, 0, 5.5, 5.5), c(0, 3, 3, 0), border = "#44aaff", col = "#44aaff")
#floating.pie(x軸の位置, y軸の位置, 円の分割数, 半径)
floating.pie(1.7, 3, c(1, 1, 1, 1), radius = 0.3,
             col = as.character(TestData[1:4, 3]))
floating.pie(3, 3, c(1, 1, 1, 1), radius = 0.3,
             col = as.character(TestData[1:4, 3]))
```

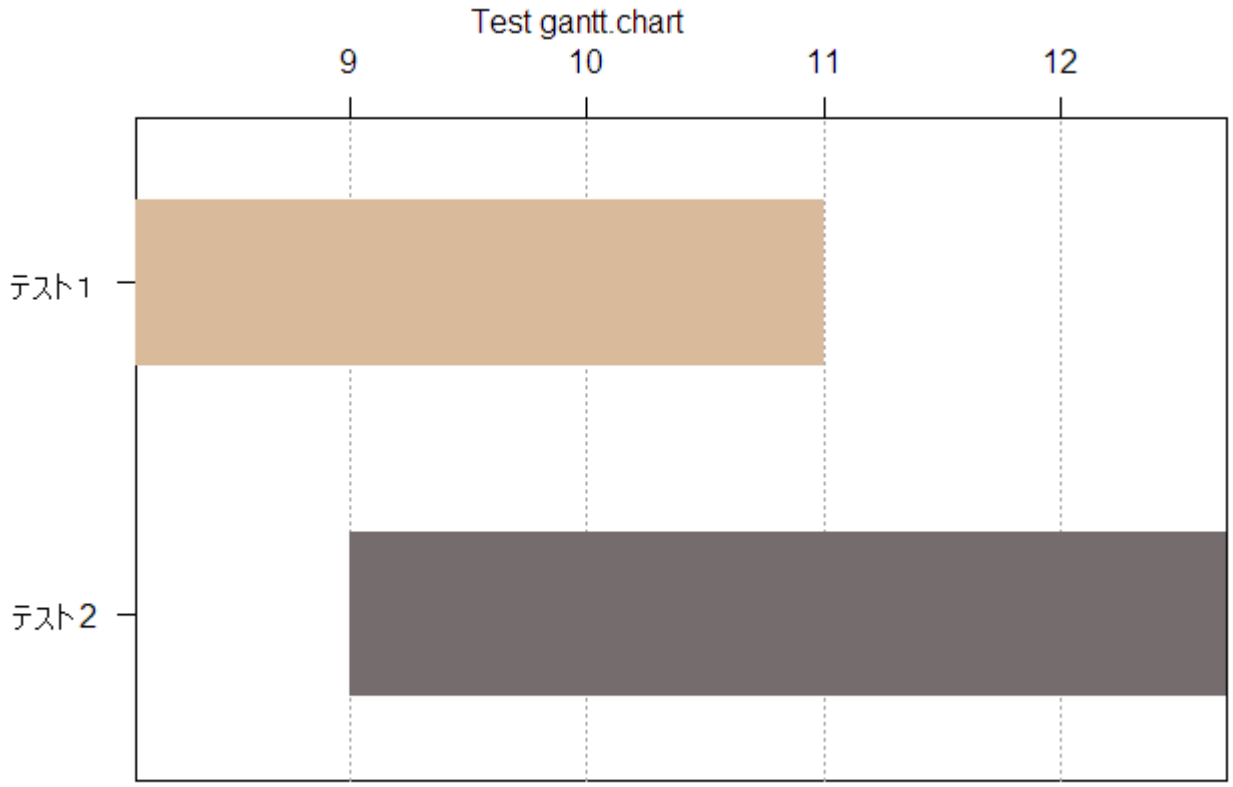
fullaxisコマンド



コマンド:

```
plot(runif(20, −1, 1), runif(20, −1, 1), xlim = c(−1, 1.5), main = "Test fullaxis",
     xlab = "X", ylab = "Y", axes = FALSE)
fullaxis(1, col = "red", col.axis = "red")
fullaxis(2, col = "blue", col.axis = "blue")
```

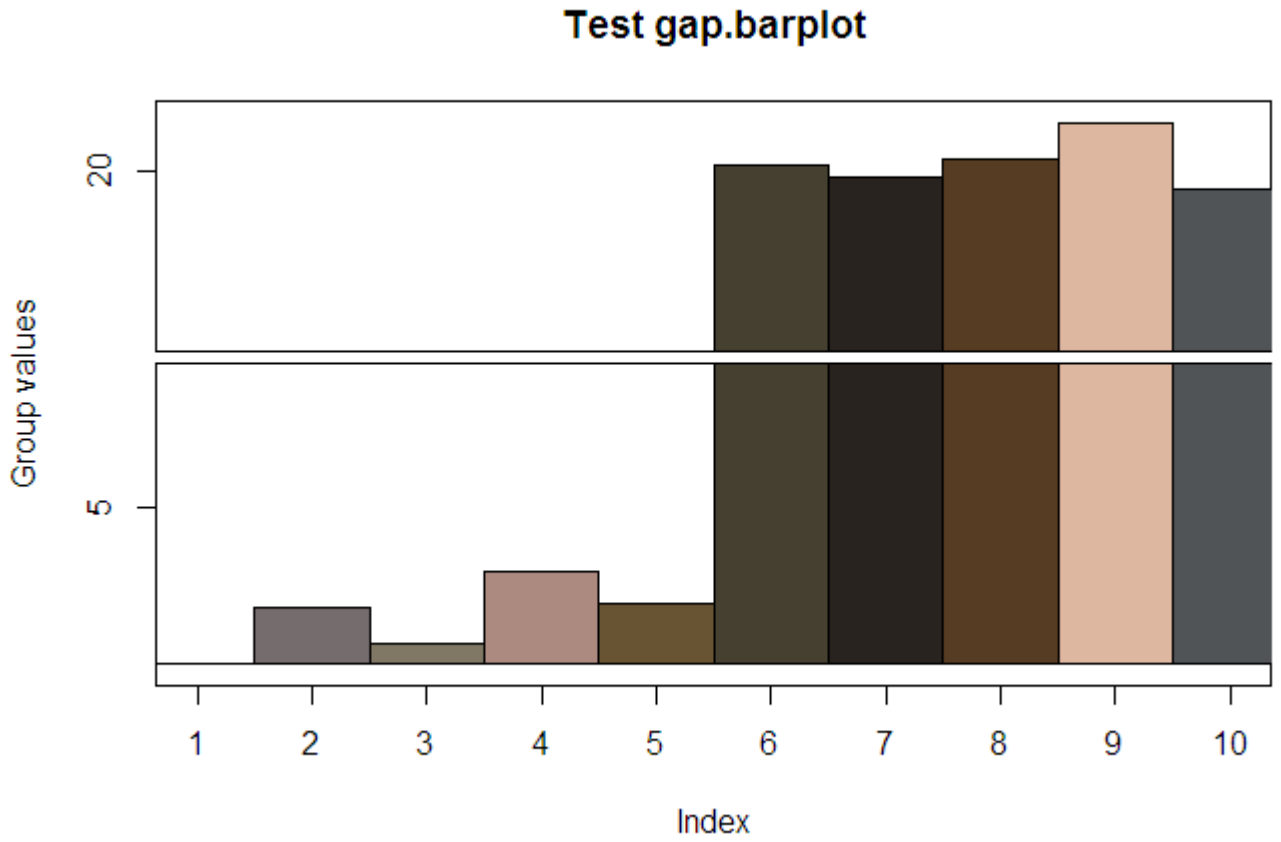
gantt.chartコマンド



コマンド:

```
PlotData <- data.frame(labels = c("テスト1", "テスト2"),
                       starts = c(8.1, 9), ends = c(11, 12.7))
gantt.chart(PlotData, vgridlab = 8:12, vgridpos = 8:12,
            main = "Test gantt.chart", taskcolors = as.character(TestData[1:2, 3]))
```

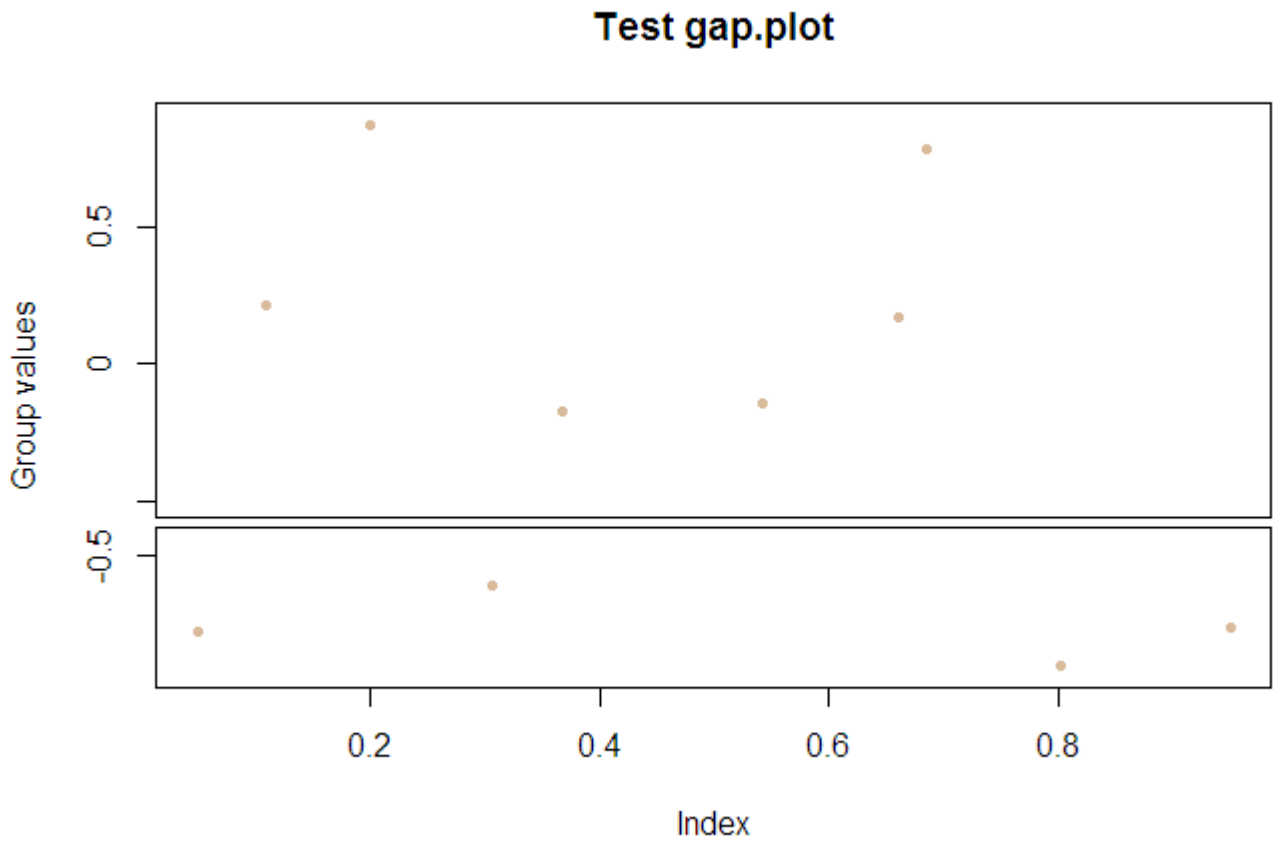
gap.barplotコマンド



コマンド:

```
twogrp <- c(rnorm(5) + 4, rnorm(5) + 20)
#軸の分割はgap = c(最小値, 最大値)で設定
gap.barplot(twogrp, gap = c(8, 16), xlab = "Index",
            ylab = "Group values", main = "Test gap.barplot",
            col = as.character(TestData[, 3]))
```

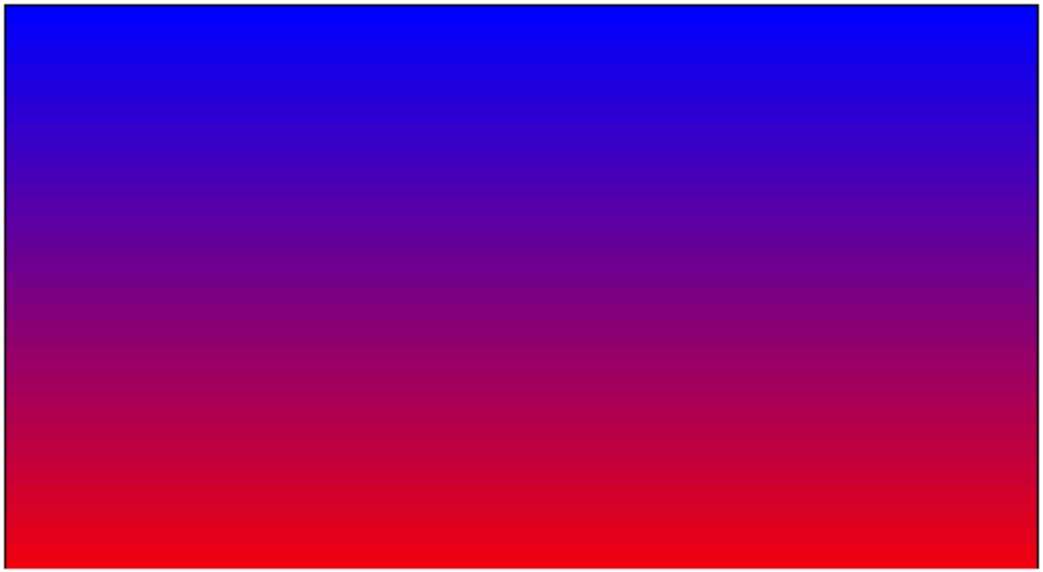
gap.plotコマンド



コマンド:

```
gap.plot(TestData[, 1], TestData[, 2], gap = c(−0.4, −0.6),
        xlab = "Index", ylab = "Group values", main = "Test gap.plot",
        col = as.character(TestData[1, 3]), pch = 20)
```

gradient.rectコマンド

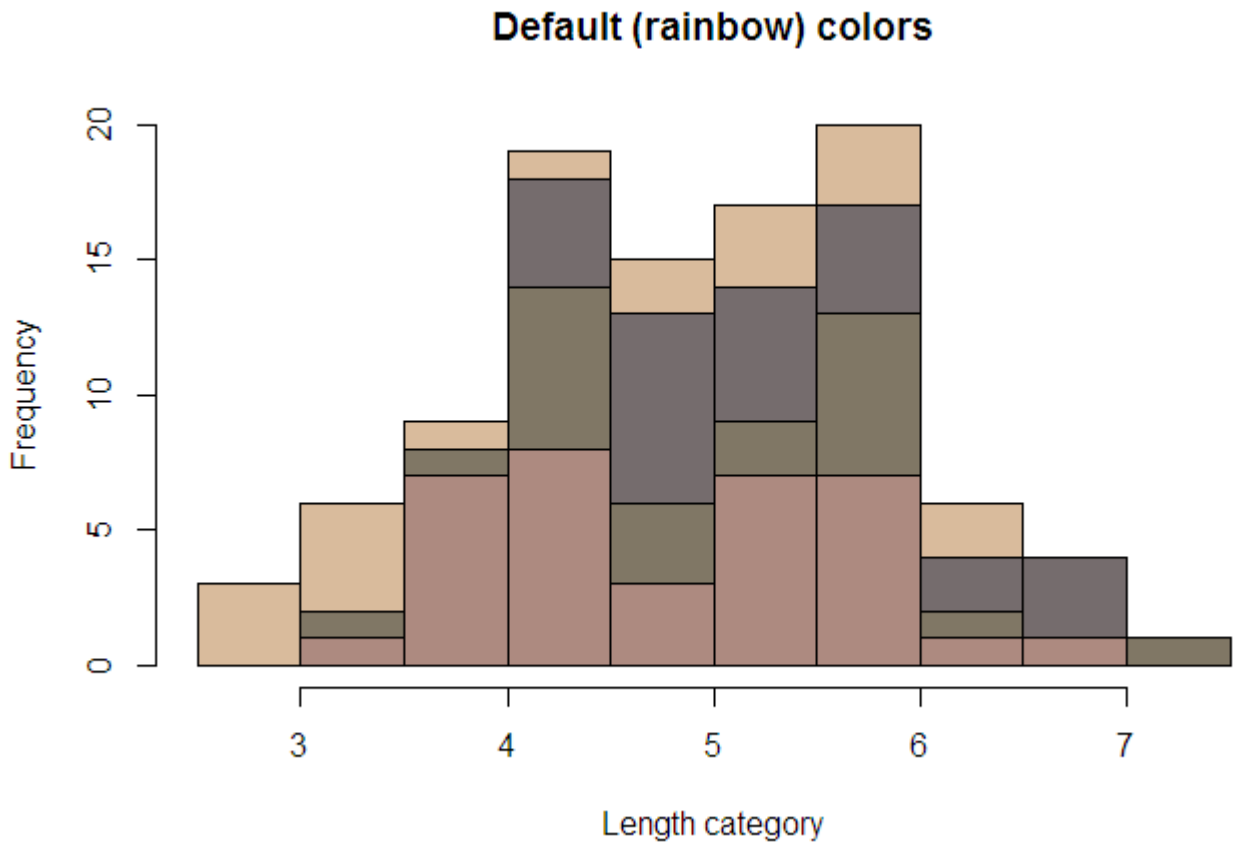


Test gradient.rect

コマンド：

```
plot(seq(nrow(TestData)), type = "n", axes = FALSE,
     xlab = "Test gradient.rect", ylab = "")
#gradient.rect(x軸開始位置, y軸開始位置, x軸終了位置, y軸終了位置)
gradient.rect(1,0,10,10, col = smoothColors("red", 38, "blue"), gradient = "y")
```

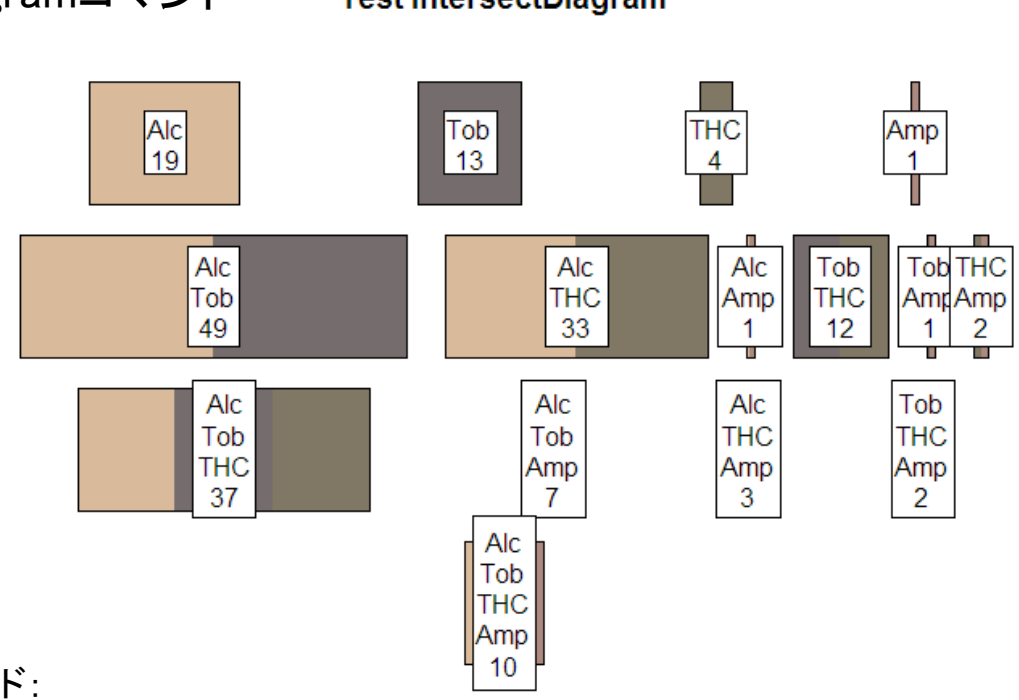
histStackコマンド



コマンド:

```
set.seed(409)
df <- data.frame(len = rnorm(100)+5,
                 grp = sample(c("A","B","C","D"), 100, replace=TRUE))
histStack(len~grp, data = df, xlab = "Test histStack", col = as.character(TestData[, 3]))
```

intersectDiagramコマンド



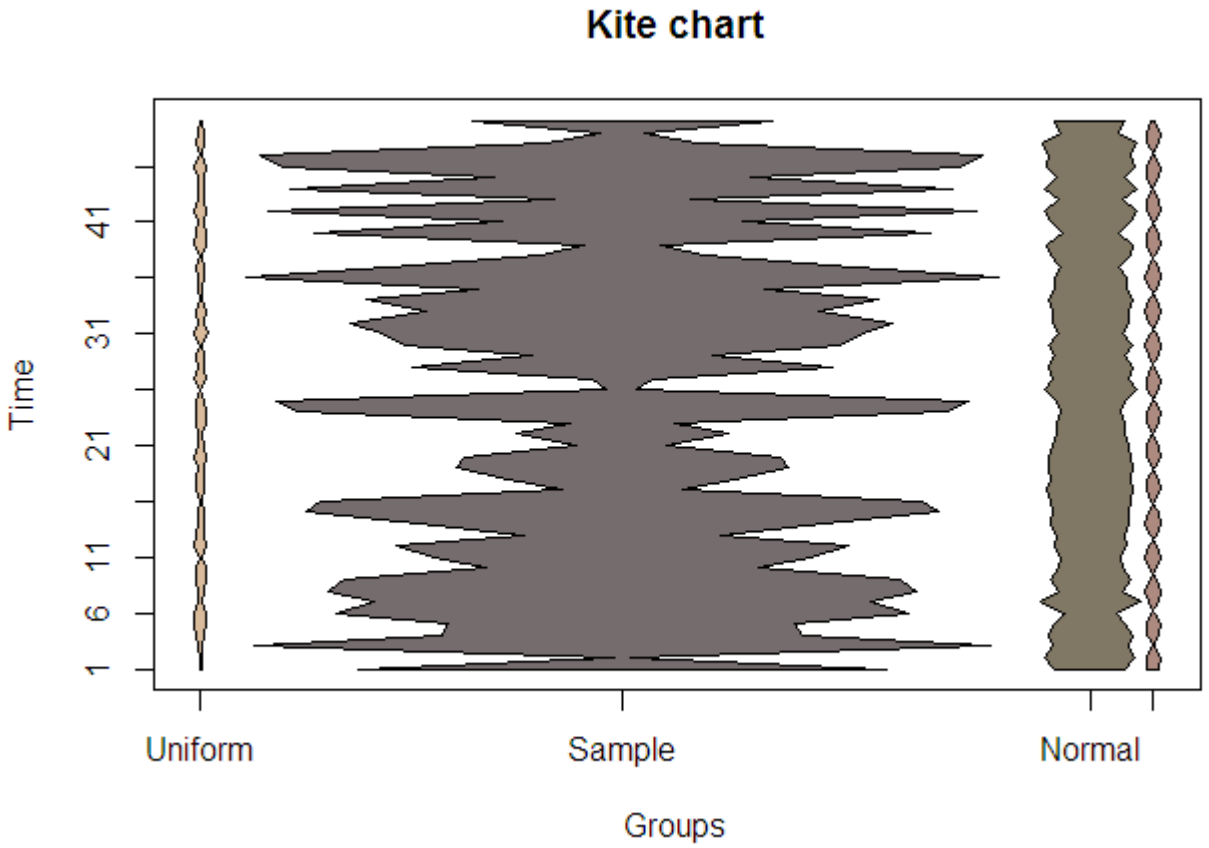
コマンド:

```
#データの作成
druguse <- matrix(c(sample(c(0, 1), 200, TRUE, prob = c(0.15, 0.85)),
                    sample(c(0, 1), 200, TRUE, prob = c(0.35, 0.65)),
                    sample(c(0, 1), 200, TRUE, prob = c(0.5, 0.5)),
                    sample(c(0, 1), 200, TRUE, prob = c(0.9, 0.1))), ncol = 4)
colnames(druguse) <- c("Alc", "Tob", "THC", "Amp")

#intersectDiagramコマンド対応のデータを作成
#組み合わせを算出します。
Druglist <- makeIntersectList(druguse,sep="¥n")

#プロット
intersectDiagram(druglist, main = "Test intersectDiagram",
                 col = as.character(TestData[, 3]), sep = "¥n")
```

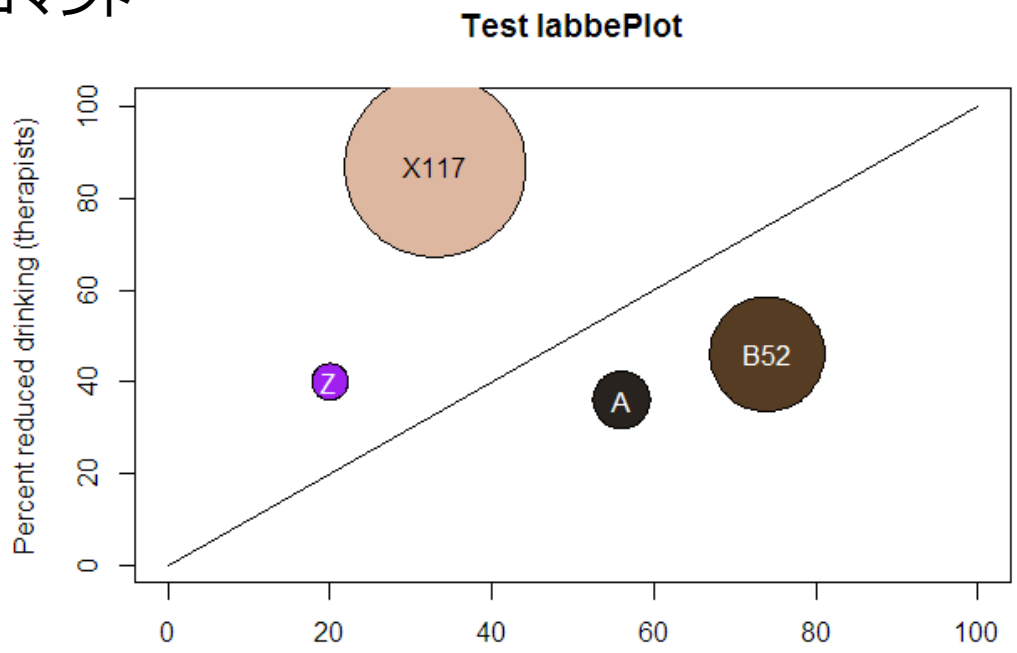
kiteChartコマンド



コマンド:

```
testmat <- matrix(c(runif(50), sample(1:50, 50), rnorm(50) + 5,
                    sin(1:50)), ncol = 50, byrow = TRUE)
kiteChart(testmat, varlabels=c("Uniform","Sample","Normal","Sine"),
          timepos = seq(1, 50, by = 5), timex = FALSE,
          col = as.character(TestData[, 3]))
```

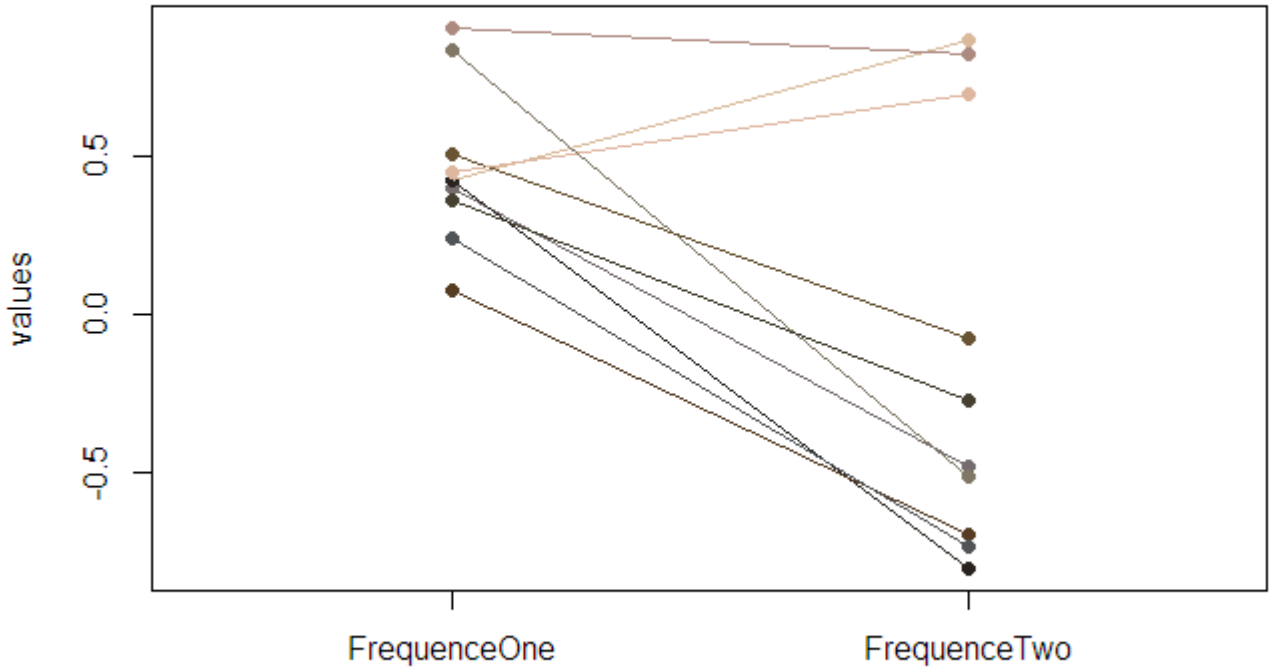
labbePlotコマンド



コマンド:

```
didf <- data.frame(subject = 1:50, interv = rep(c("therapist", "ex-drinker"), each = 25),
                   outcome = sample(c("more", "less"), 50, TRUE))
didf.tab <- table(didf$interv, didf$outcome)
#x軸, y軸, 半径の設定
didf2 <-c(74, 46, 200)
didf3 <- c(33, 87, 500)
x <- list(didf.tab, didf2, didf3)
labbecol <- list("#28231e", "#563c22", "#deb7a0")
labbePlot(x, main = "Test labbePlot",
          xlab = "Percent reduced drinking (ex-drinkers)",
          ylab = "Percent reduced drinking (therapists)",
          labels = list("A","B52","X117"), col = labbecol)
labbePlot(list(c(20, 40, 20)), col = list("purple"), labels = list("Z"), add=TRUE)
```

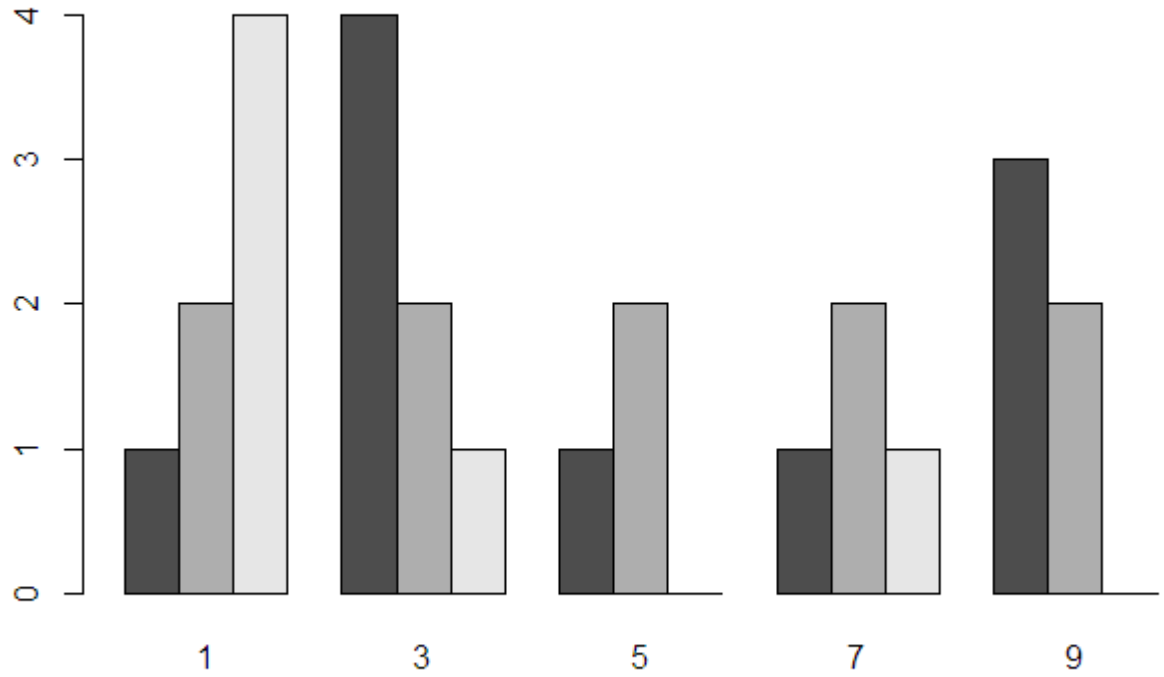
ladderplotコマンド



コマンド：

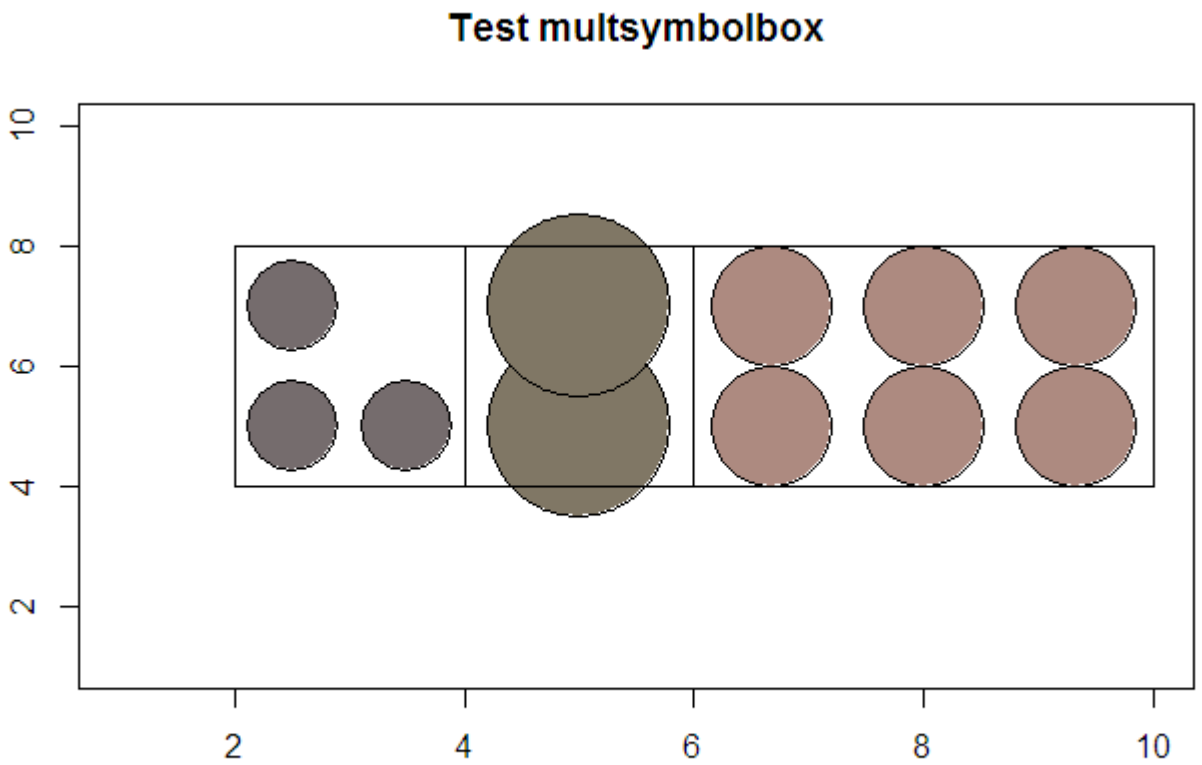
```
ladderplot(TestData[, 1:2], col = as.character(TestData[, 3]))
```

multhistコマンド



コマンド:
multhist(list(runif(10) * 10, 1:10, c(1, 1, 1, 1, 4, 8)))

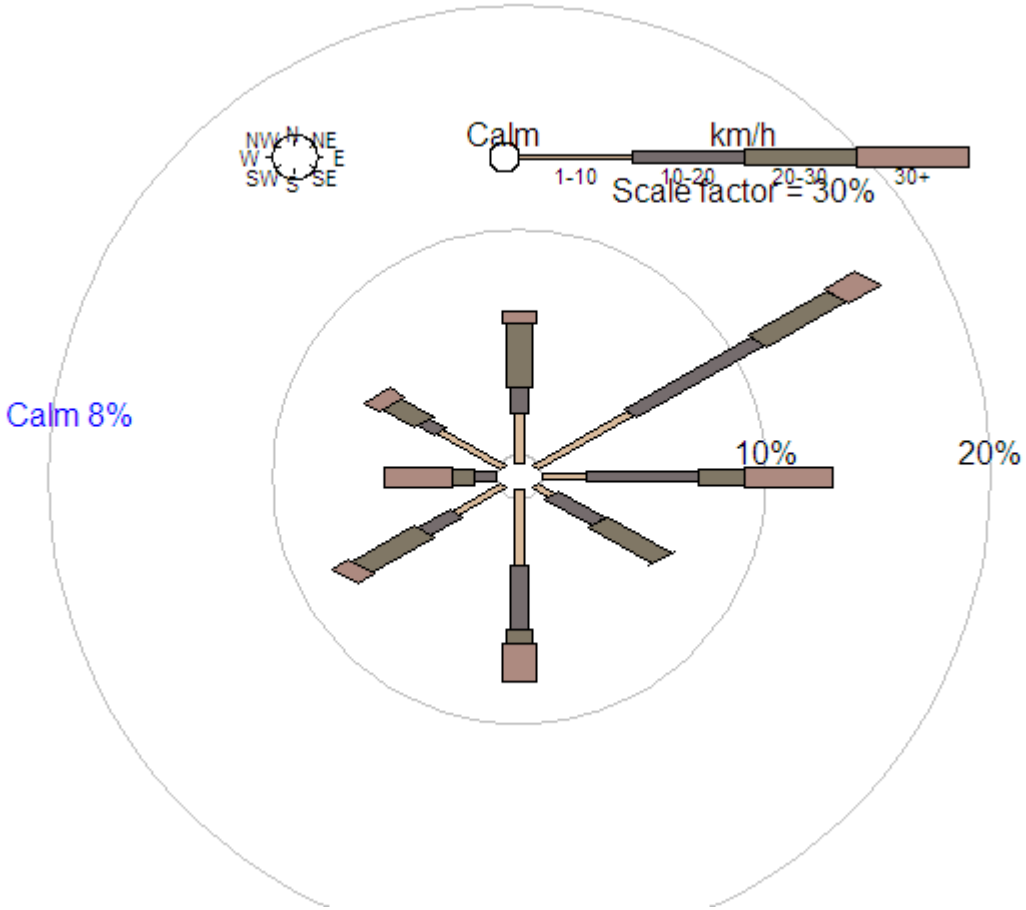
multsymbolboxコマンド



コマンド:

```
plot(1:10, 1:10, type = "n", xlab = "", ylab = "", main = "Test multsymbolbox")
#multsymbolbox(x軸の開始, y軸の開始, x軸の終了, y軸の終了)
#複数の箱を描写する時はx軸の終了の設定値をx軸の開始値と合わせると
#上手くプロットされます。
multsymbolbox(c(2, 4, 6), 4, c(4, 6, 10), 8, tot = c(3, 2, 6),
                    bg = as.character(TestData[2:4, 3]))
```

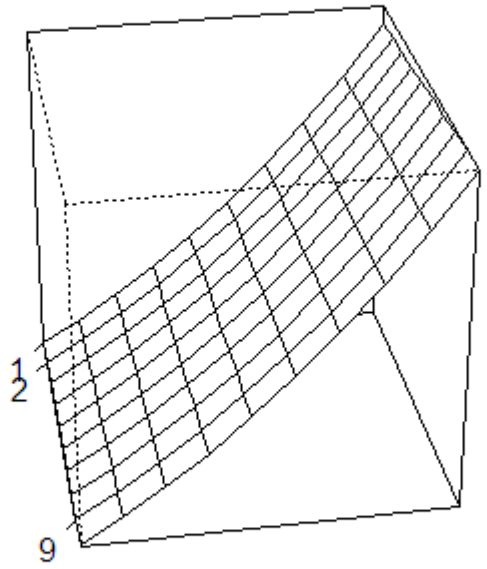
multhistコマンド



コマンド:

```
windagg <-  matrix(c(8,0,0,0,0,0,0,0,4,6,2,1,6,3,0,4,2,8,5,3,5,2,1,1,
                     5,5,2,4,1,4,1,2,1,2,4,0,3,1,3,1), nrow = 5, byrow = TRUE)
oz.windrose(windagg, speed.col = as.character(TestData[, 3]))
```

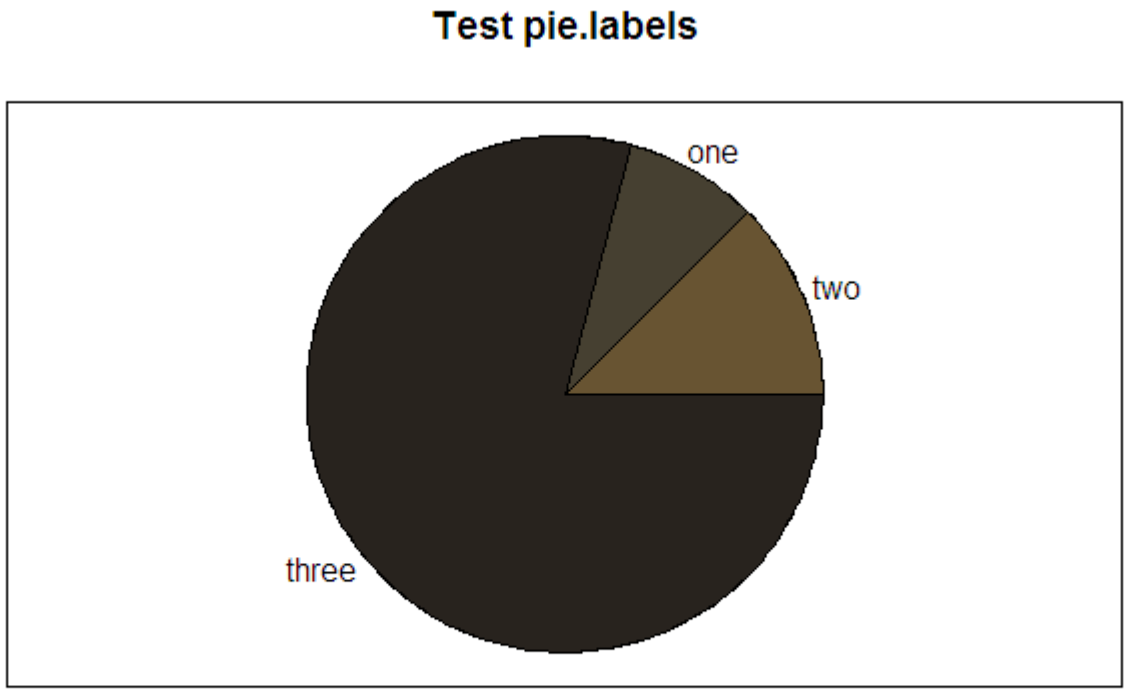
perspxコマンド



コマンド:

```
x <- 1:10
y <- 1:10
z <- outer(x, y, function(x, y) { 3*sin(2*pi*x)/(2*pi*x) + exp(y/10)+(x*y)/1000 })
pp <- perspx(x, y , z, ticktype = "detailed", phi = 30, theta = 80, nticks = 3, r=10,
                    axes = FALSE)
paxis3d("X-", pp, at = c(1, 2, 9))
```

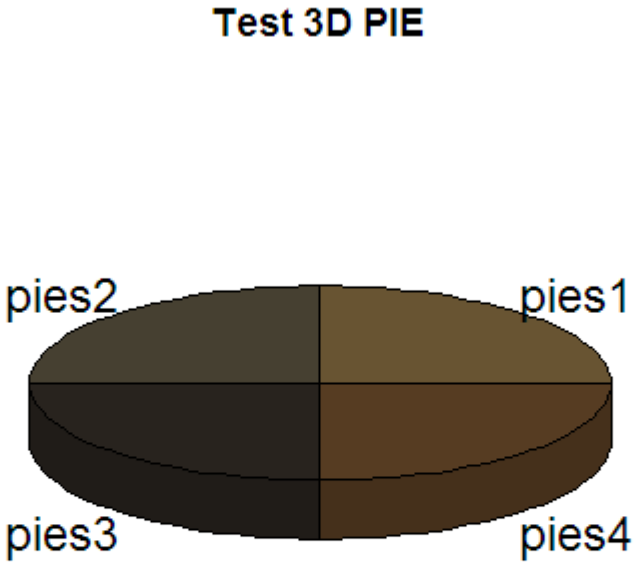
floating.pieコマンド



コマンド:

```
plot(1:5, type = "n", axes = FALSE, xlab = "", ylab = "", main = "Test pie.labels")
box()
#floating.pie(x, y, 合計が100になるように分割した数列)
bisect.angles <- floating.pie(3, 3, c(15, 10, 95), col = as.character(TestData[5:7, 3]))
#ラベルの付与
pie.labels(3, 3, bisect.angles, c("two","one","three"))
```

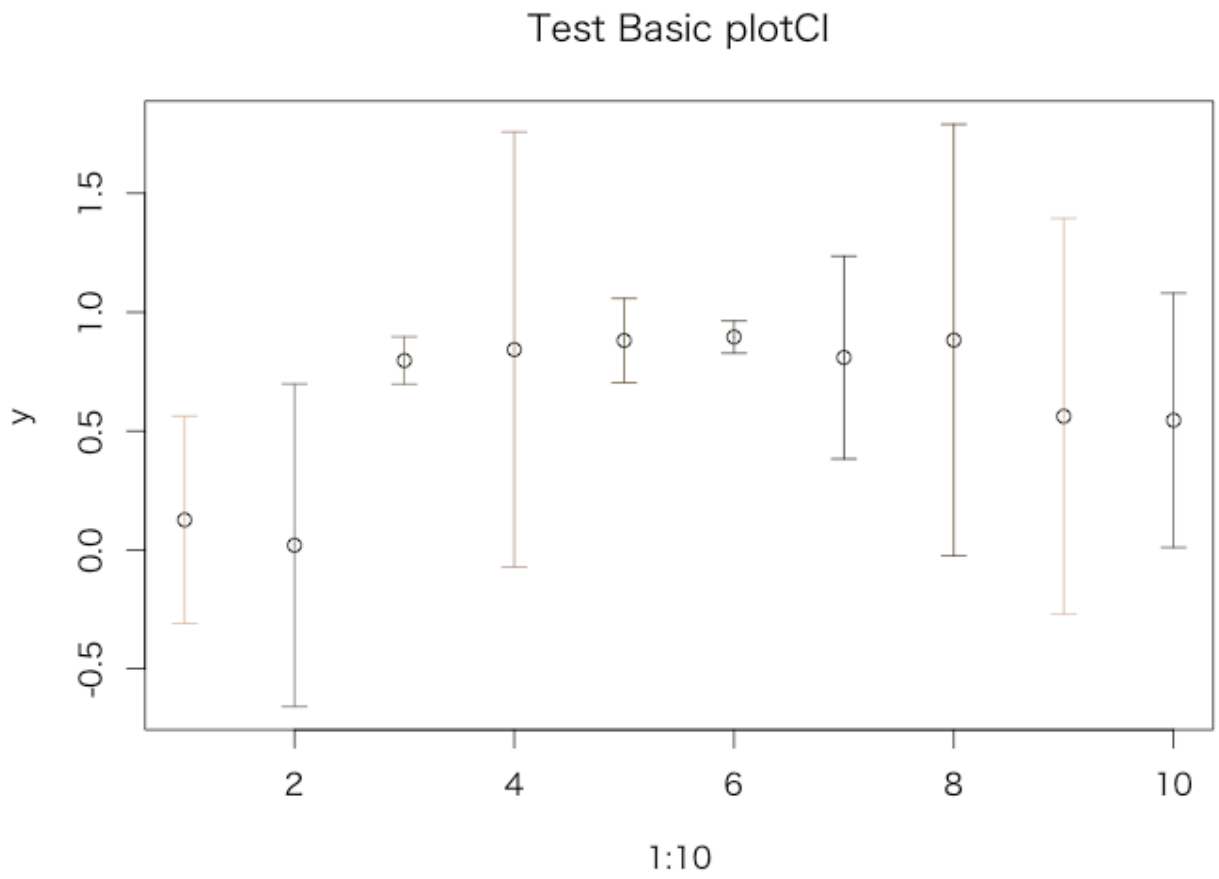
pie3Dコマンド



コマンド:

```
#割合は自動で計算されます
bisectors <- pie3D(c(1, 1, 1, 1), explode = 0,
                   main = "Test 3D PIE",
                   col = as.character(TestData[5:8, 3]))
pielabels <- c("pies1", "pies2", "pies3", "pies4")
pie3D.labels(bisectors, labels = pielabels)
```

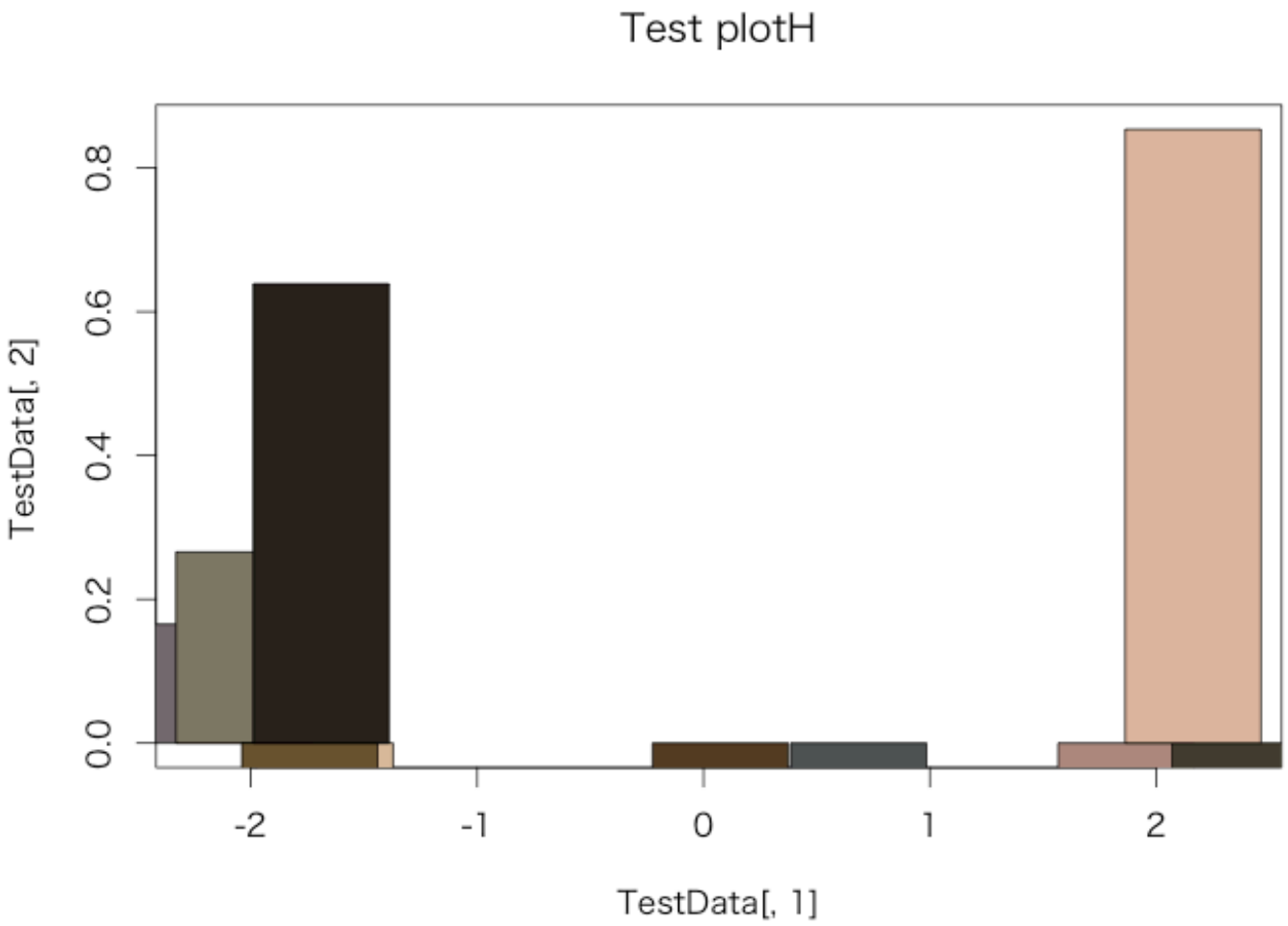
plotCIコマンド



コマンド:

```
y <- runif(10)
err <- runif(10)
plotCI(1:10, y, err,main = "Test Basic plotCI",
        scol = as.character(TestData[, 3]), slty = 1)
```

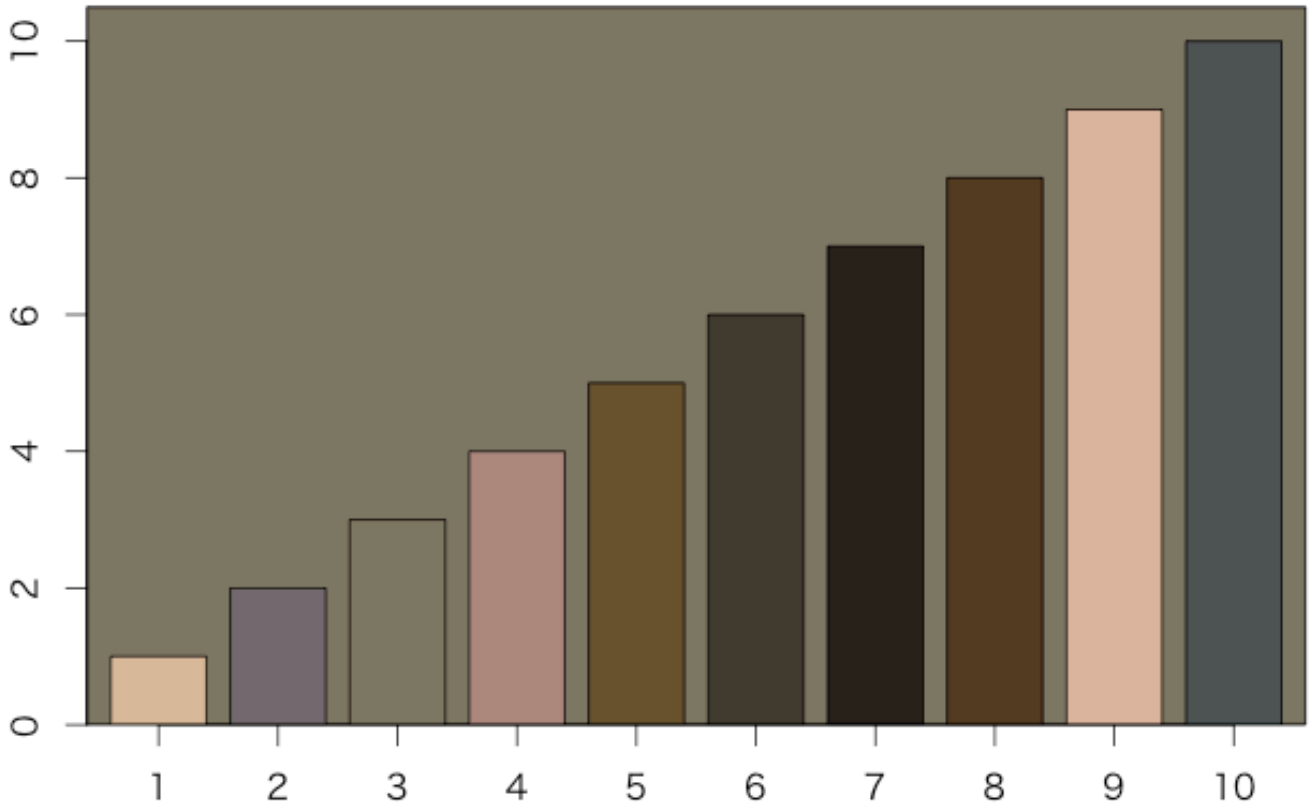
plotHコマンド



コマンド:

```
plotH(TestData[, 2]~TestData[, 1], data = TestData,
        col = as.character(TestData[, 3]), main = "Test plotH")
```

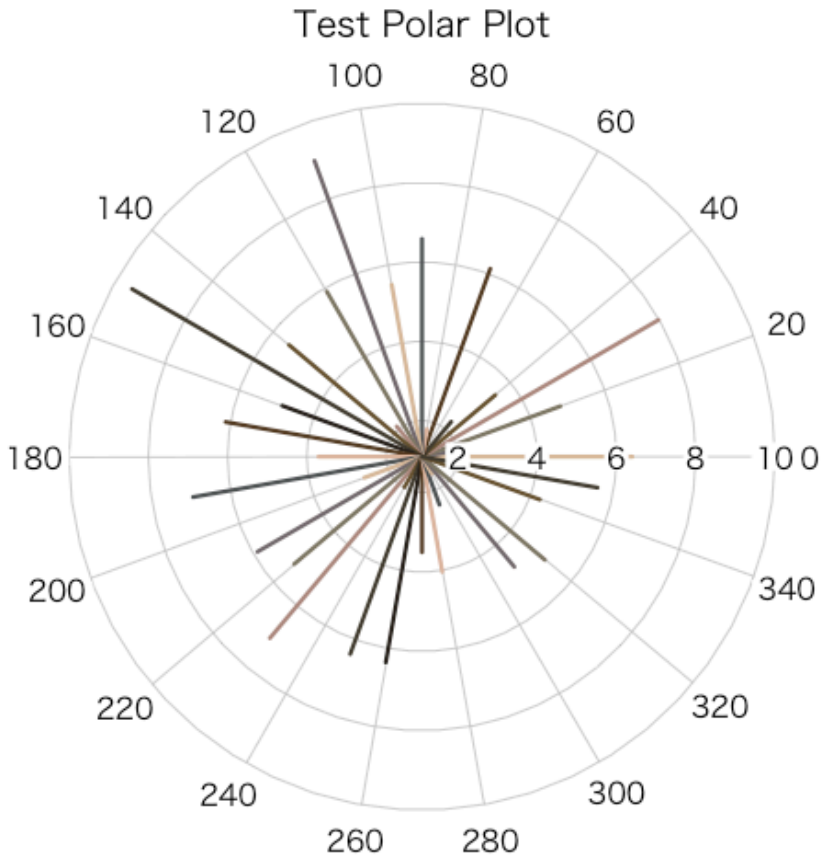
barpコマンド



コマンド:

```
barp(1:10, do.first = "plot_bg(as.character(TestData[3, 3]))",
        col = as.character(TestData[, 3]))
```

polar.plotコマンド



コマンド:

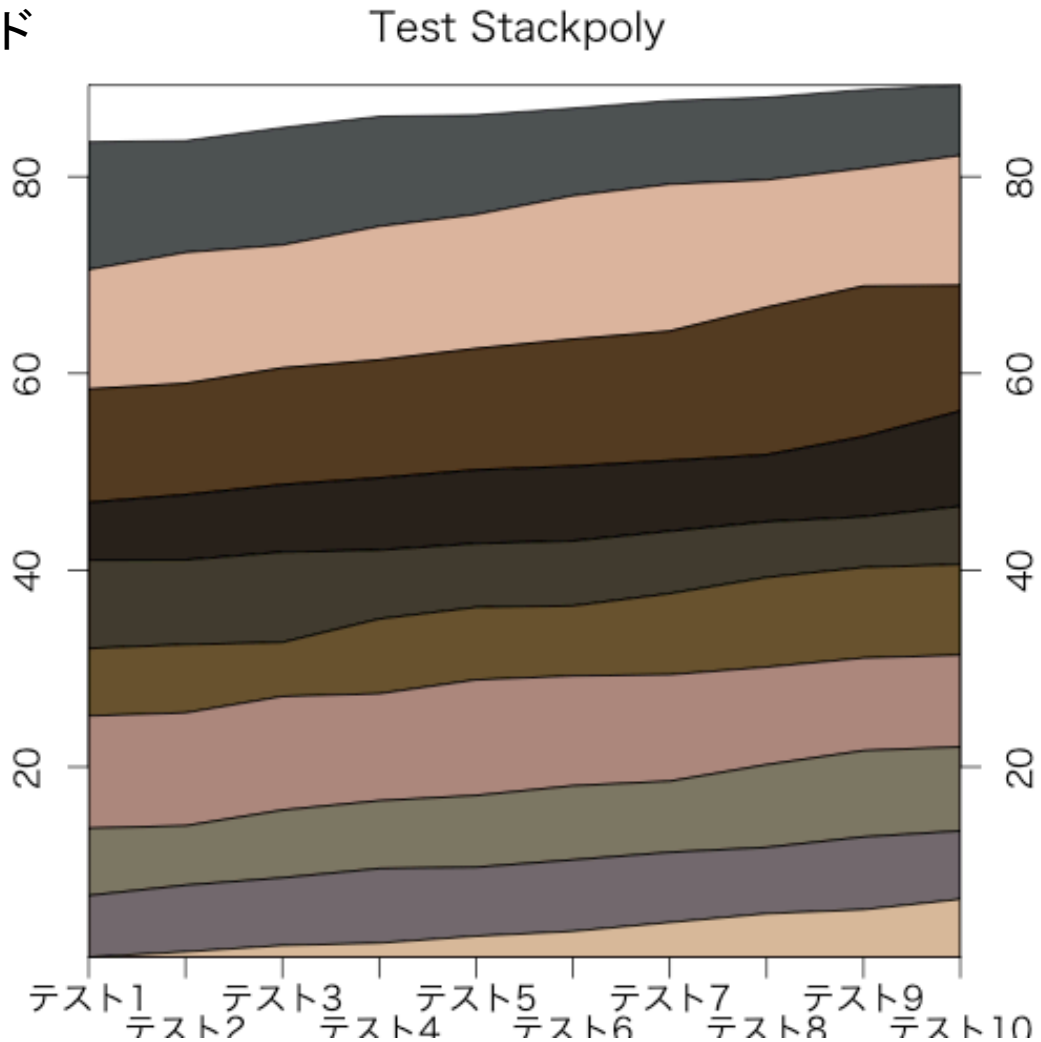
```
testlen <- c(rnorm(36)*2+5)
testpos <- seq(0, 350, by = 10)
polar.plot(testlen, testpos, main = "Test Polar Plot",
           lwd = 3, line.col = as.character(TestData[, 3]))
```

smoothColorsコマンド

コマンド:

```
plot(1:10, main = "Test smoothColors", type = "n", axes = FALSE)
box()
rect(1:7, 1:7, 3:9, 3:9, col = smoothColors(as.character(TestData[3, 3]), 2,
                                            as.character(TestData[5, 3]), 2,
                                            as.character(TestData[4, 3])))
```

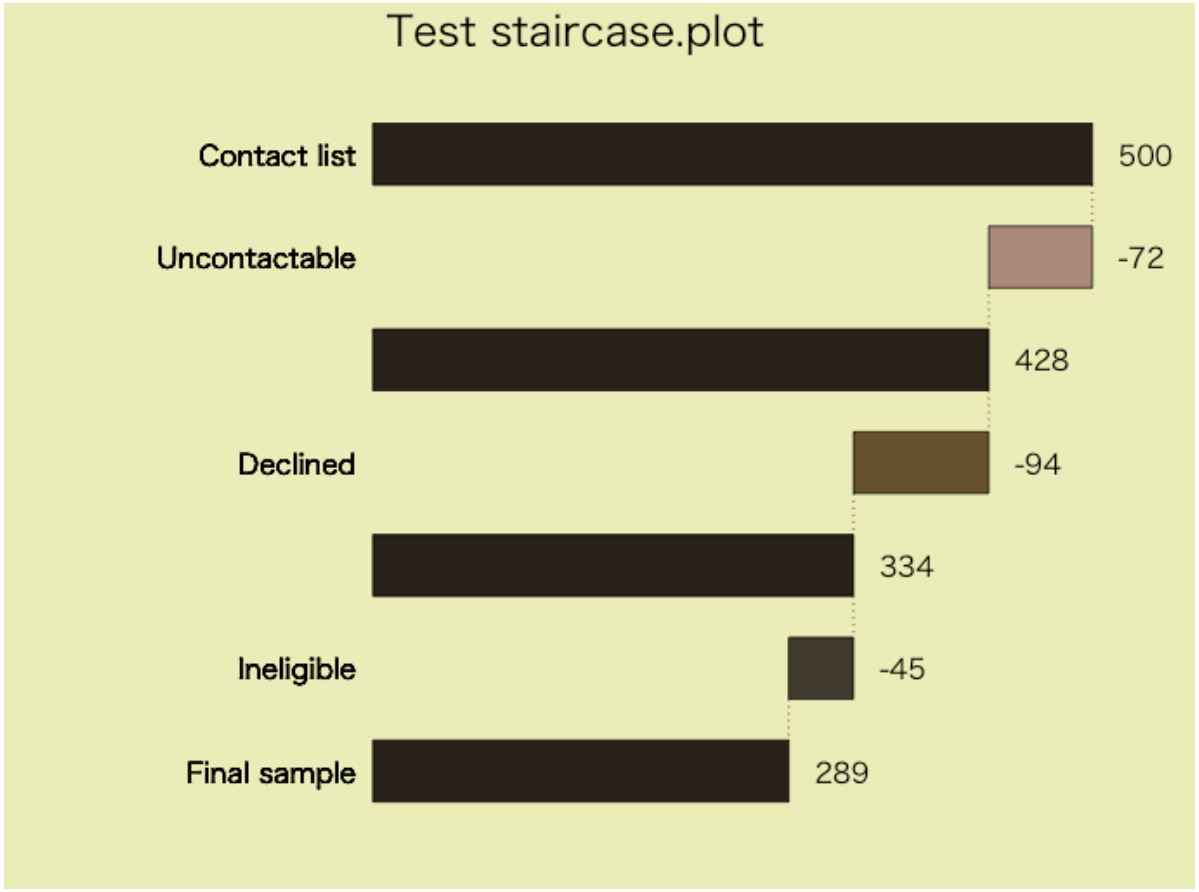
stackpolyコマンド



コマンド:

```
testx <- matrix(abs(rnorm(100)), nrow = 10)
stackpoly(matrix(cumsum(testx), nrow = 10), main = "Test Stackpoly ",
          xaxlab = row.names(TestData), border = "black", staxx=TRUE,
          col = as.character(TestData[, 3]))
```

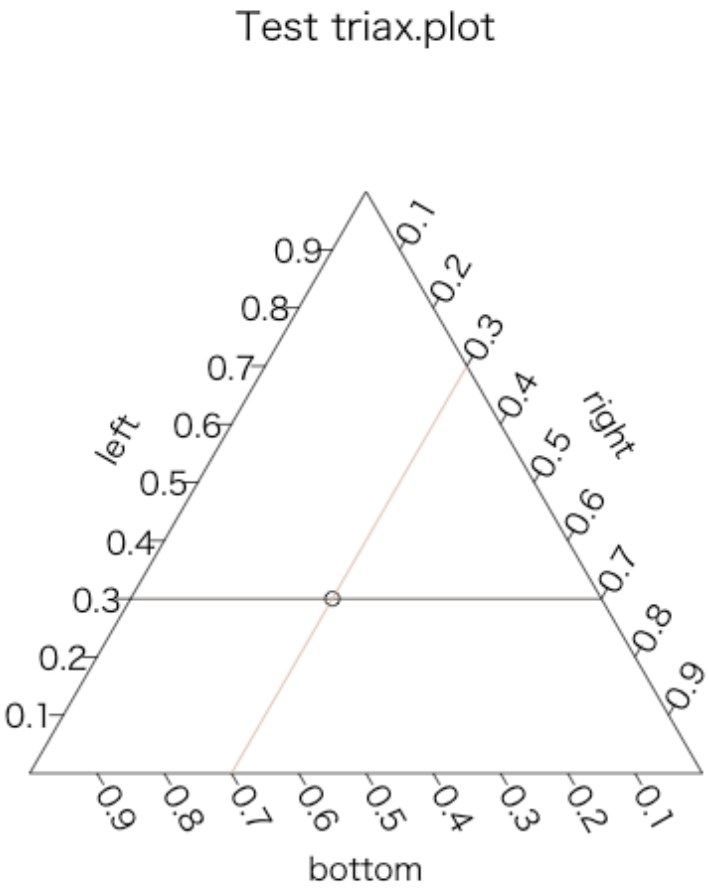
stackpolyコマンド



コマンド:

```
sample_size <- c(500, -72, 428, -94, 334, -45, 289)
totals <- c(TRUE, FALSE, TRUE, FALSE, TRUE, FALSE, TRUE)
labels <- c("Contact list", "Uncontactable", "", "Declined", "", "Ineligible", "Final sample")
staircase.plot(sample_size, totals, labels, main = "Test staircase.plot",
               total.col = "#28231e", inc.col = as.character(TestData[4:6, 3]),
               bg.col = "#eeeebb", direction = "s")
```

triax.plotコマンド



コマンド:

```
triax.return <- triax.plot(data.frame(bottom = 0.4, right = 0.3, left = 0.3),
                           main = "Test triax.plot", no.add=FALSE)
triax.abline(l = 0.3, col = "#28231e")
triax.abline(r = 0.3, col = "#deb7a0")
```