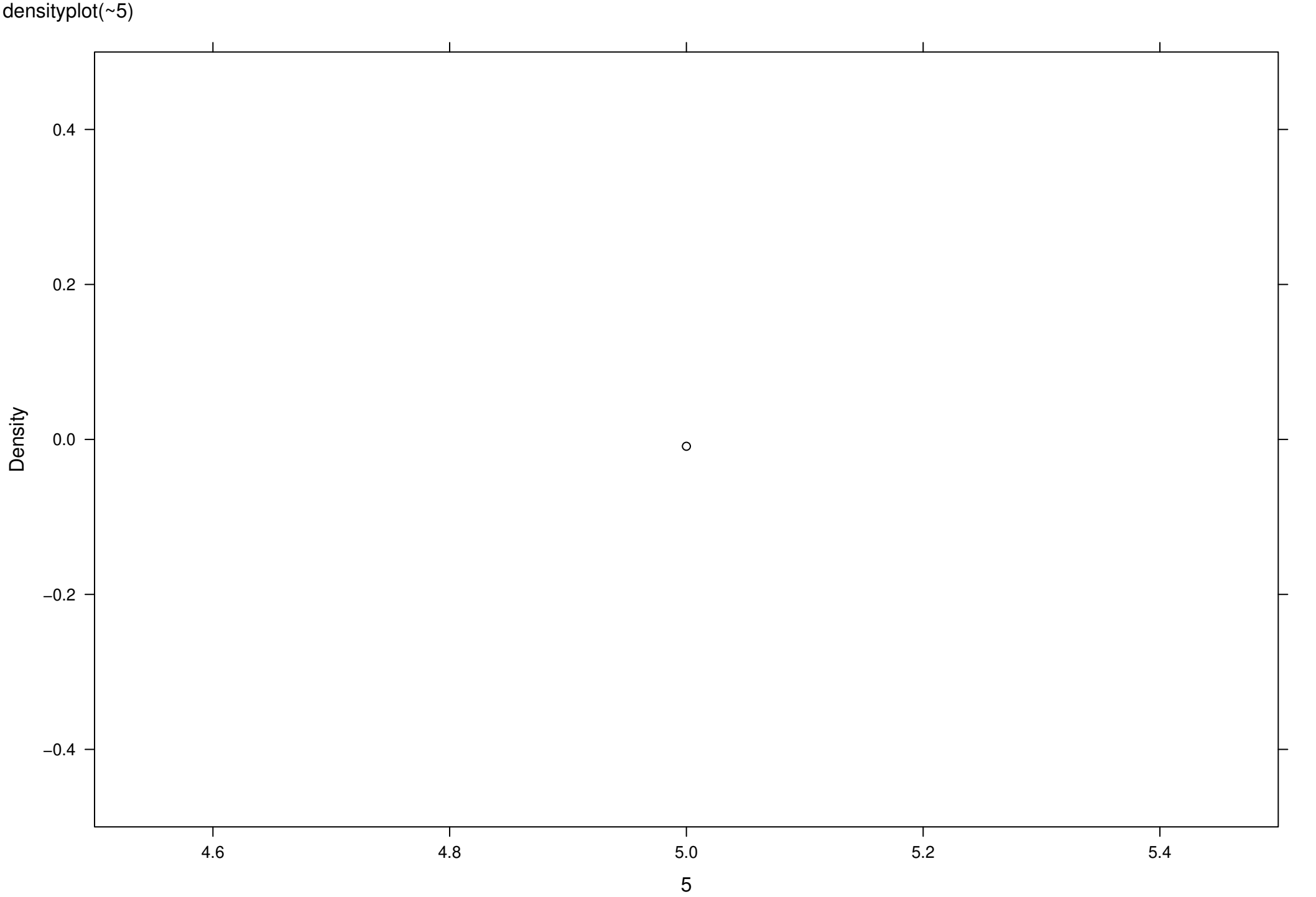
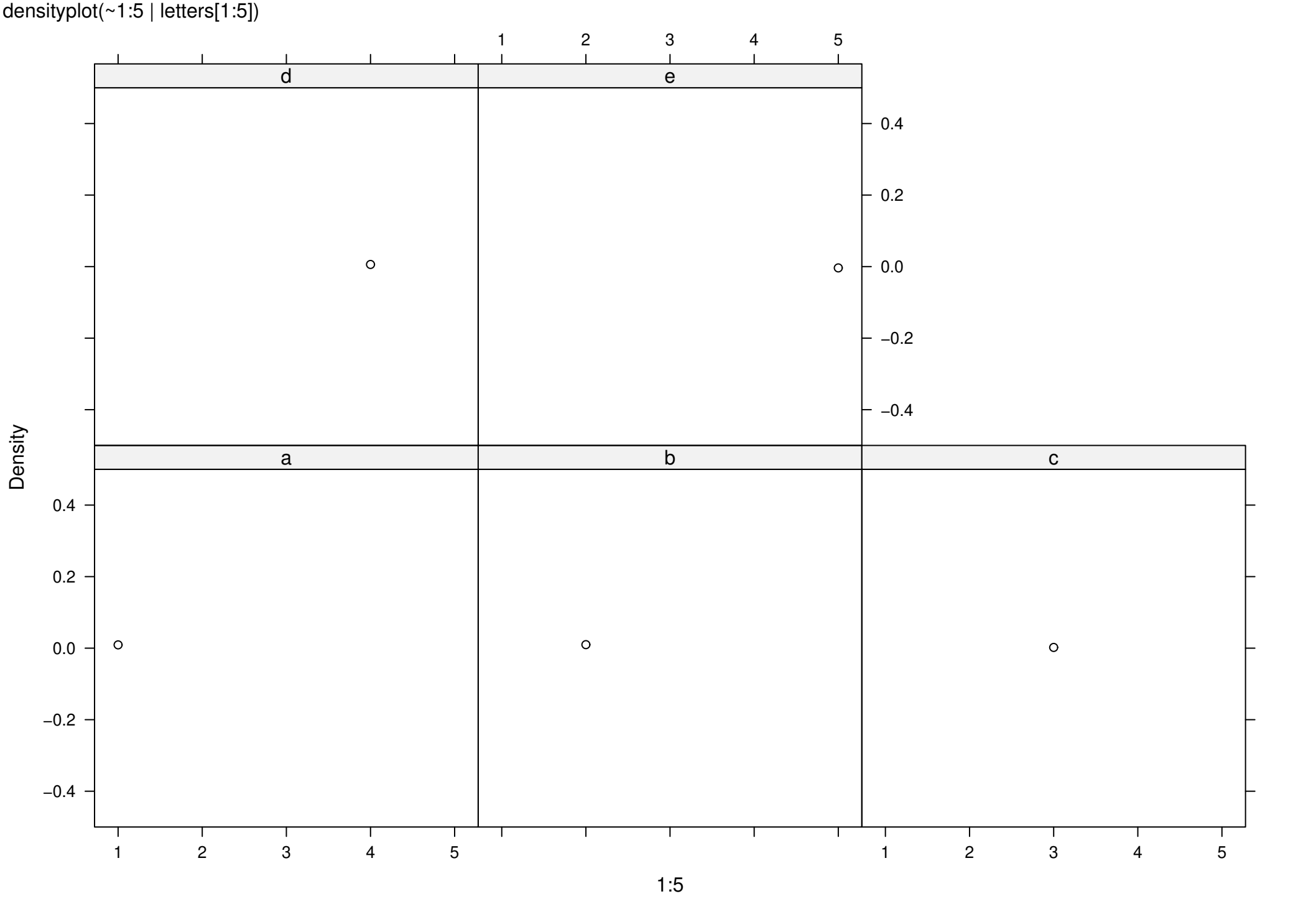
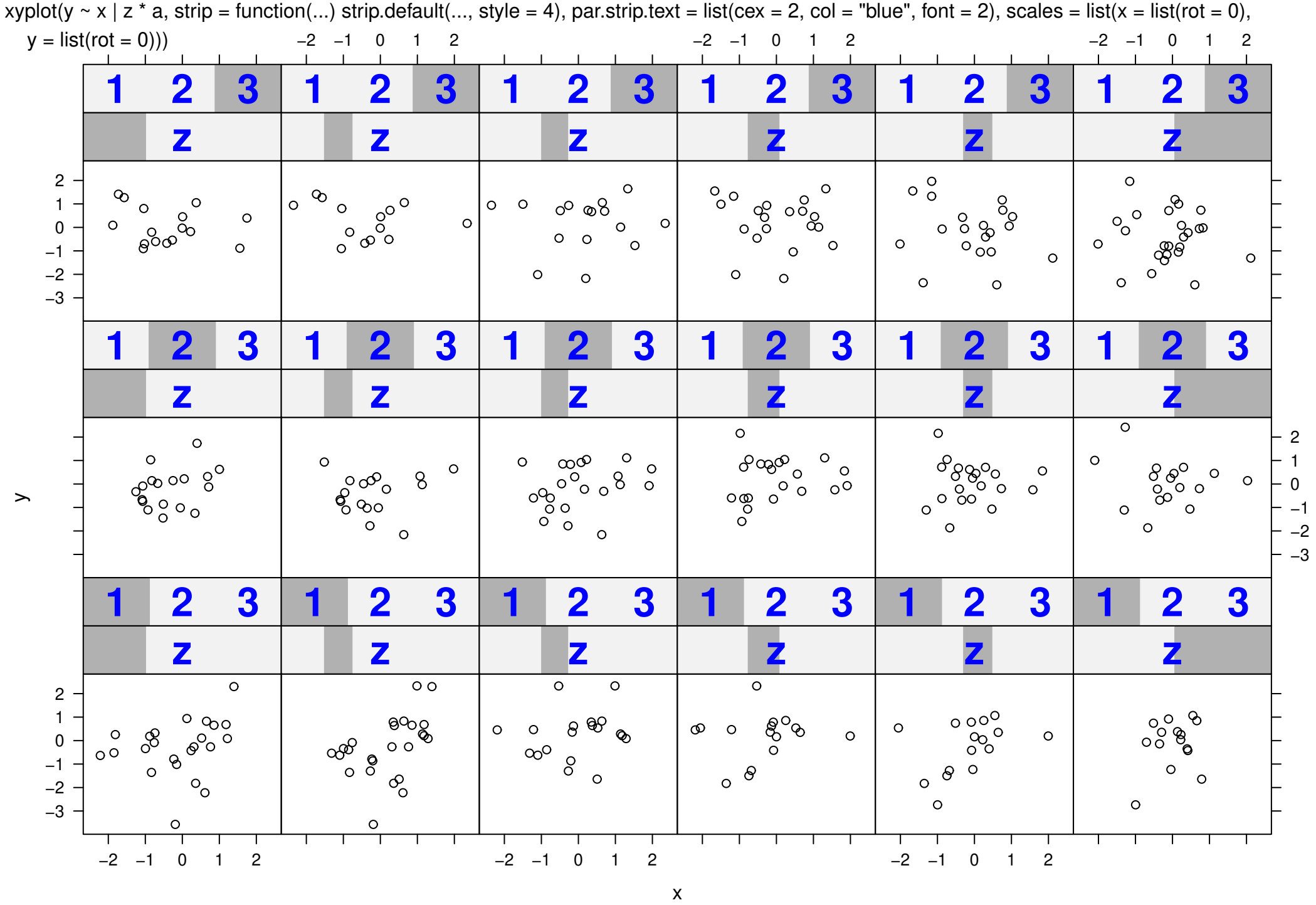
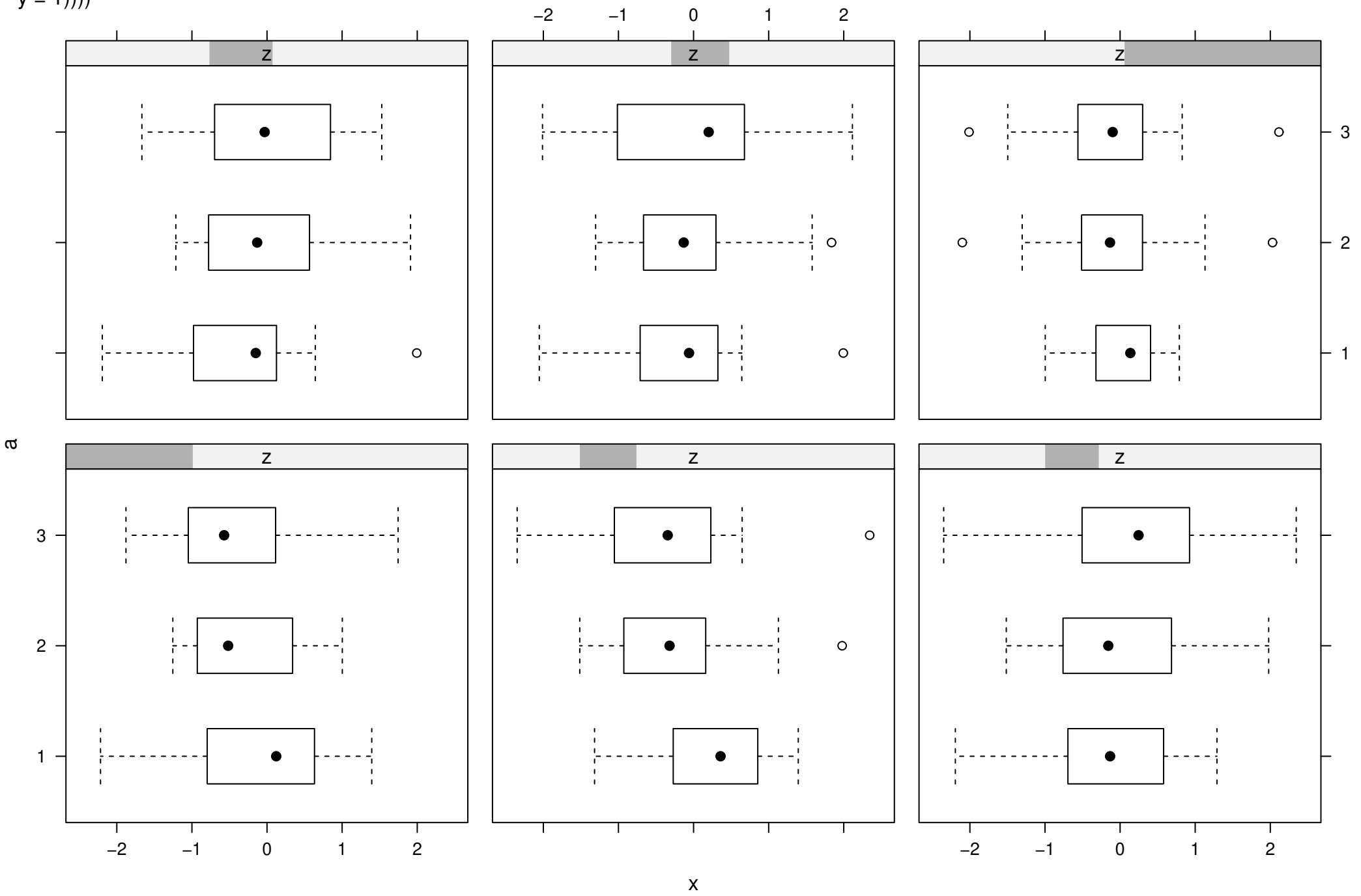
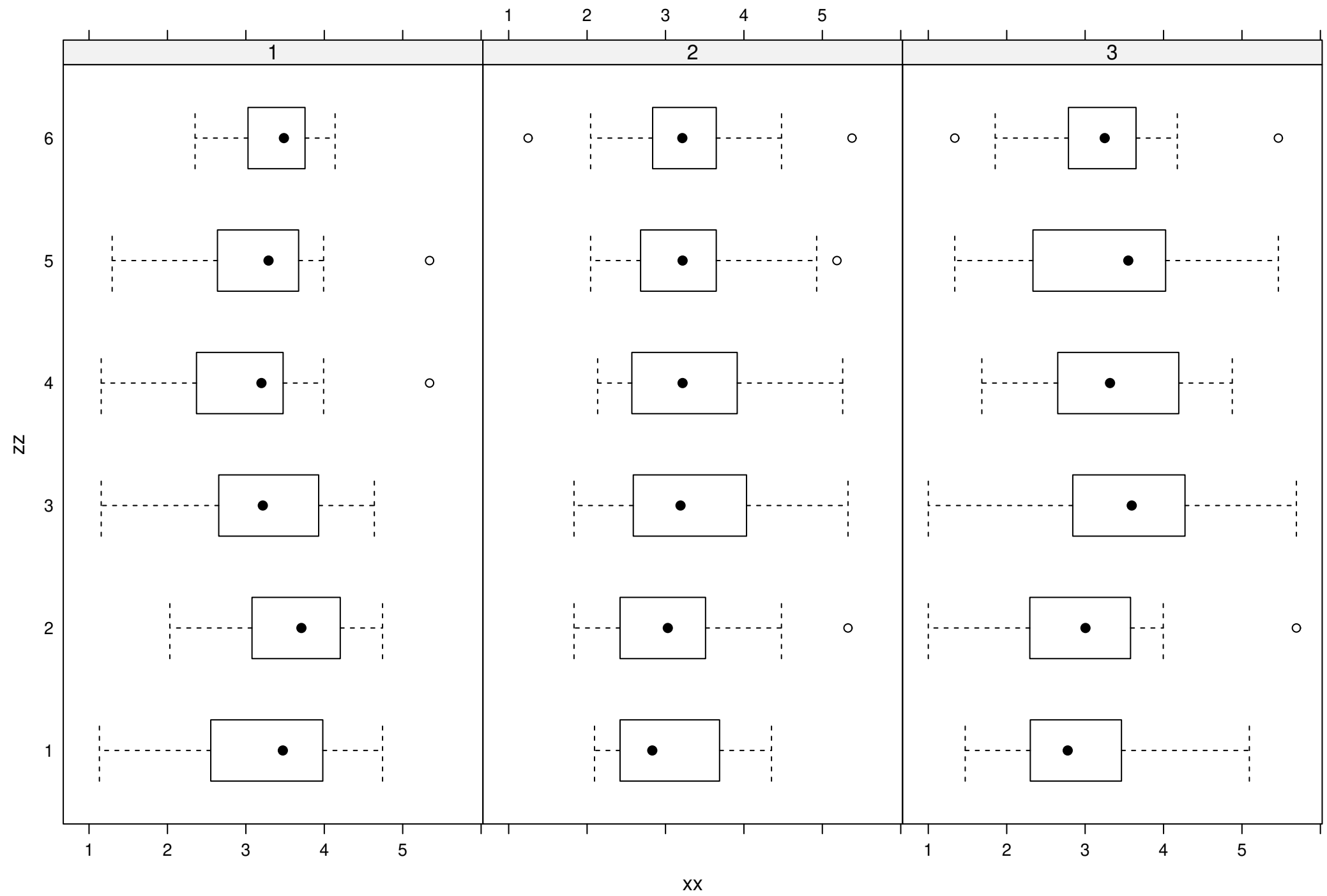

densityplot(~5)
0.4
0.2
0.0
−0.2
−0.4
Density
4.6
4.8
5.0
5.2
5.4
5

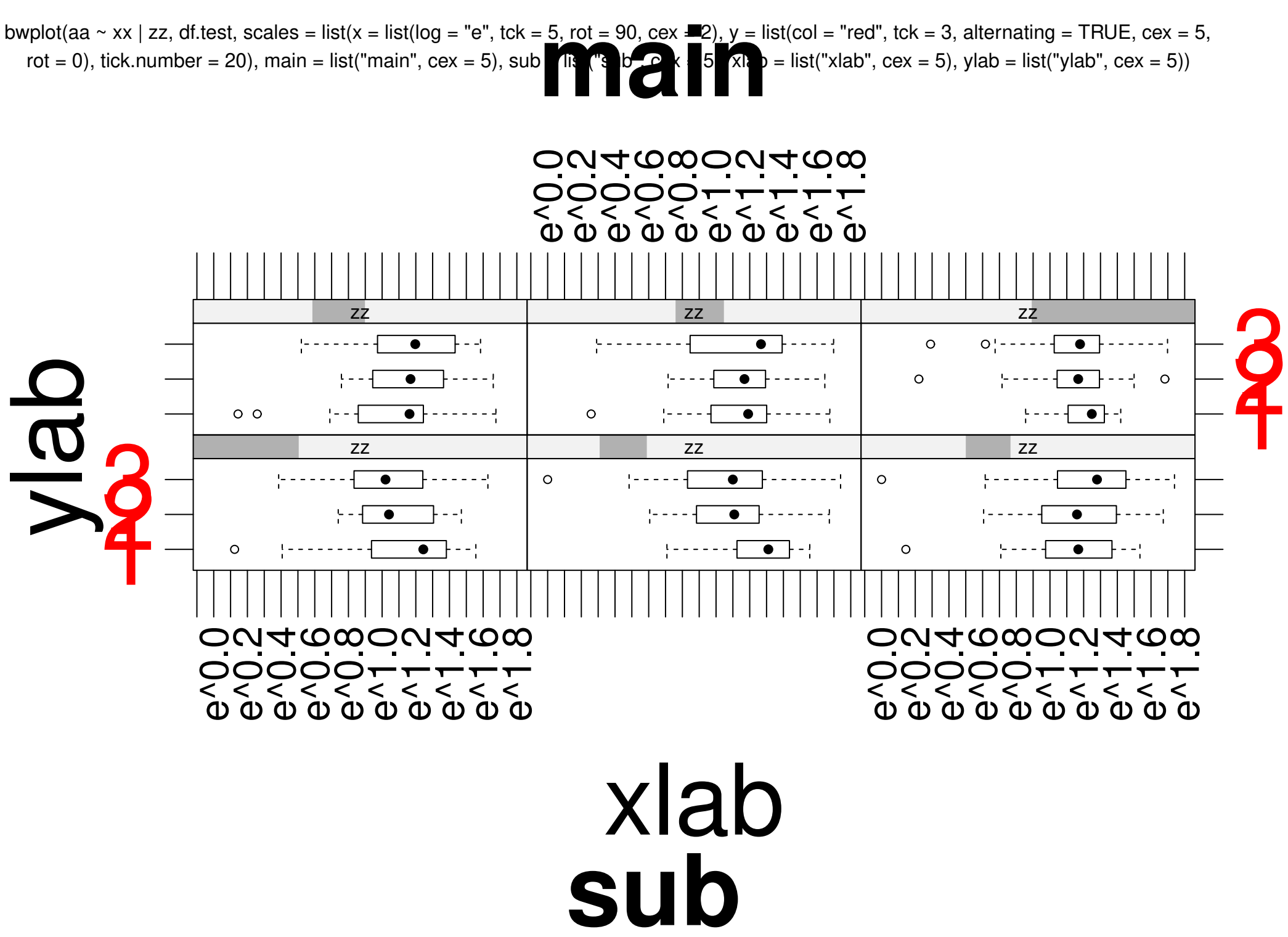

densityplot(~1:5 | letters[1:5])
d
e
a
b
c
Density
1:5
1
2
3
4
5
0.4
0.2
0.0
−0.2
−0.4


xyplot(y ~ x | z * a, strip = function(...) strip.default(..., style = 4), par.strip.text = list(cex = 2, col = "blue", font = 2), scales = list(x = list(rot = 0),
y = list(rot = 0)))
1 2 3
z
x
y

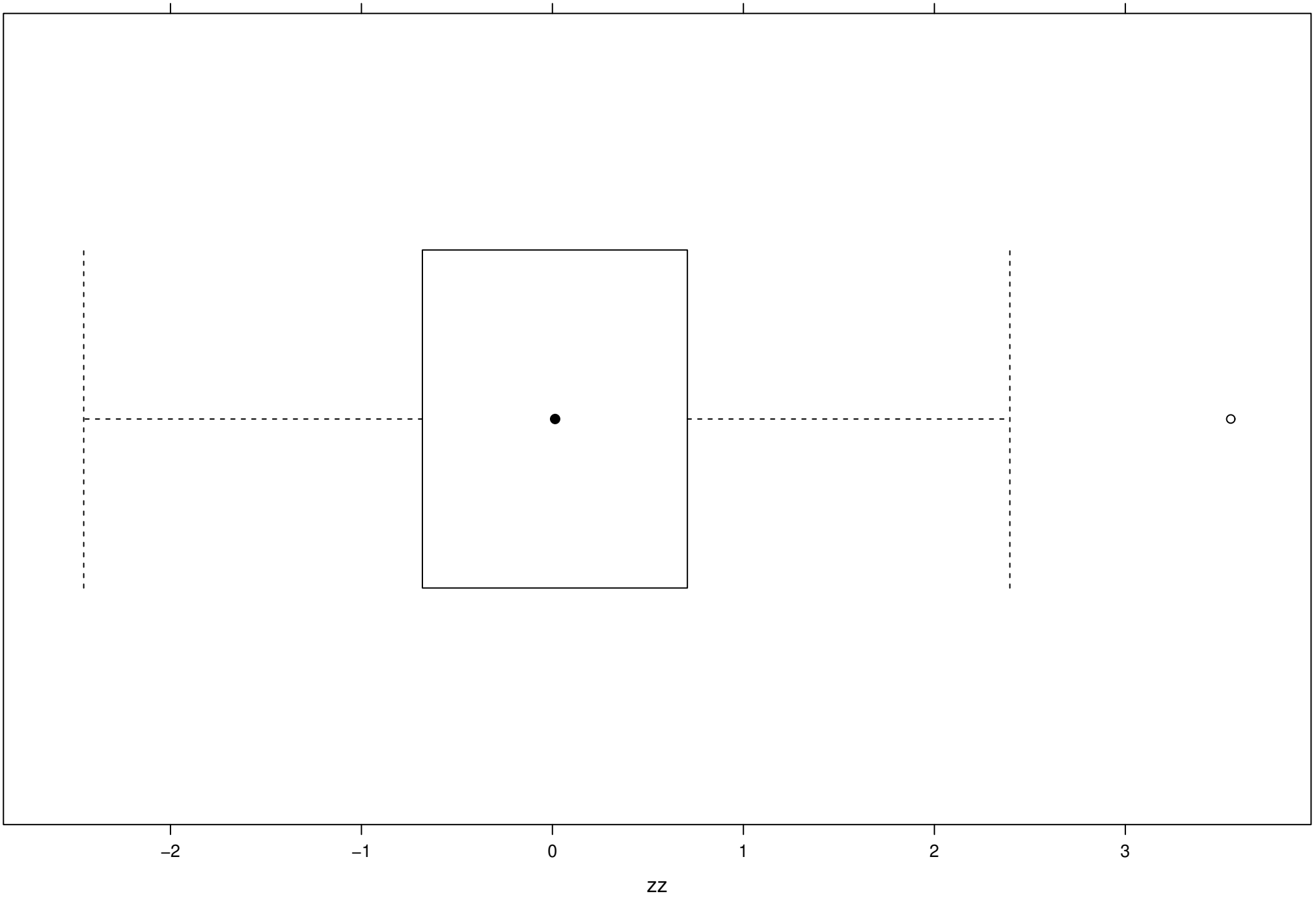
xyplot(a ~ x | z, main = "xyplot call with modified options", lattice.options = list(panel.xyplot = "panel.bwplot", default.args = list(between = list(x = 1, y = 1))))

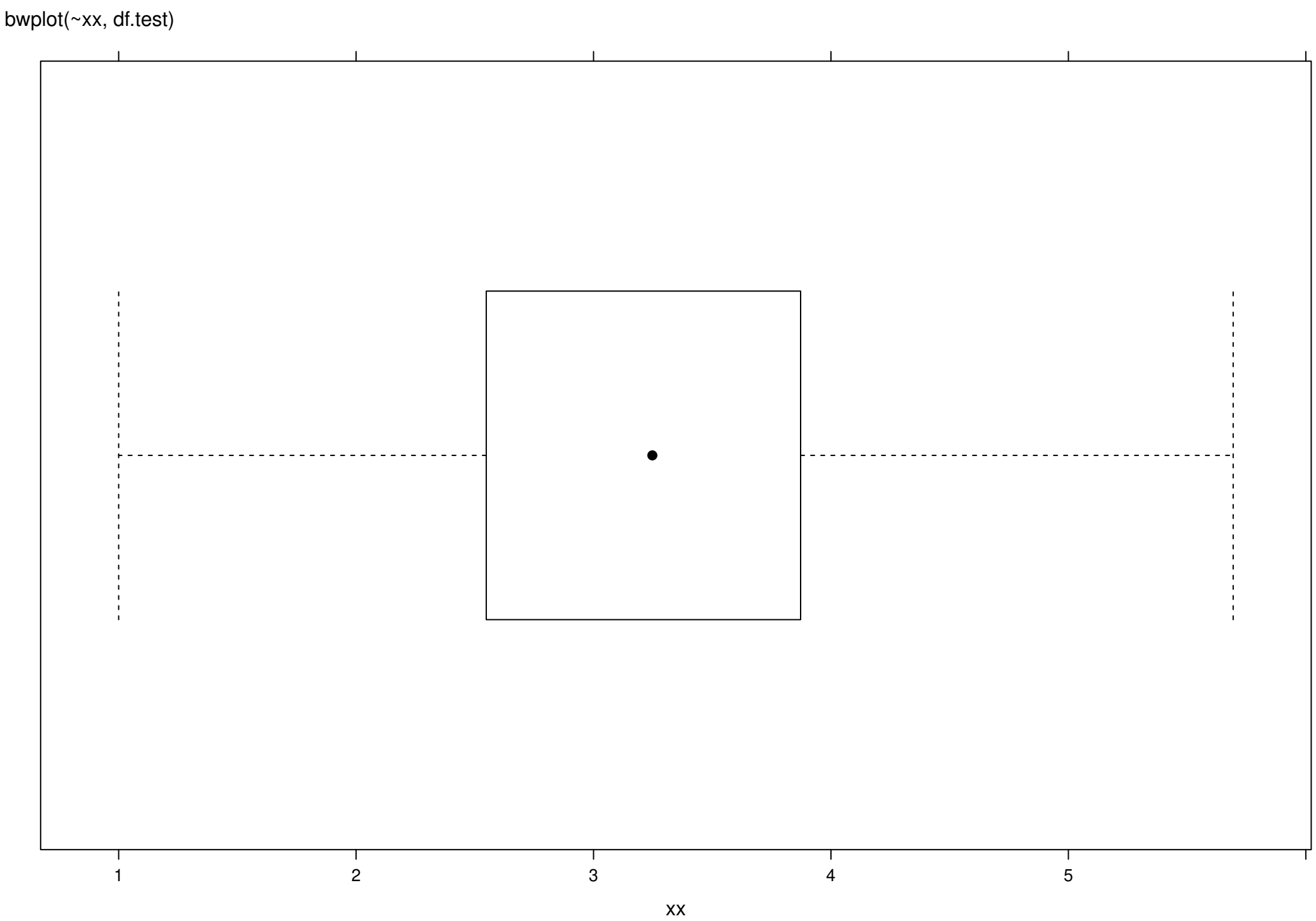


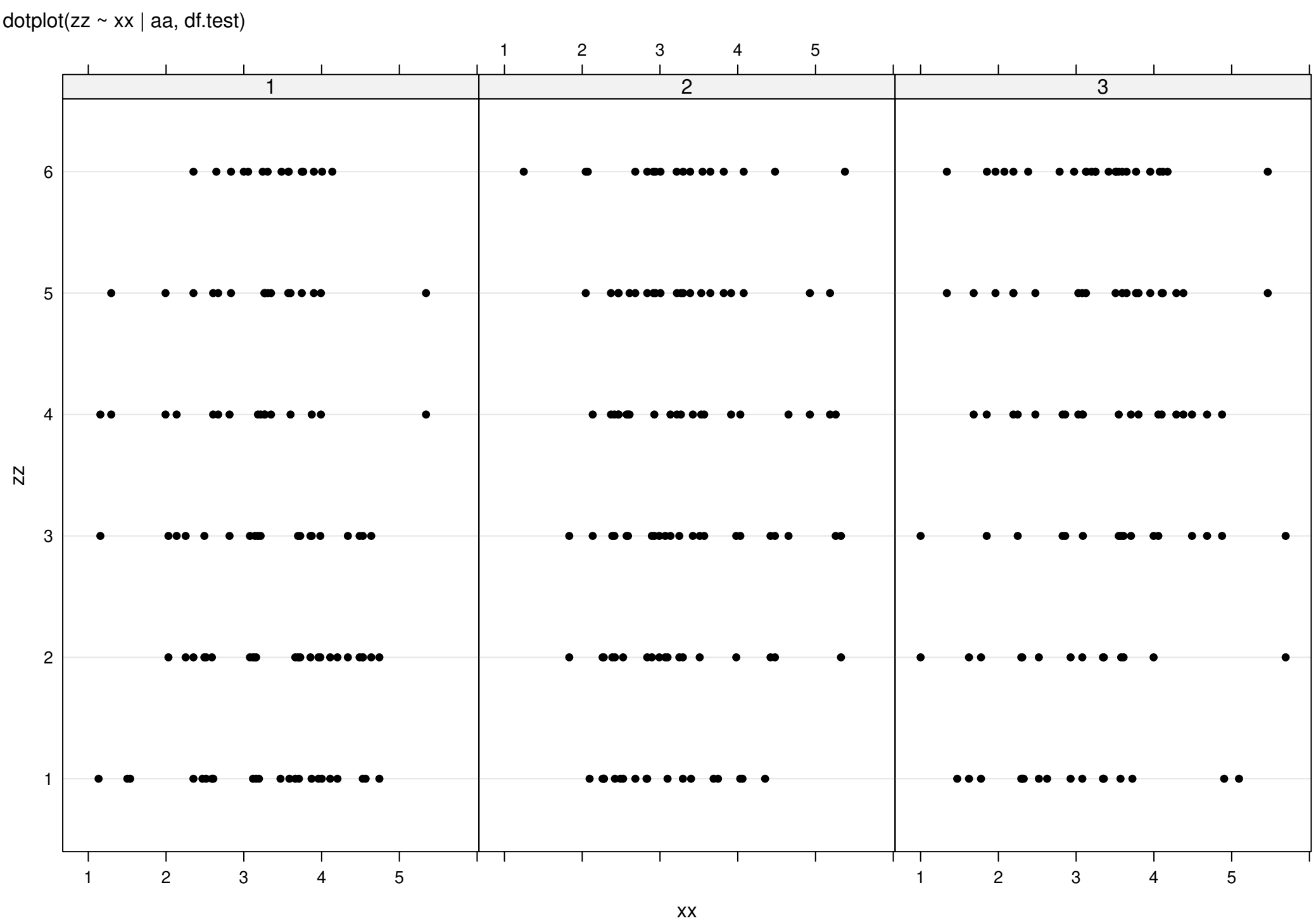

bwplot(zz ~ xx | aa, df.test)
1
2
3
zz
xx

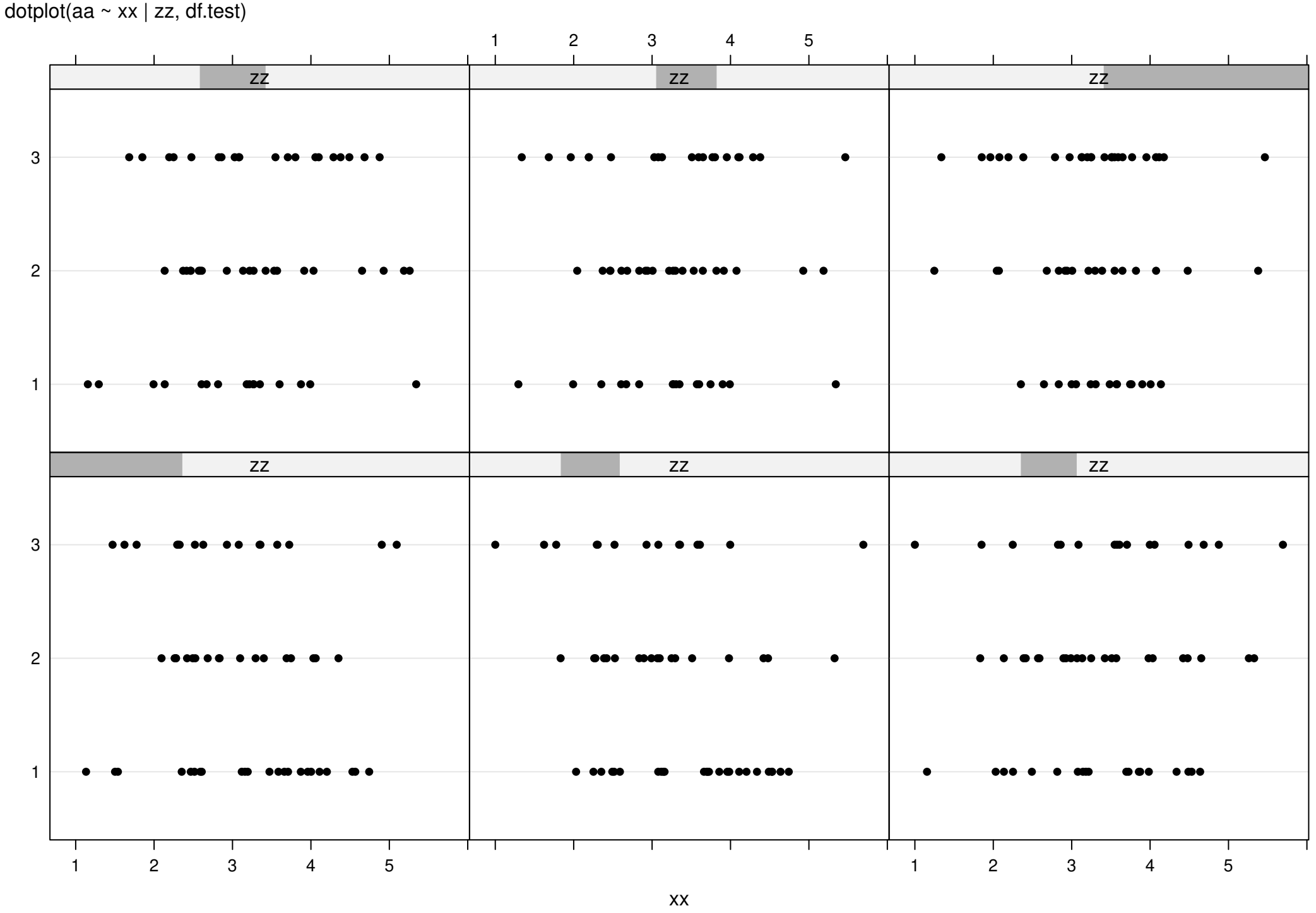

bwplot(aa ~ xx | zz, df.test, scales = list(x = list(log = "e", tck = 5, rot = 90, cex = 2), y = list(col = "red", tck = 3, alternating = TRUE, cex = 5,
rot = 0), tick.number = 20), main = list("main", cex = 5), sub = list("sub", cex = 5), xlab = list("xlab", cex = 5), ylab = list("ylab", cex = 5))
main
zz
ylab
xlab
sub

bwplot(~zz, df.test)

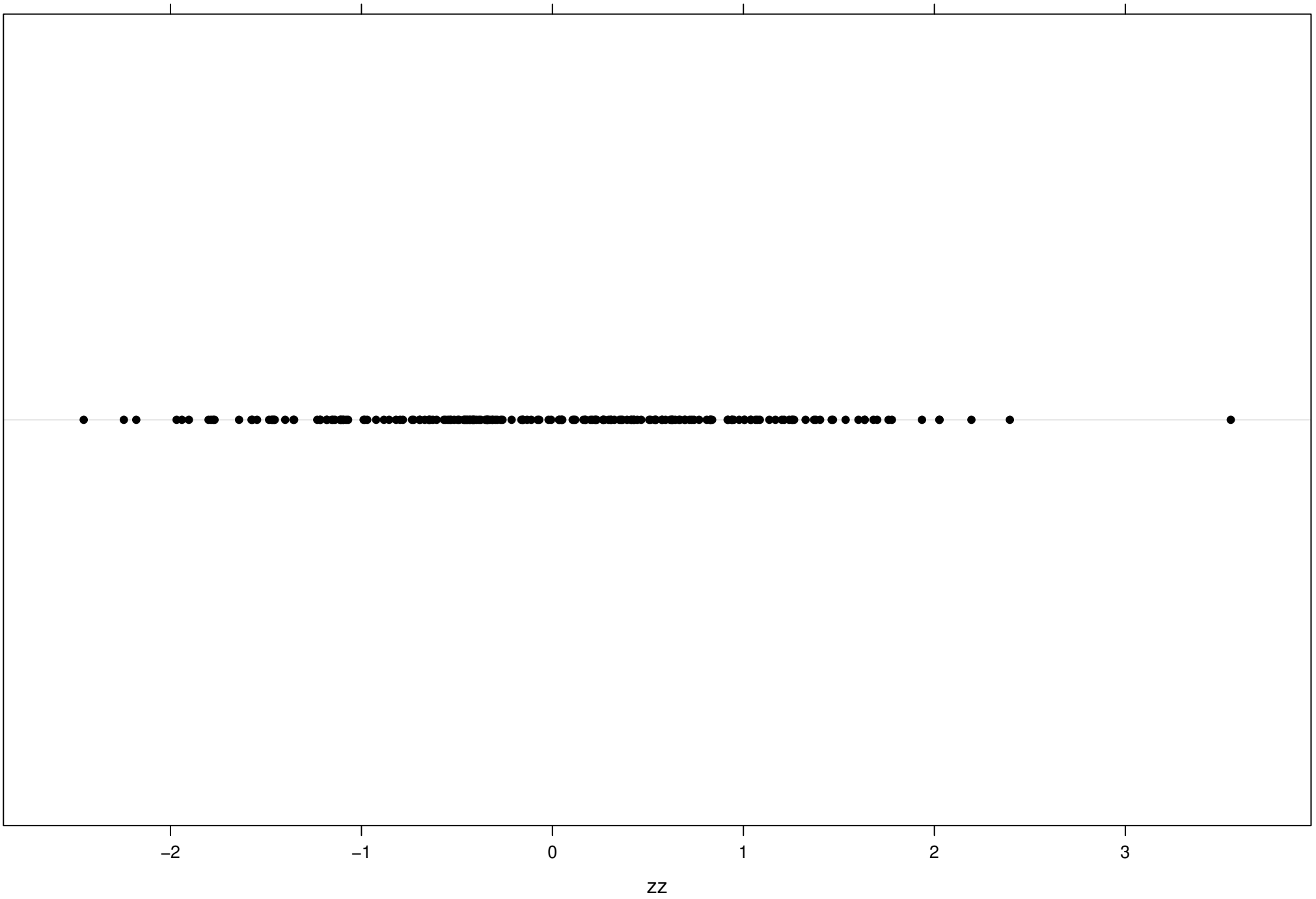



bwplot(~xx, df.test)
1
2
3
4
5
xx

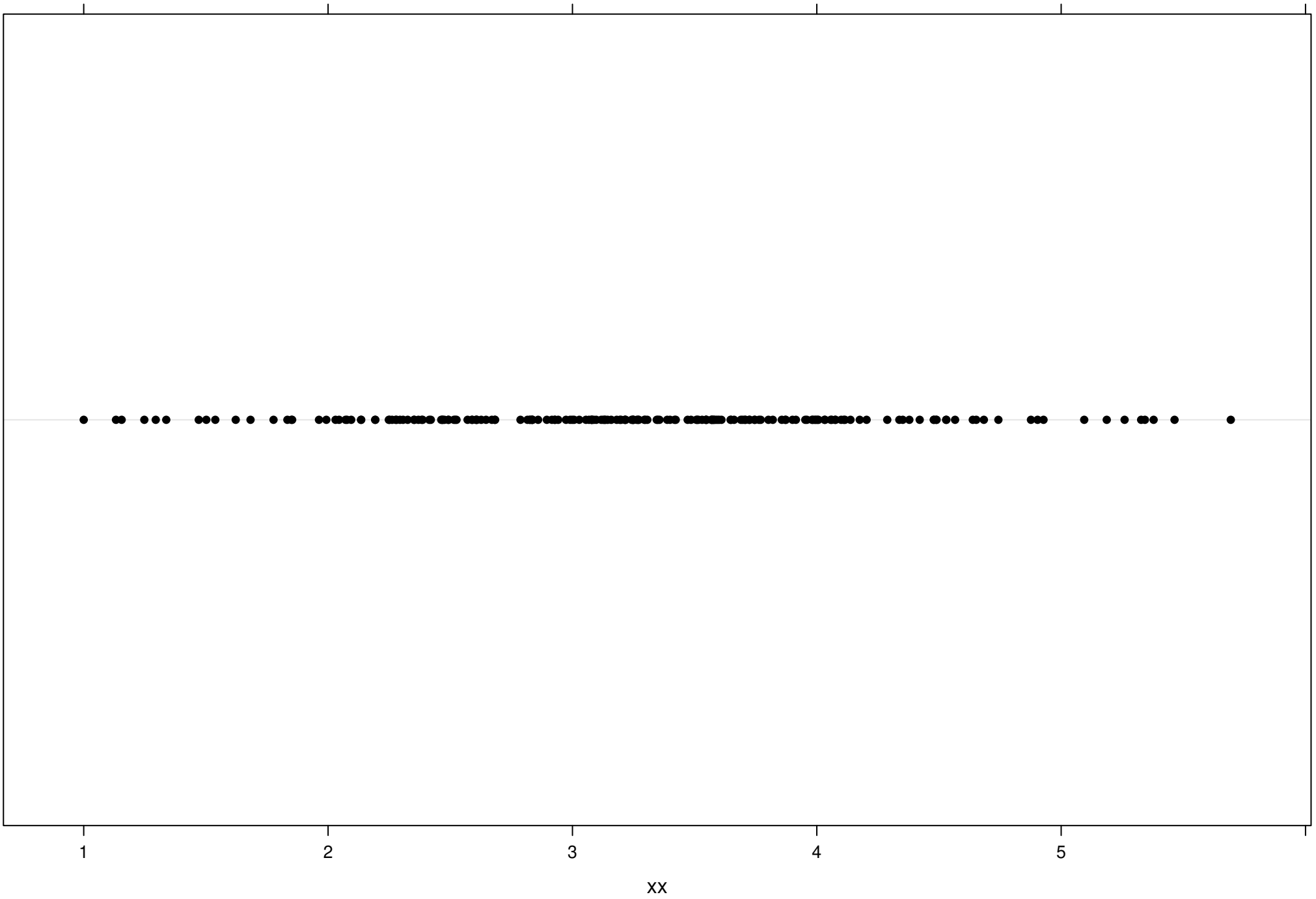

dotplot(zz ~ xx | aa, df.test)
1
2
3
zz
xx


dotplot(aa ~ xx | zz, df.test)
zz
zz
zz
zz
zz
zz
1
2
3
4
5
1
2
3
4
5
xx

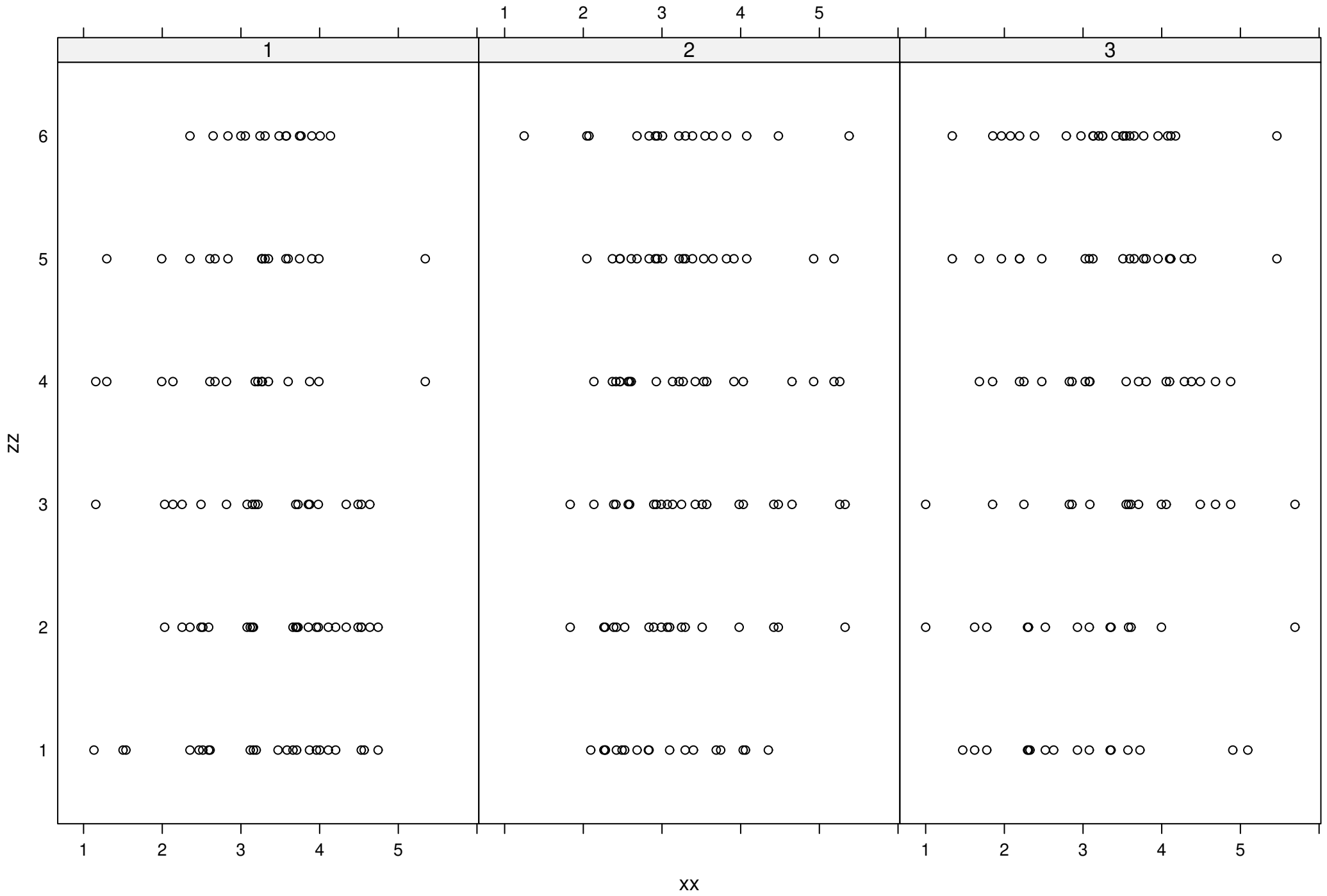
dotplot(~zz, df.test)

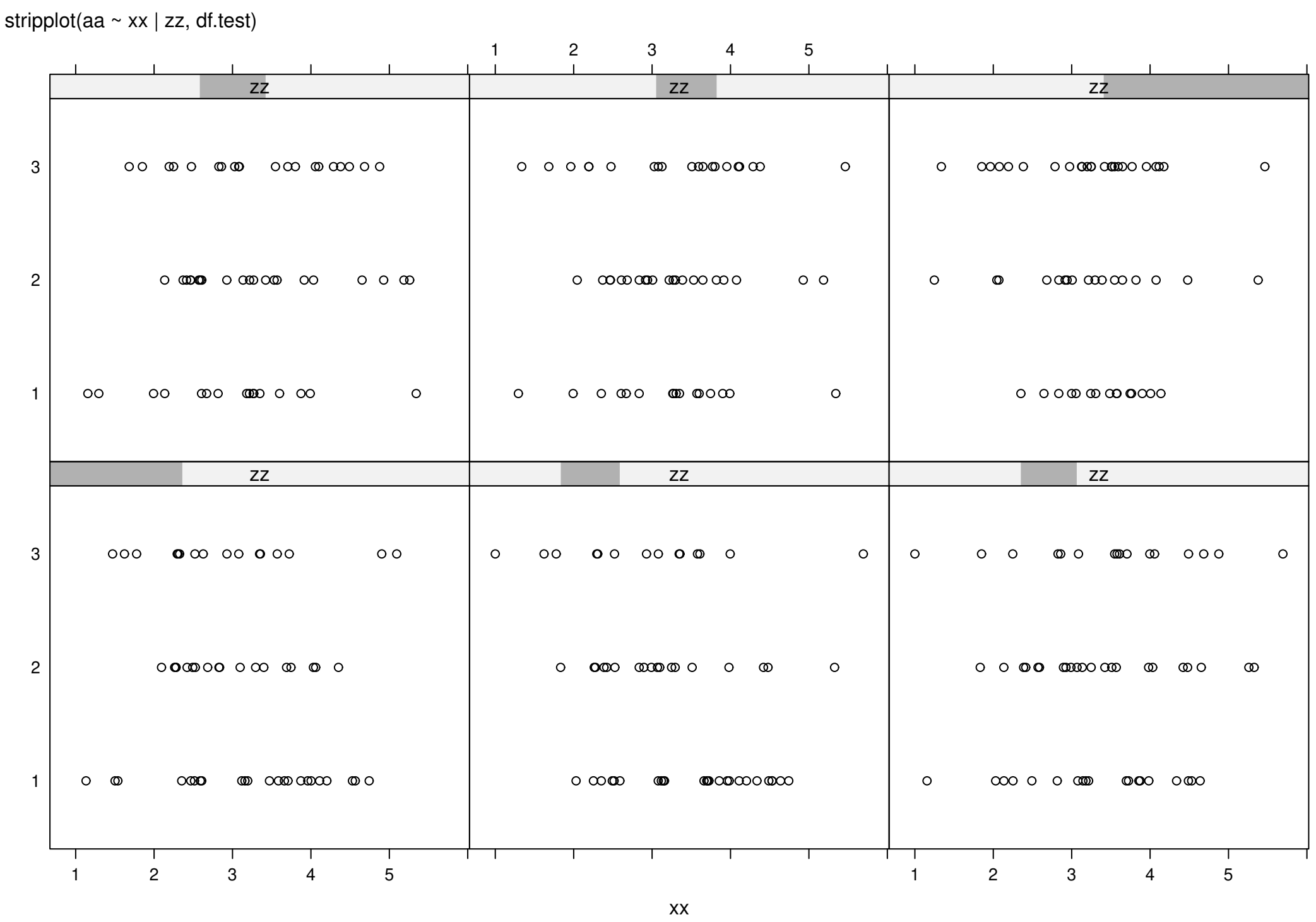


dotplot(~xx, df.test)

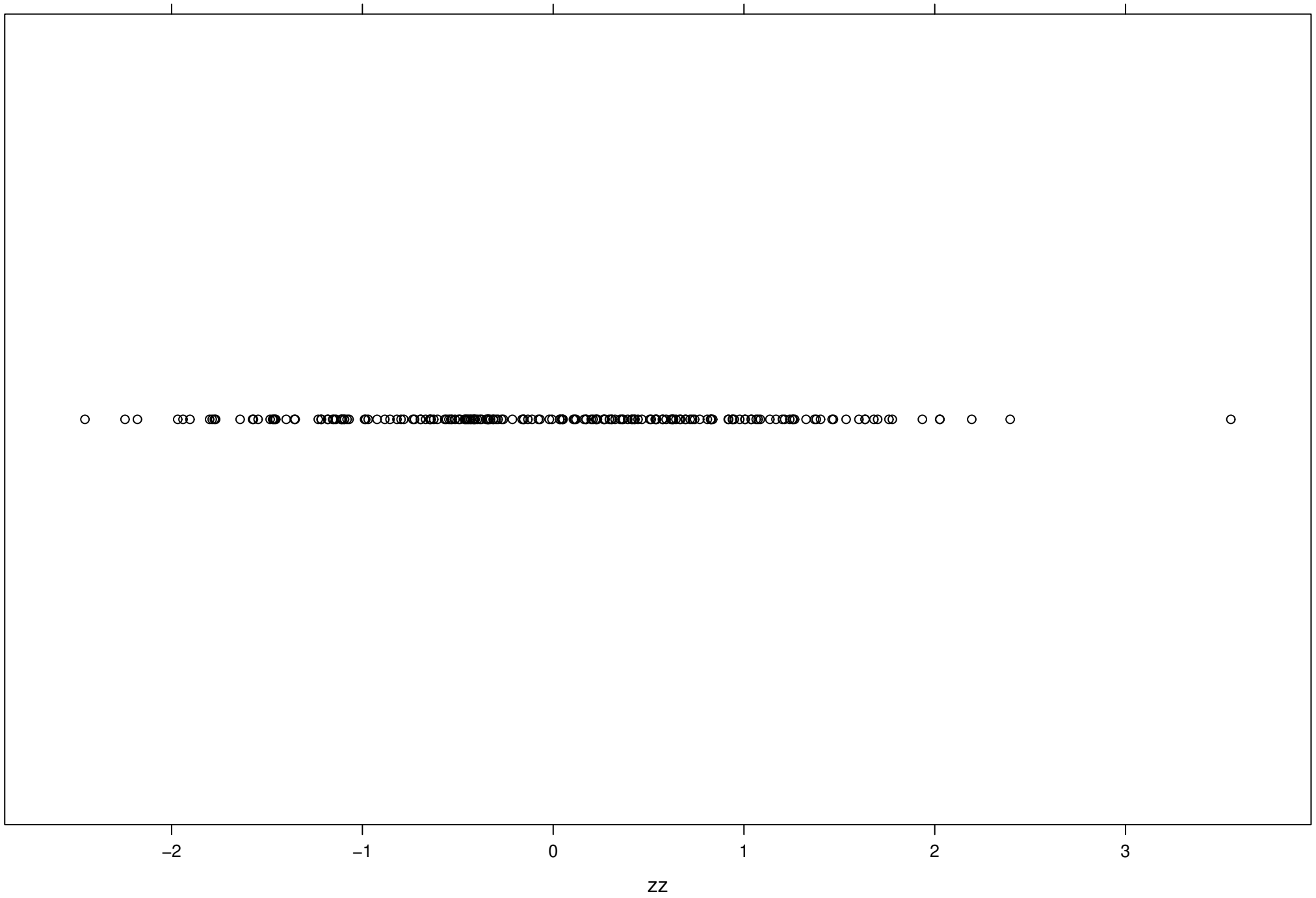


stripplot(zz ~ xx | aa, df.test)

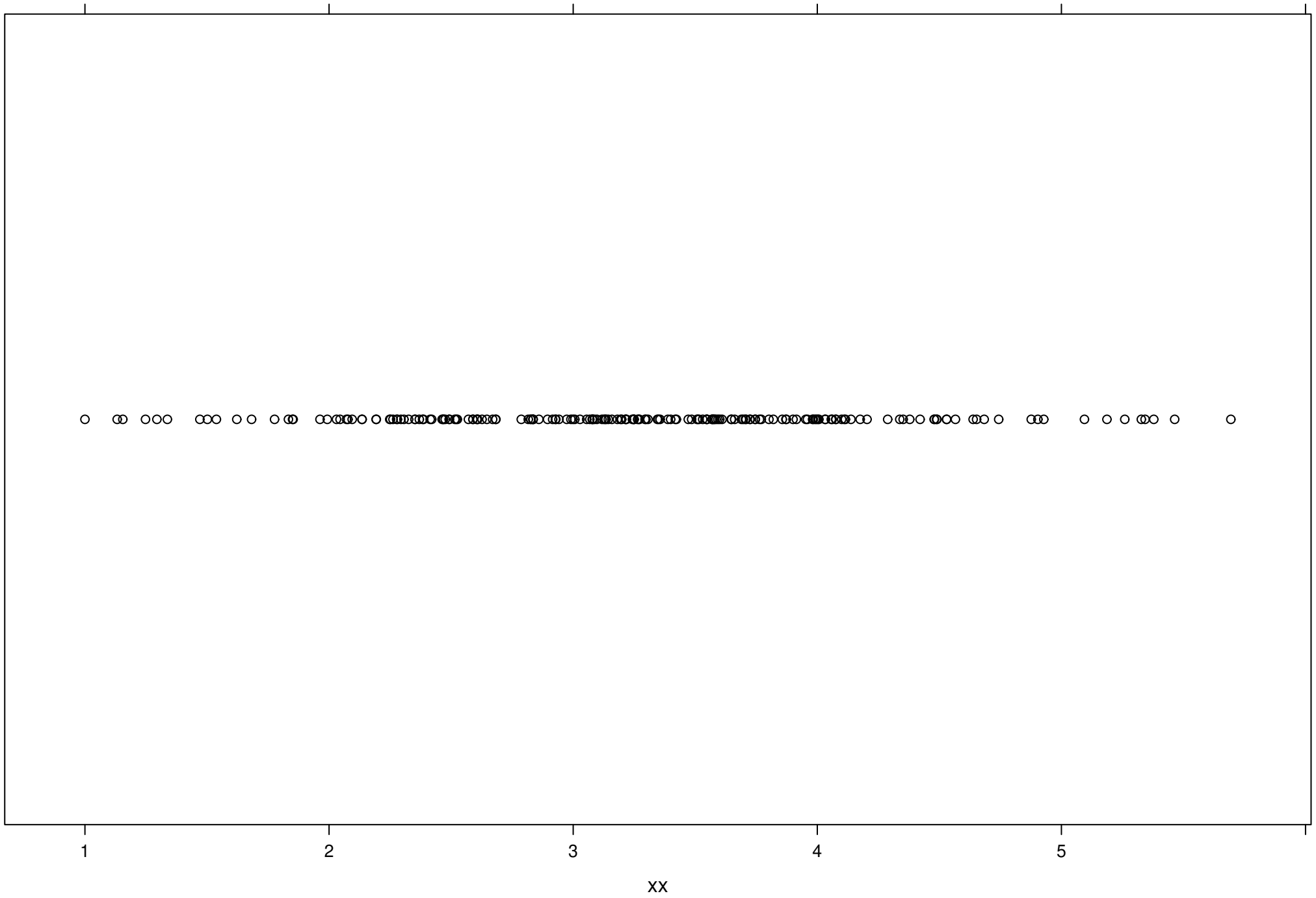


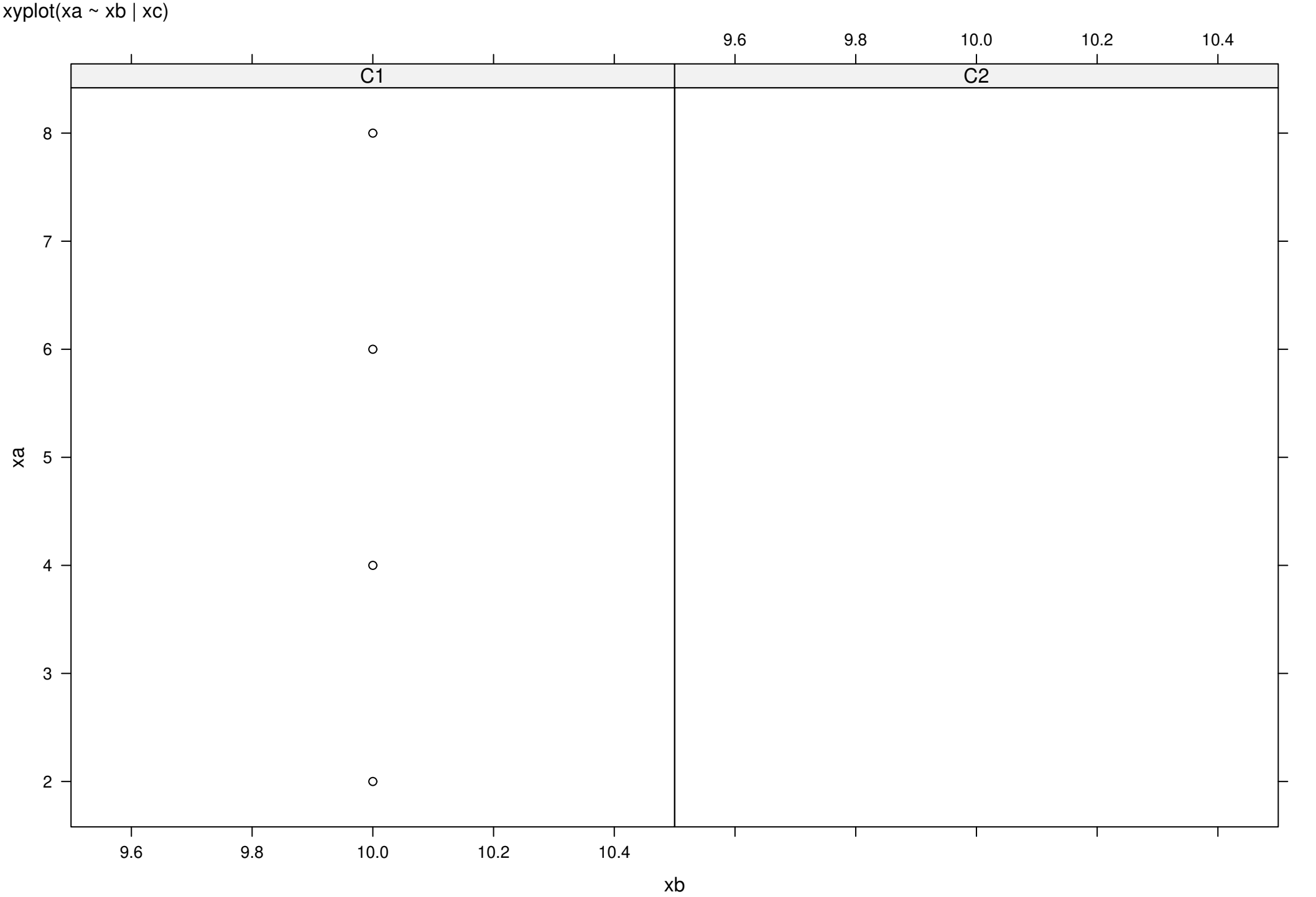

stripplot(aa ~ xx | zz, df.test)
zz
zz
zz
zz
zz
zz
1
2
3
4
5
xx

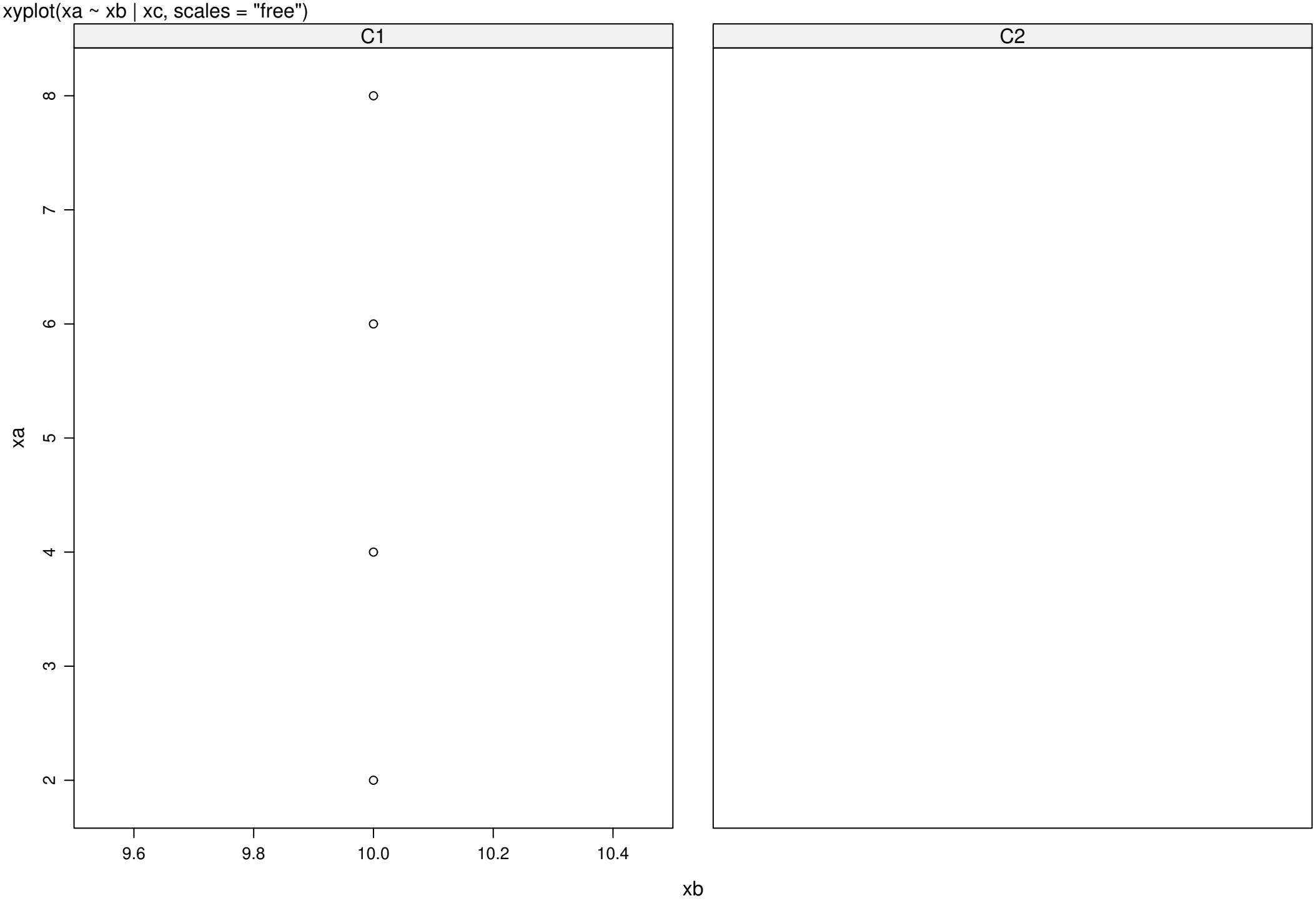
stripplot(~zz, df.test)

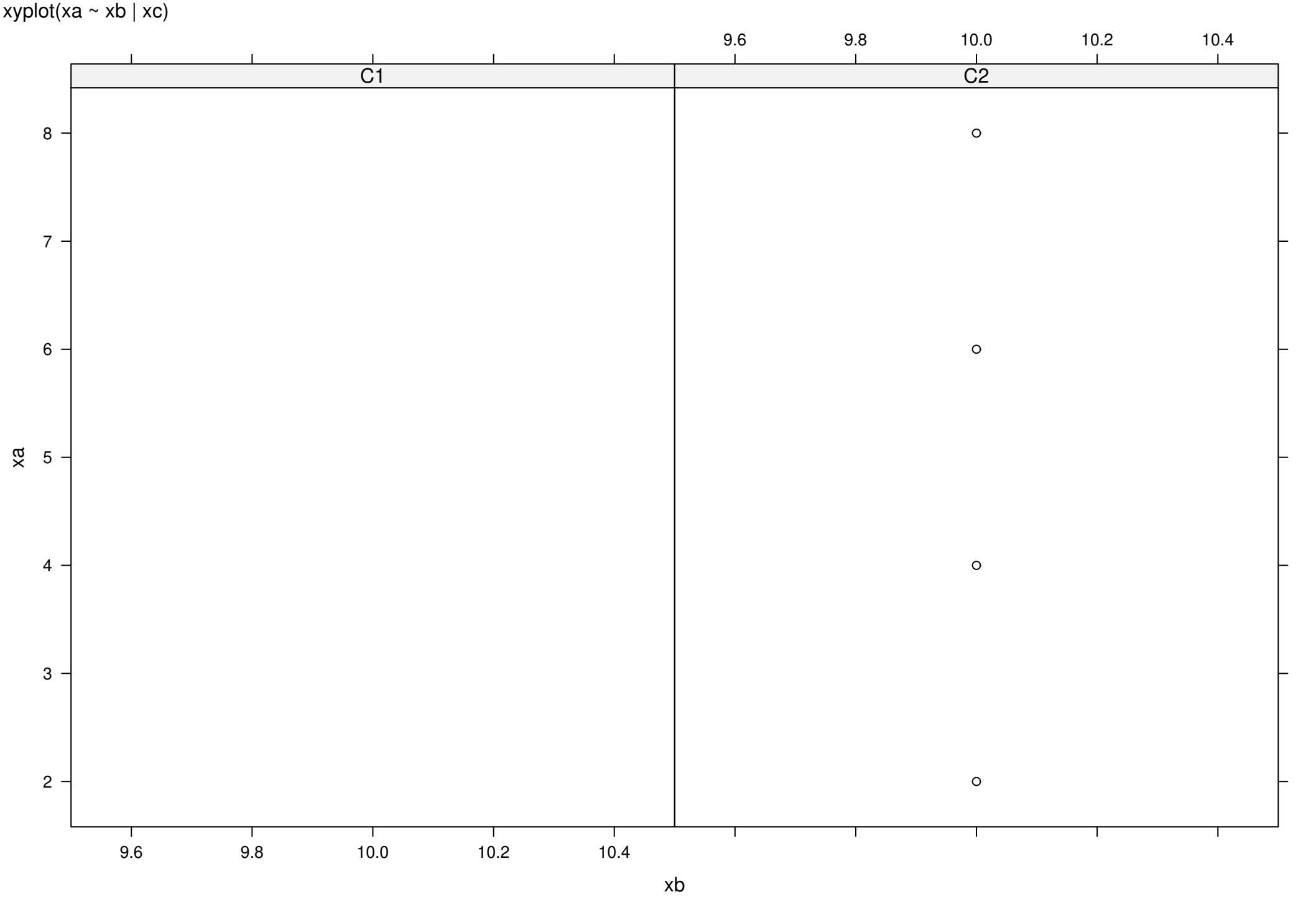


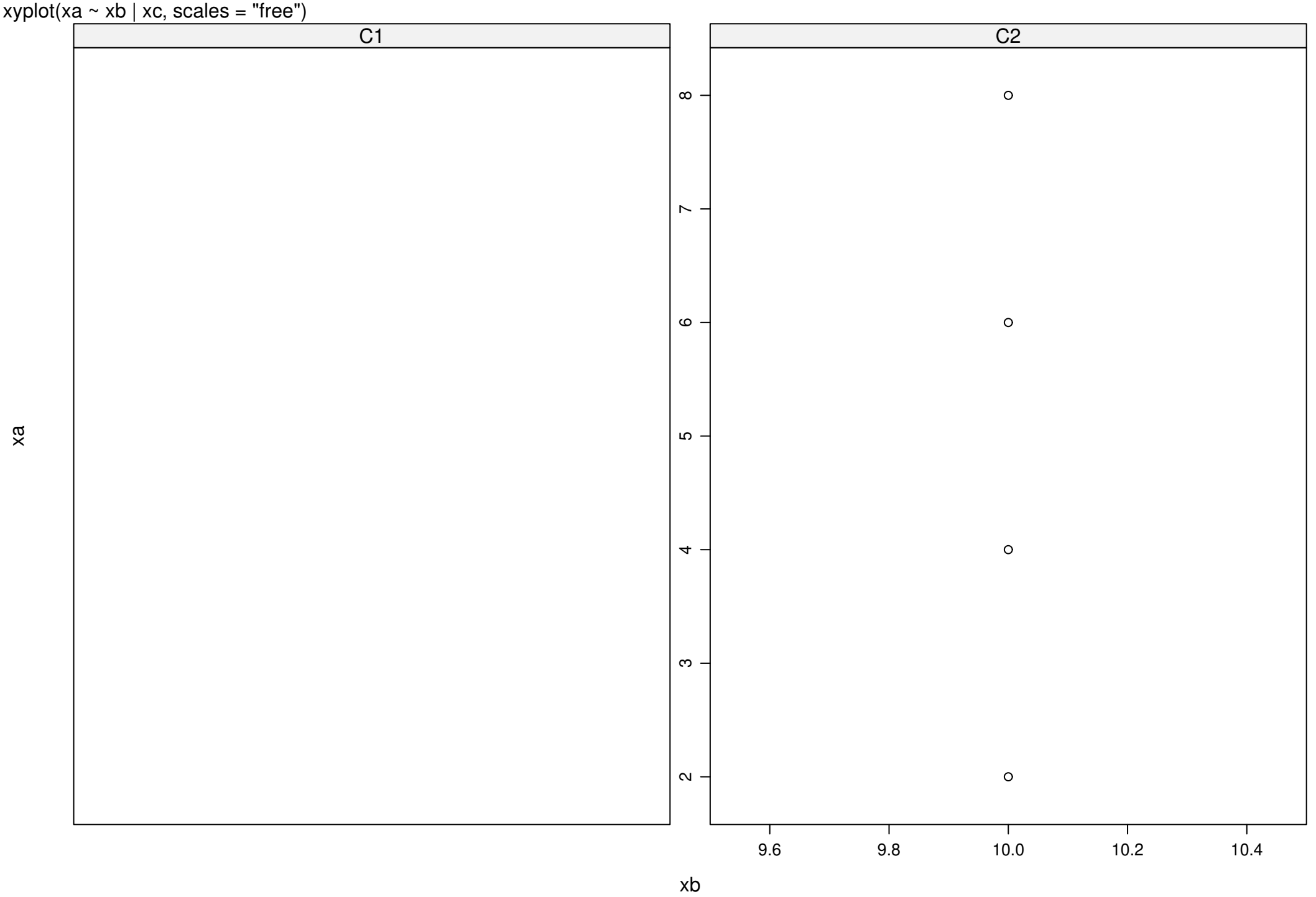
stripplot(~xx, df.test)

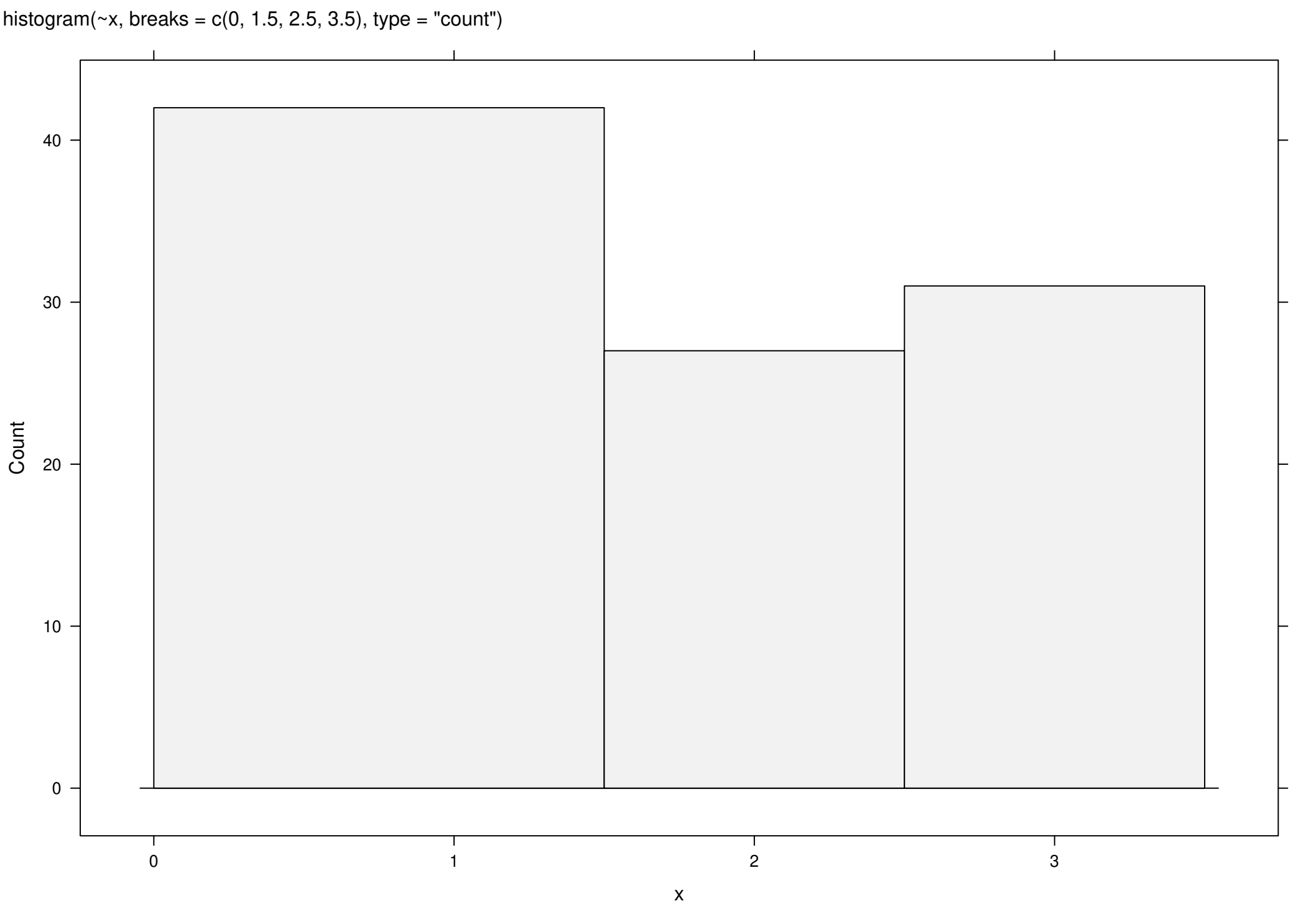


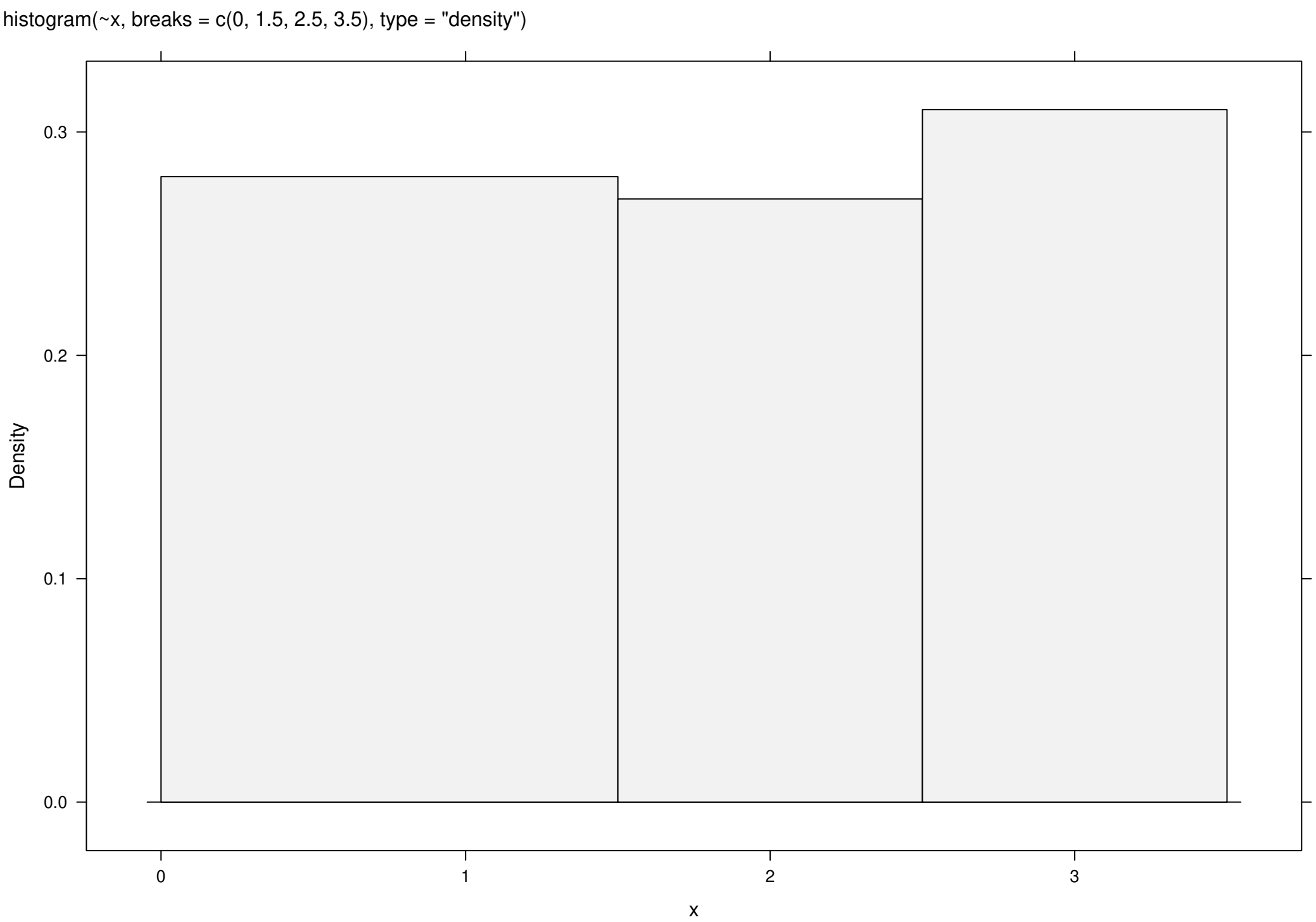

xyplot(xa ~ xb | xc)
C1
C2
xa
xb
2
3
4
5
6
7
8
9.6
9.8
10.0
10.2
10.4


xyplot(xa ~ xb | xc, scales = "free")
C1
C2
xa
xb
2
3
4
5
6
7
8
9.6
9.8
10.0
10.2
10.4

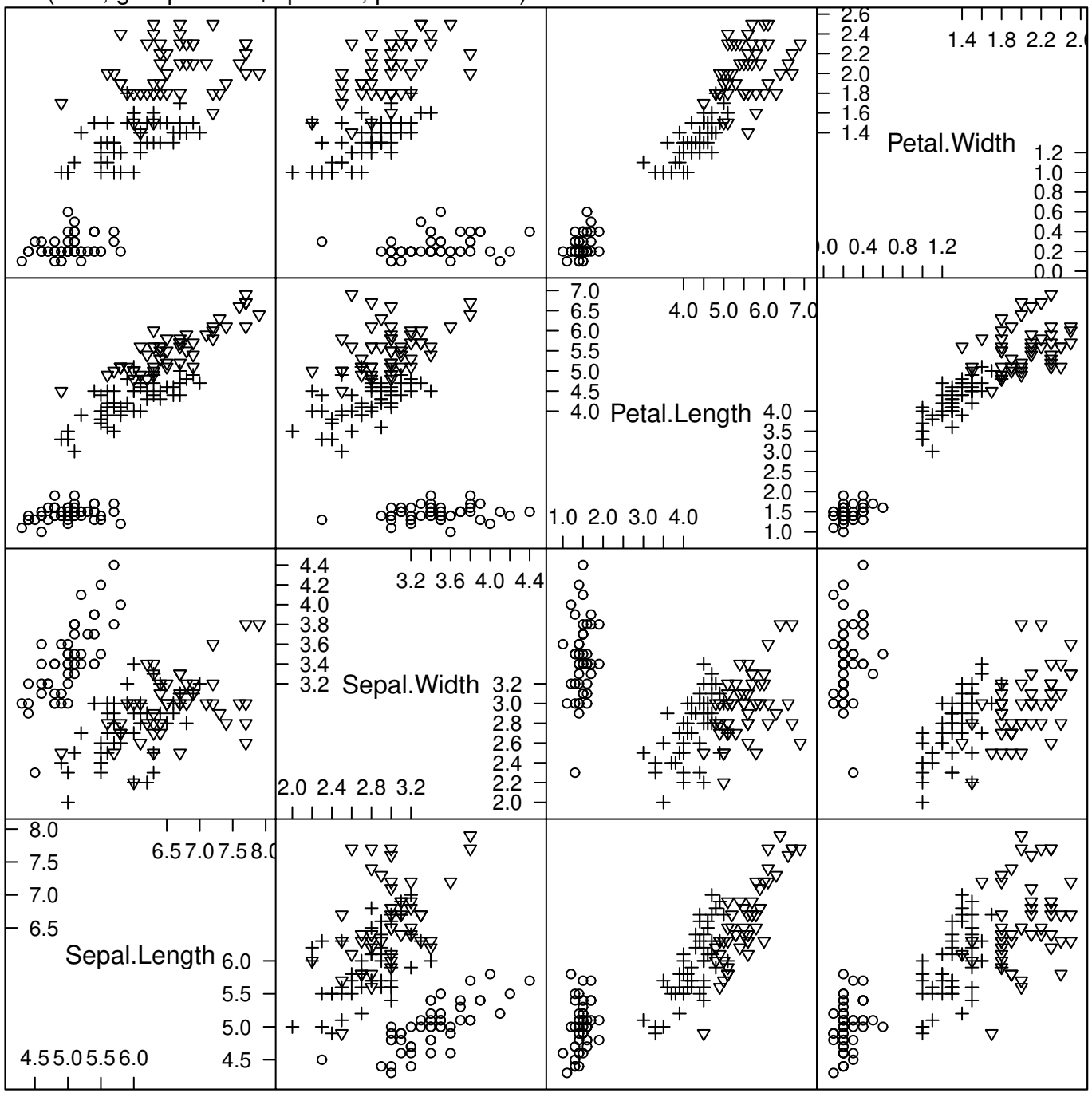

xyplot(xa ~ xb | xc)
C1
C2
9.6
9.8
10.0
10.2
10.4
2
3
4
5
6
7
8
xa
xb


xyplot(xa ~ xb | xc, scales = "free")
C1
C2
xa
xb
2
3
4
5
6
7
8
9.6
9.8
10.0
10.2
10.4


histogram(~x, breaks = c(0, 1.5, 2.5, 3.5), type = "count")
Count
0
10
20
30
40
0
1
2
3
x

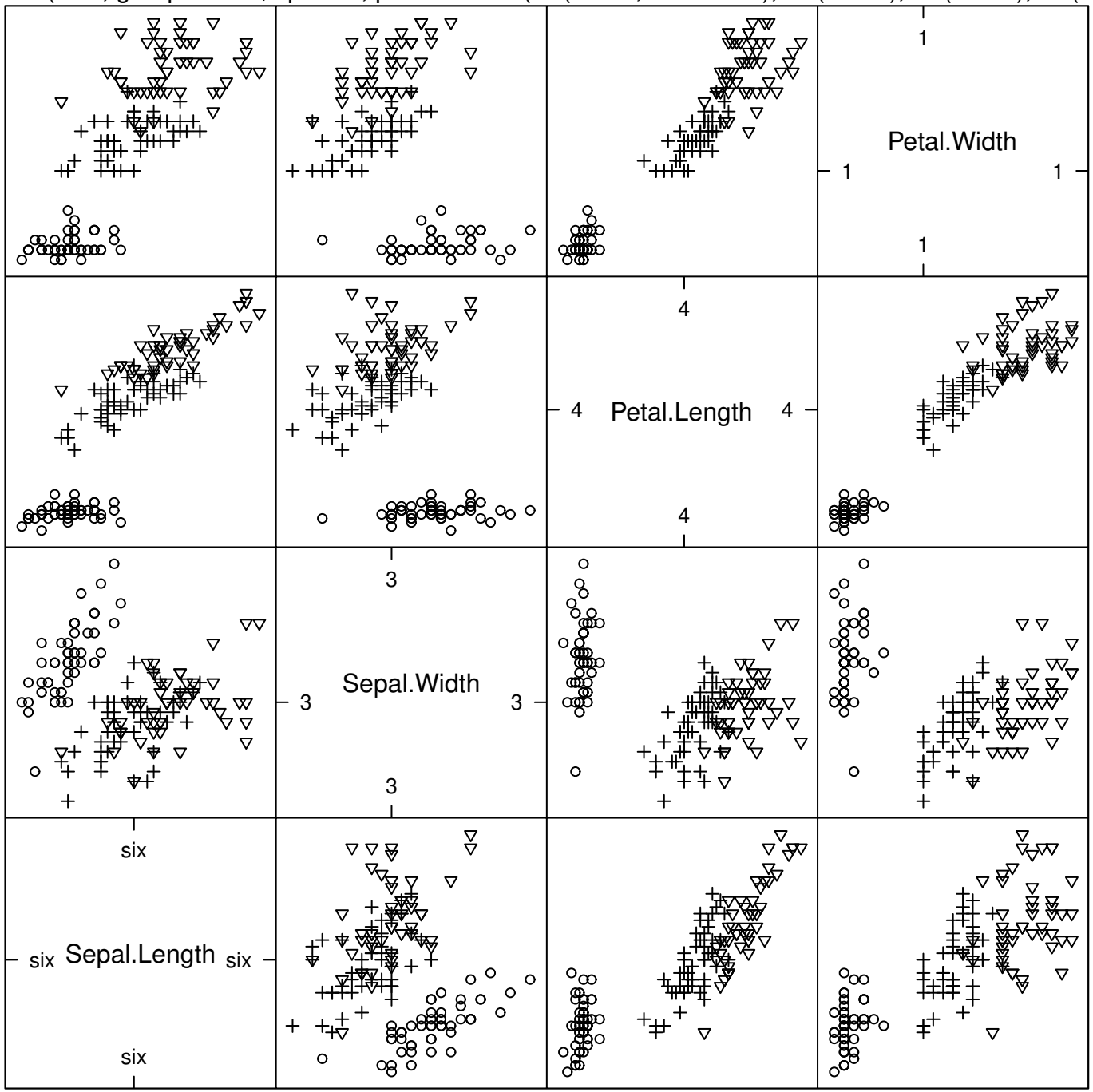

histogram(~x, breaks = c(0, 1.5, 2.5, 3.5), type = "density")
0.3
0.2
0.1
0.0
Density
0
1
2
3
x

splom(iris2, groups = iris$Species, pscales = 10)

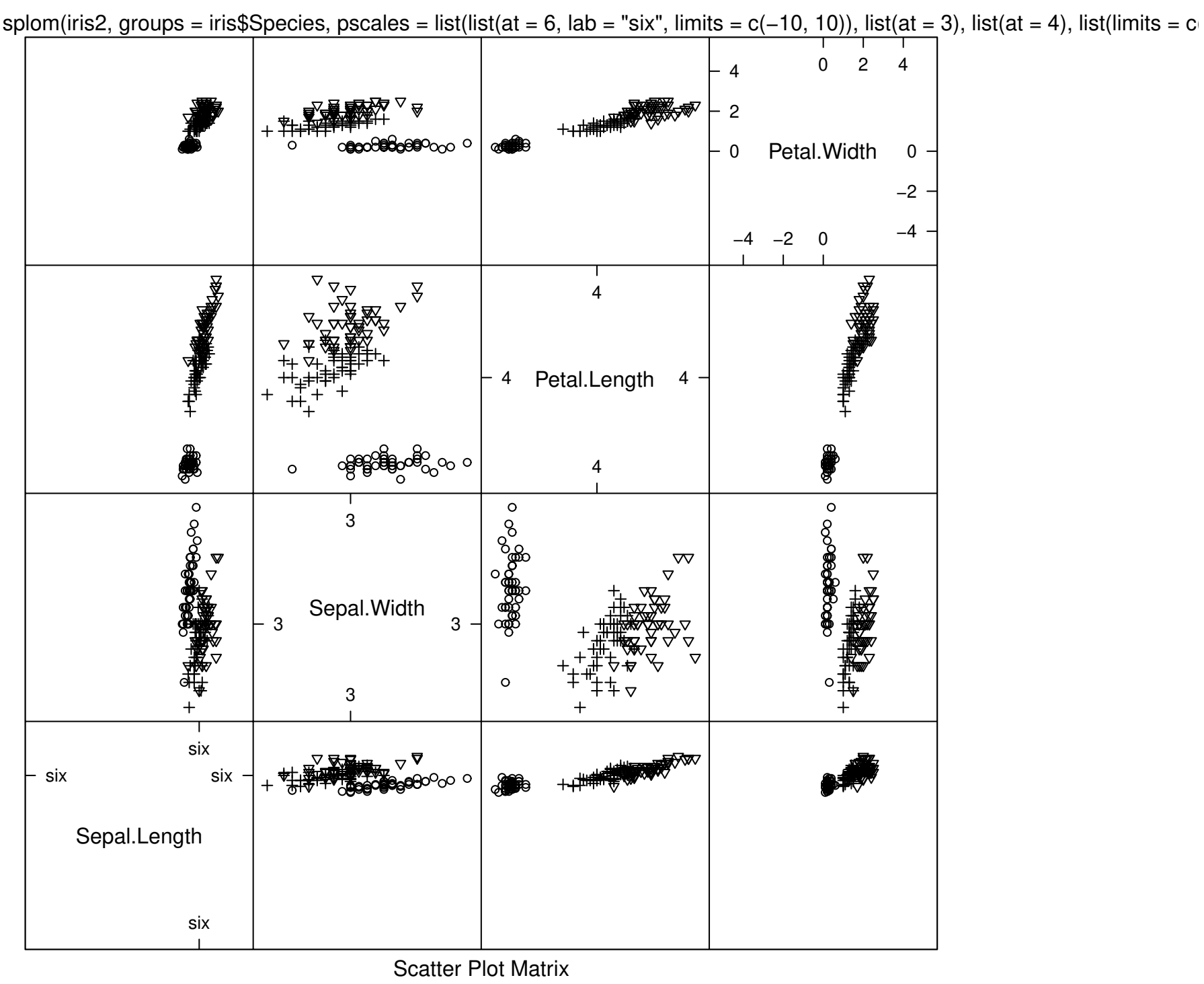


Scatter Plot Matrix

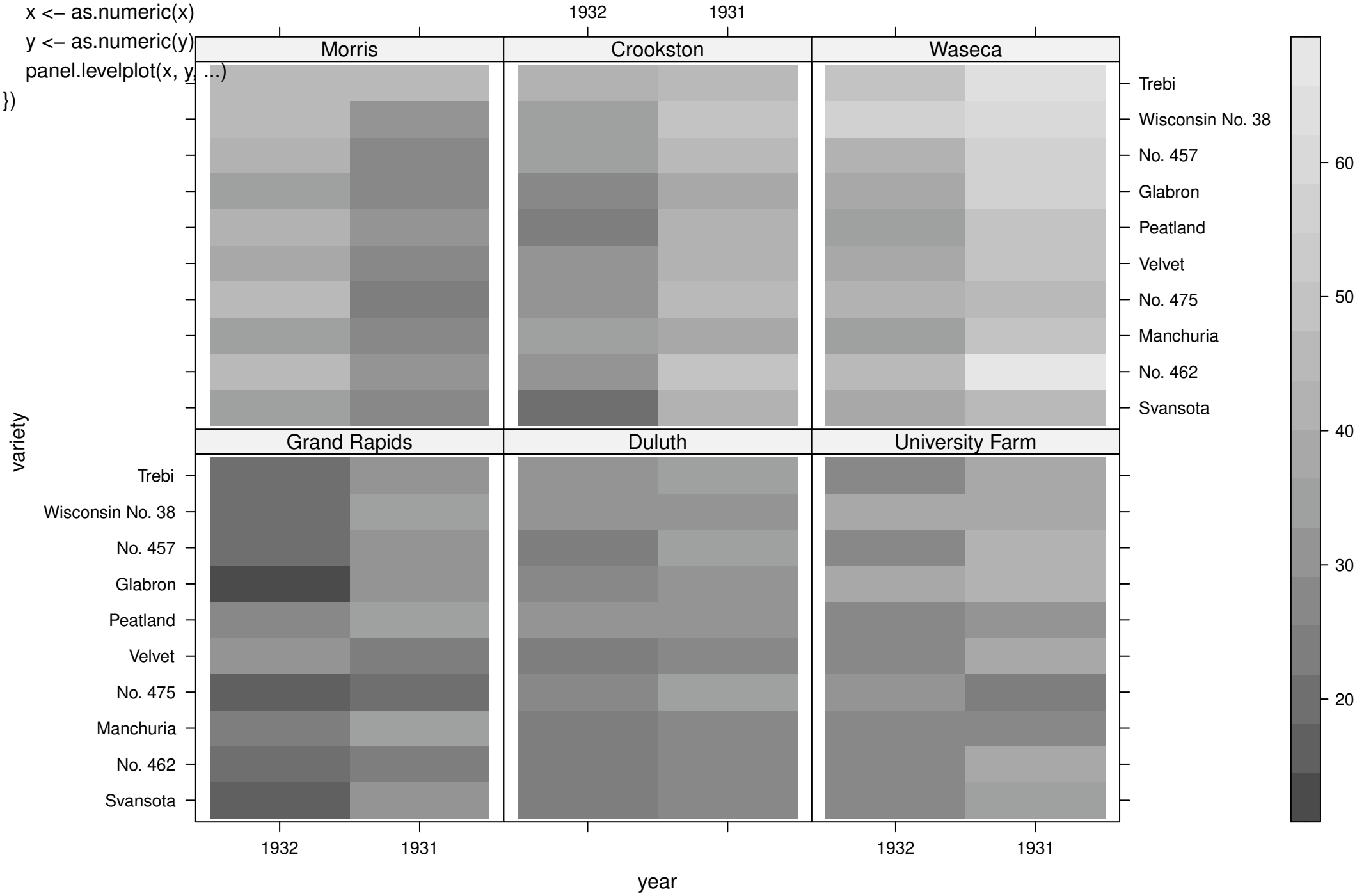
splom(iris2, groups = iris$Species, pscales = list(list(at = 6, lab = "six"), list(at = 3), list(at = 4), list(at = 1)))

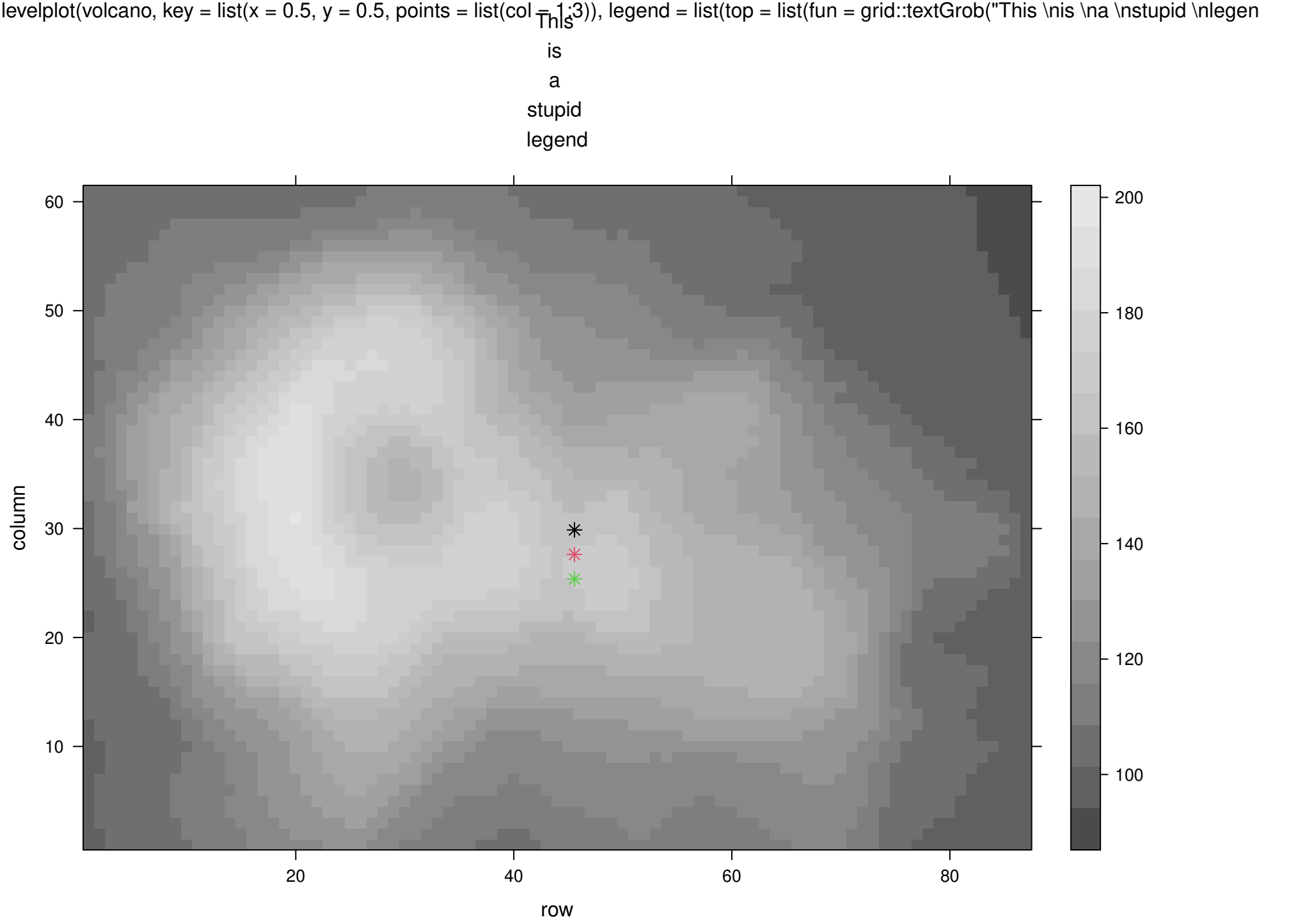


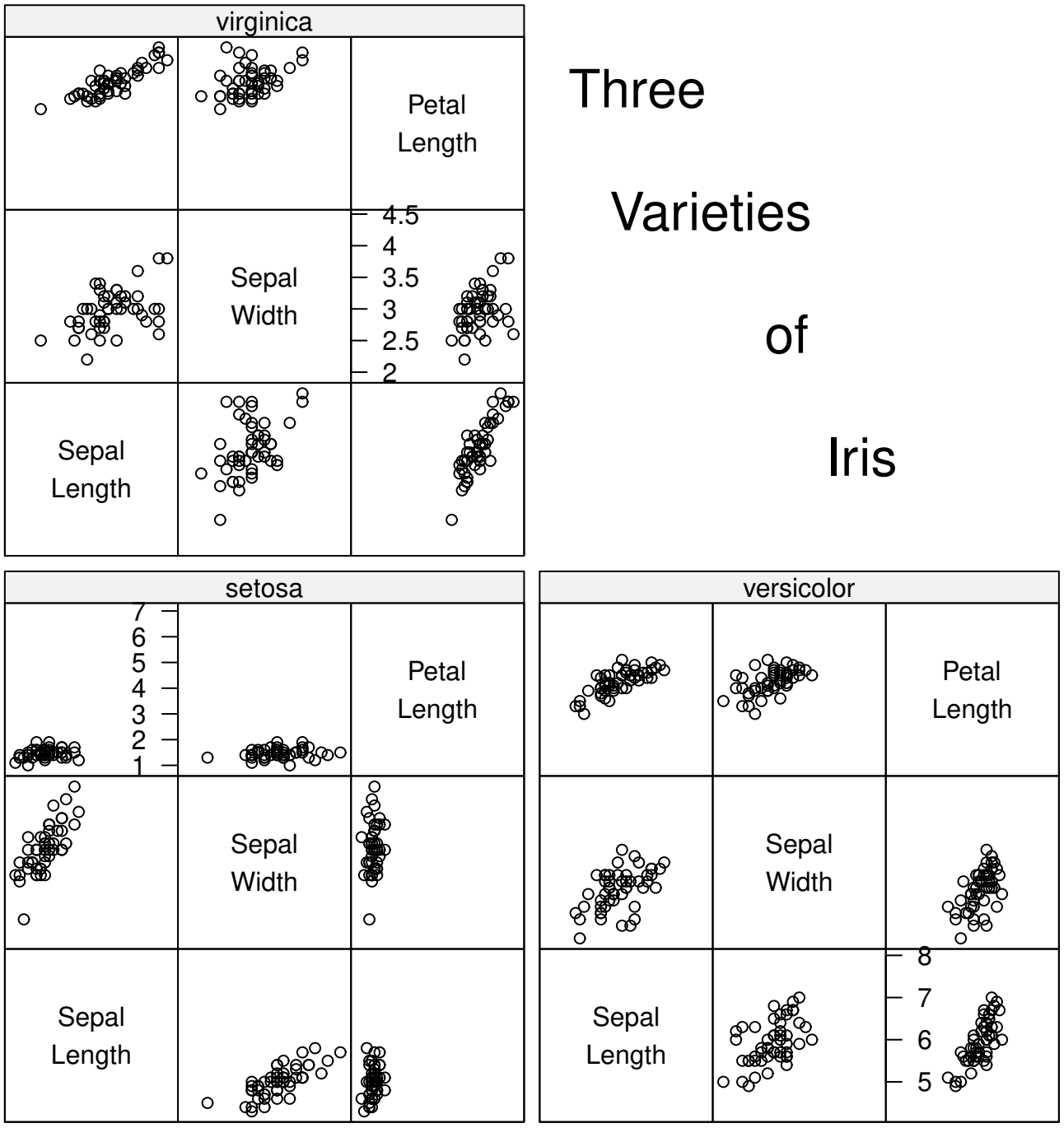
Scatter Plot Matrix


splom(iris2, groups = iris$Species, pscales = list(list(at = 6, lab = "six", limits = c(−10, 10)), list(at = 3), list(at = 4), list(limits = c
Petal.Width
Petal.Length
Sepal.Width
Sepal.Length
six
Scatter Plot Matrix

```
levelplot(yield ~ year * variety | site, barley, panel = function(x, y, ...) {
    x <- as.numeric(x)
    y <- as.numeric(y)
    panel.levelplot(x, y, ...)
})
```

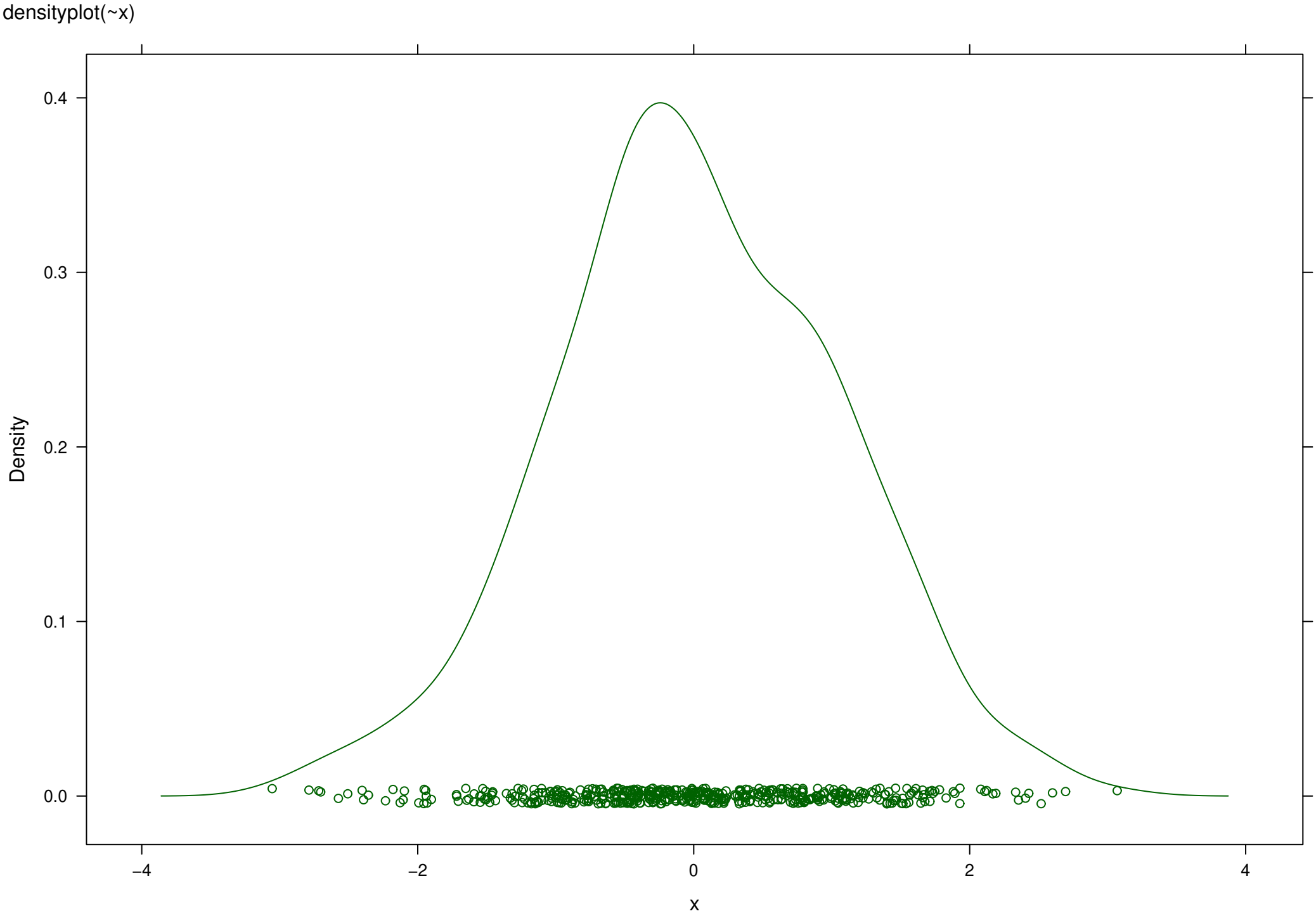



levelplot(volcano, key = list(x = 0.5, y = 0.5, points = list(col = 1:3)), legend = list(top = list(fun = grid::textGrob("This \nis \na \nstupid \nlegen
This
is
a
stupid
legend
60
50
40
30
20
10
column
20
40
60
80
row
200
180
160
140
120
100

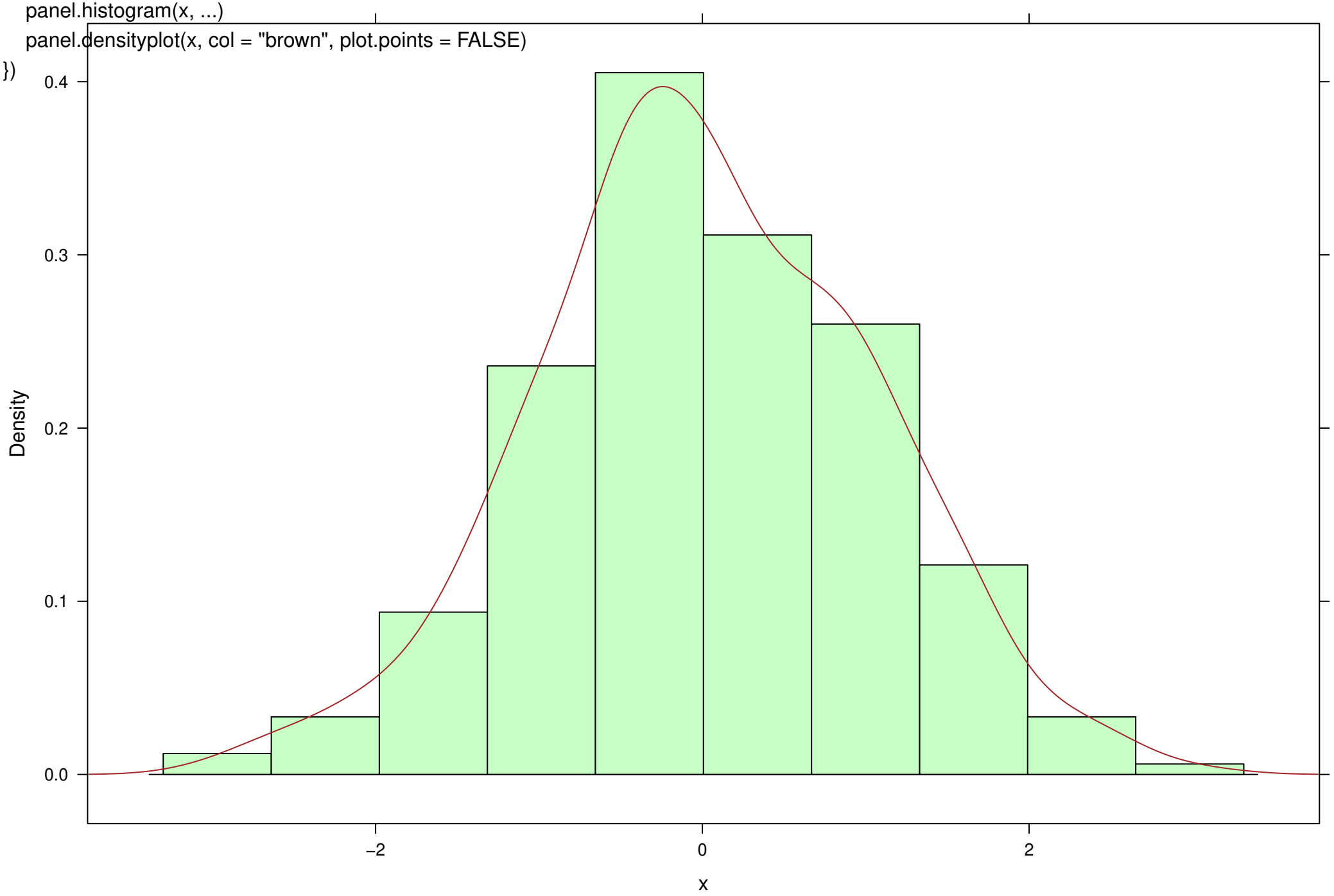


Three Varieties of Iris

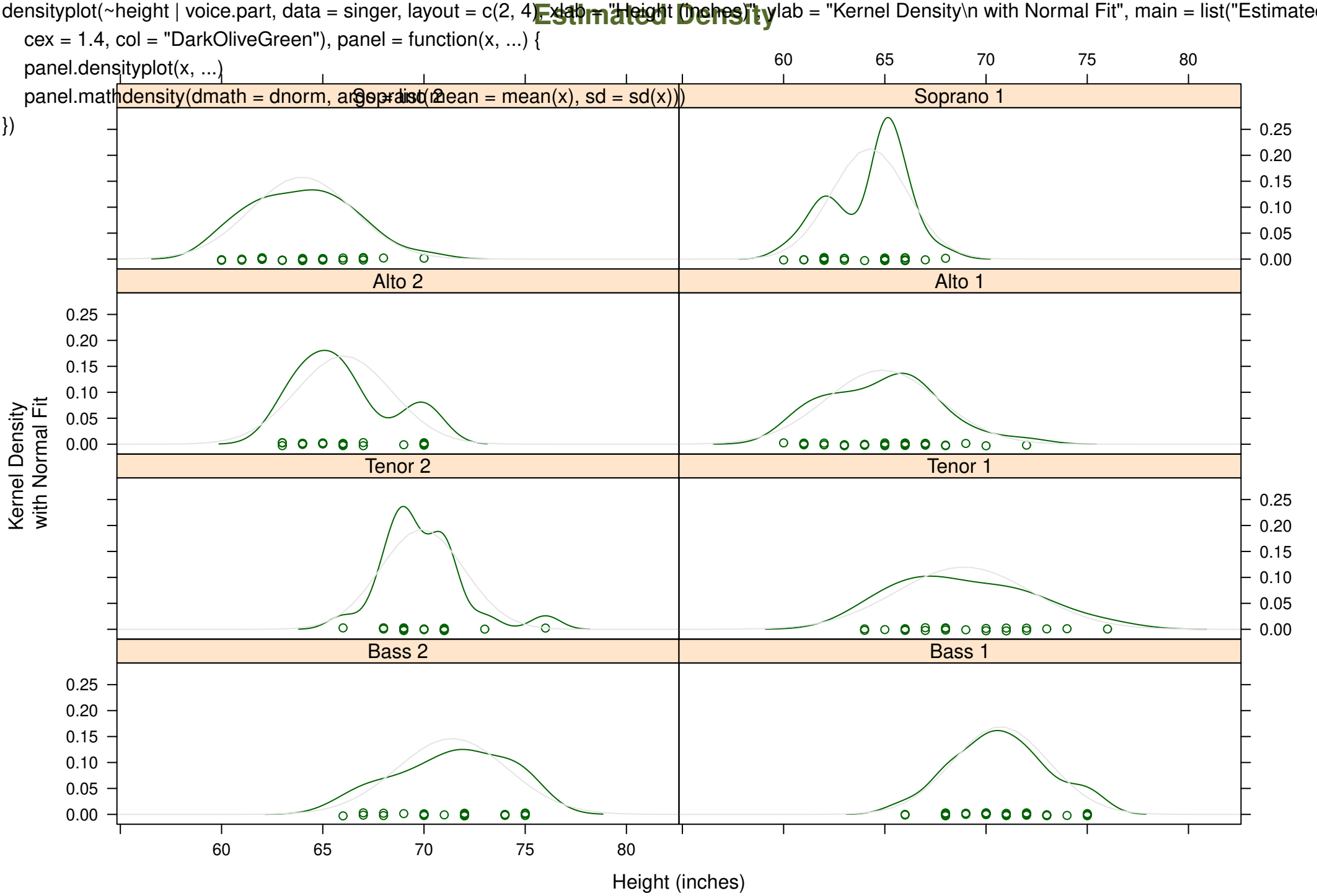
Scatter Plot Matrix


densityplot(~x)
0.4
0.3
0.2
0.1
0.0
Density
−4
−2
0
2
4
x

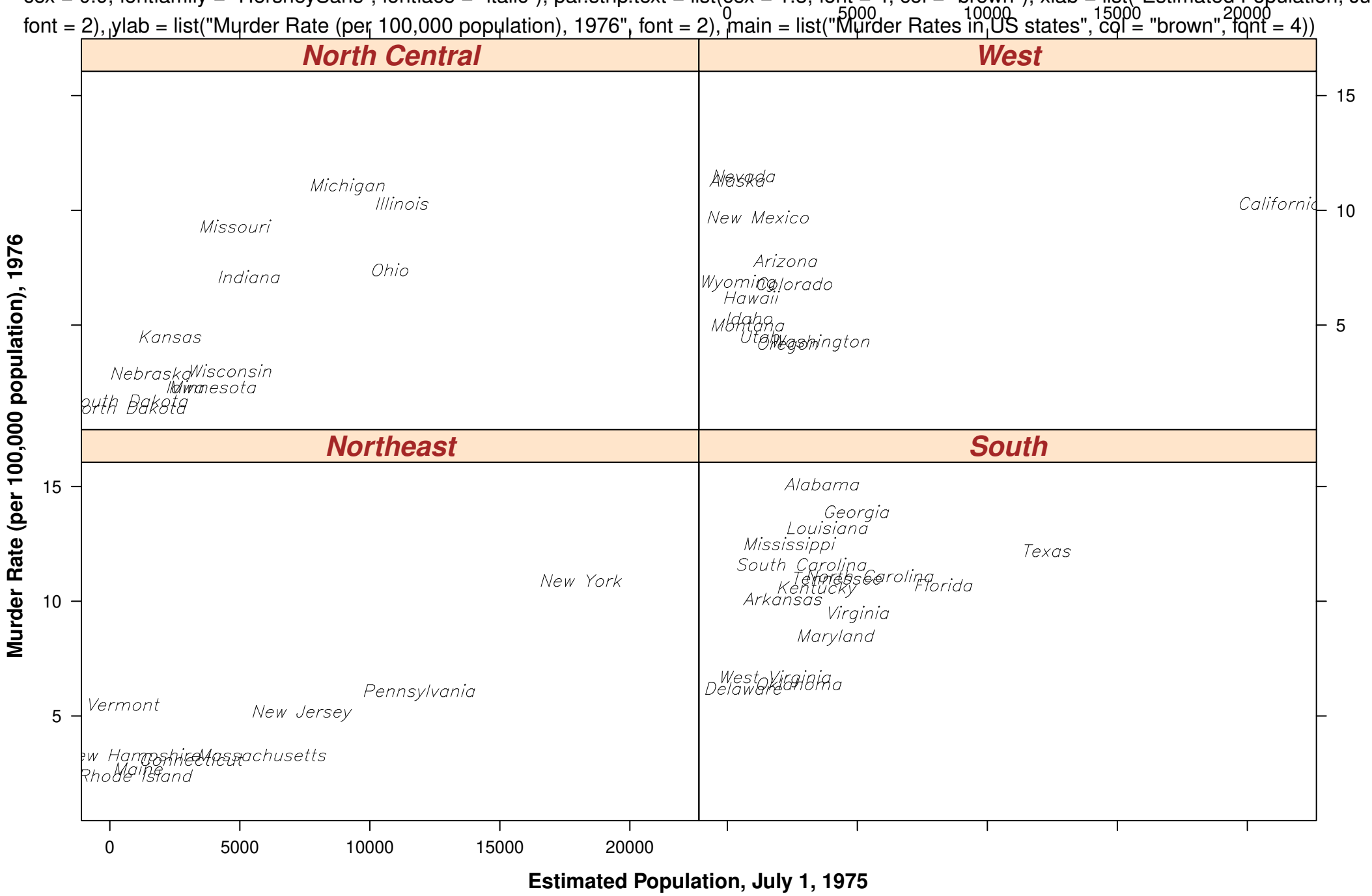
```
histogram(x, type = "density", panel = function(x, ...) {
    panel.histogram(x, ...)
    panel.densityplot(x, col = "brown", plot.points = FALSE)
})
```



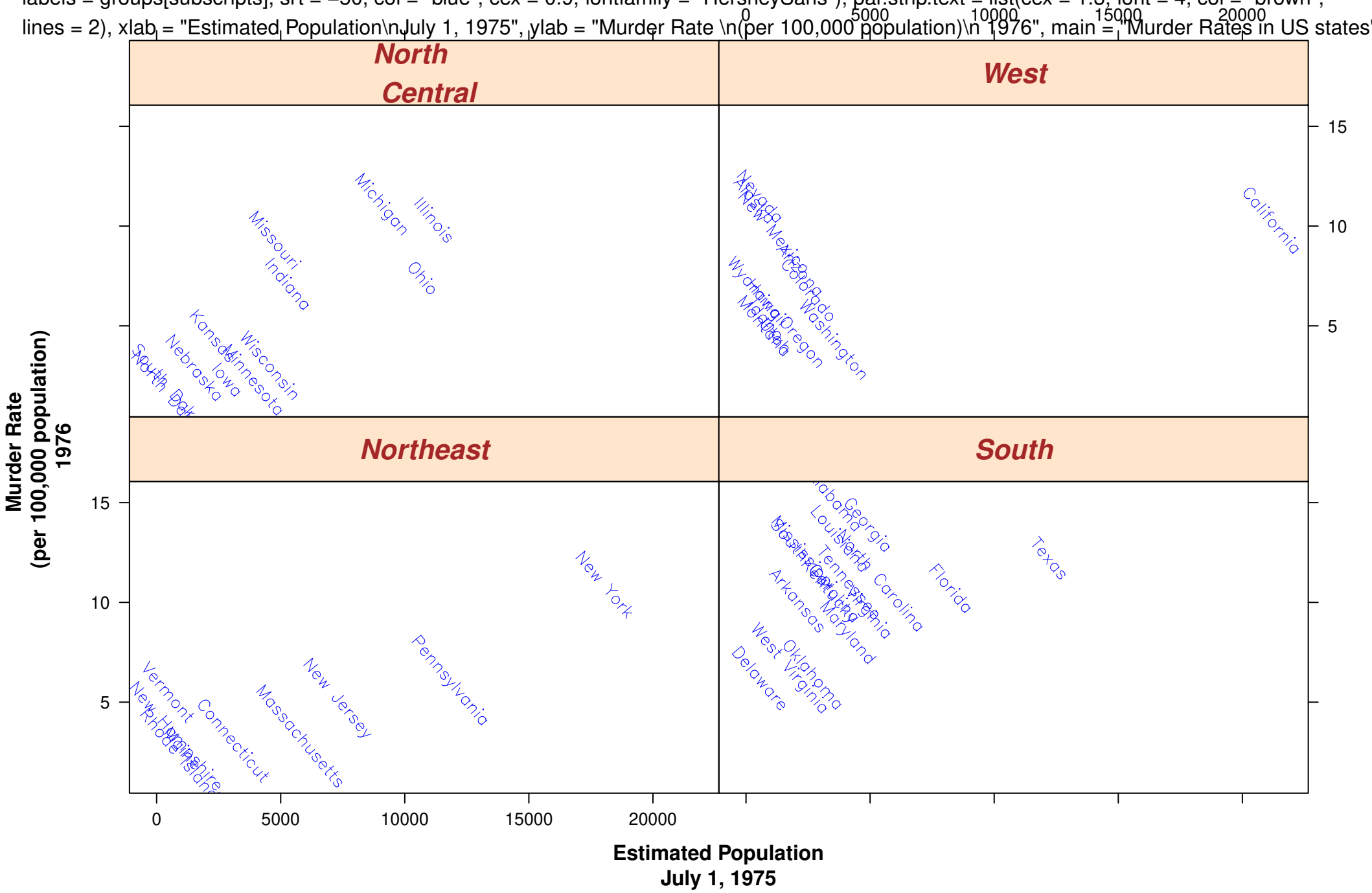
```
densityplot(~height | voice.part, data = singer, layout = c(2, 4), xlab = "Height (inches)", ylab = "Kernel Density\n with Normal Fit", main = list("Estimated
    cex = 1.4, col = "DarkOliveGreen"), panel = function(x, ...) {
    panel.densityplot(x, ...)
    panel.mathdensity(dmath = dnorm, args = list(mean = mean(x), sd = sd(x)))
})
```

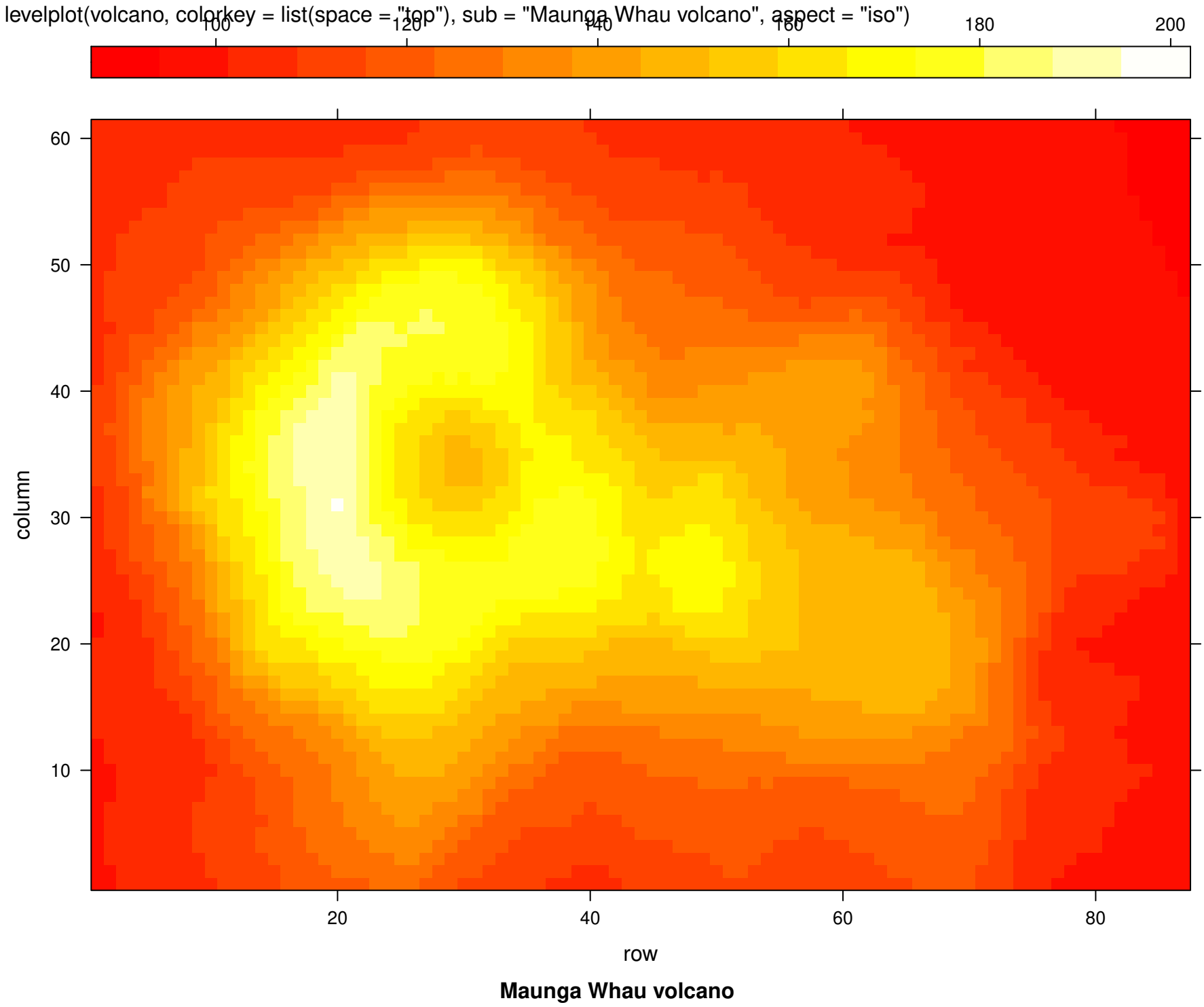


```
xyplot(Murder ~ Population | state.region, data = states, groups = state.name, panel = function(x, y, subscripts, groups) ltext(x = x, y = y, labels = groups
    cex = 0.9, fontfamily = "HersheySans", fontface = "italic"), par.strip.text = list(cex = 1.3, font = 4, col = "brown"), xlab = list("Estimated Population, July
    font = 2), ylab = list("Murder Rate (per 100,000 population), 1976", font = 2), main = list("Murder Rates in US states", col = "brown", font = 4))
```

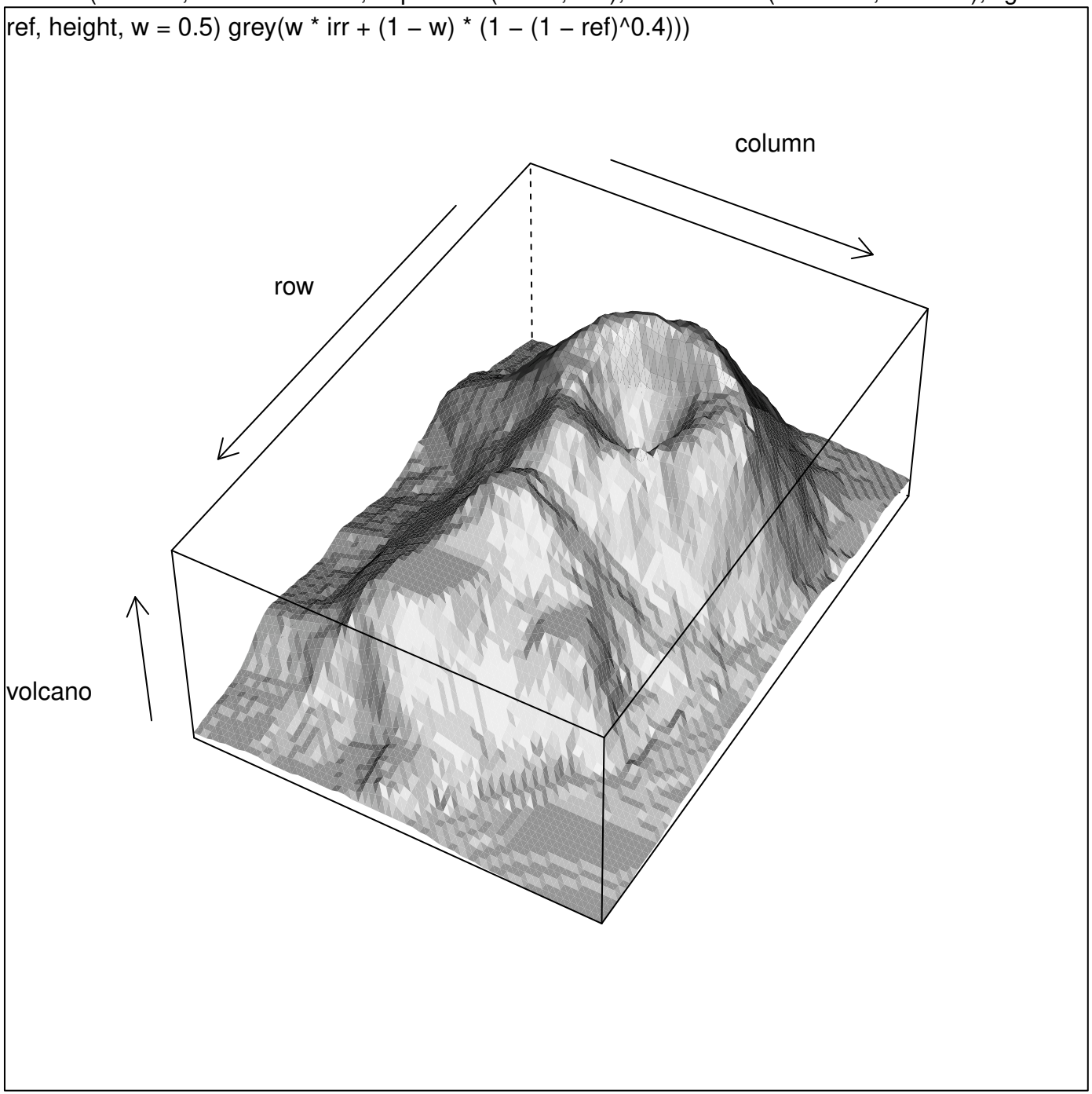


```
xyplot(Murder ~ Population | state.region, data = states, groups = as.character(state.name), panel = function(x, y, subscripts, groups) ltext(x = x, y = y,
    labels = groups[subscripts], srt = -50, col = "blue", cex = 0.9, fontfamily = "HersheySans"), par.strip.text = list(cex = 1.3, font = 4, col = "brown",
    lines = 2), xlab = "Estimated Population\nJuly 1, 1975", ylab = "Murder Rate \n(per 100,000 population)\n 1976", main = "Murder Rates in US states"
```

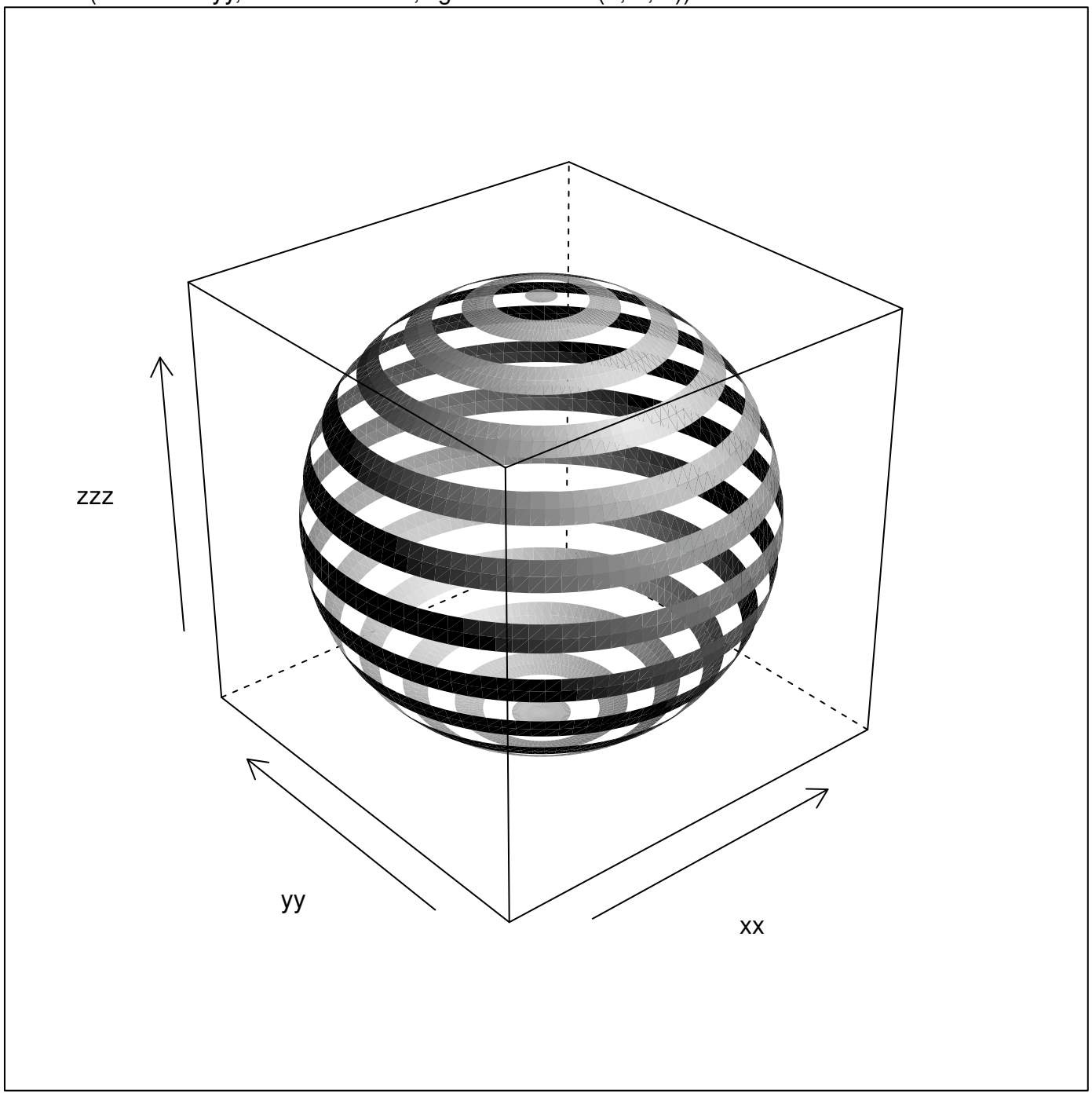


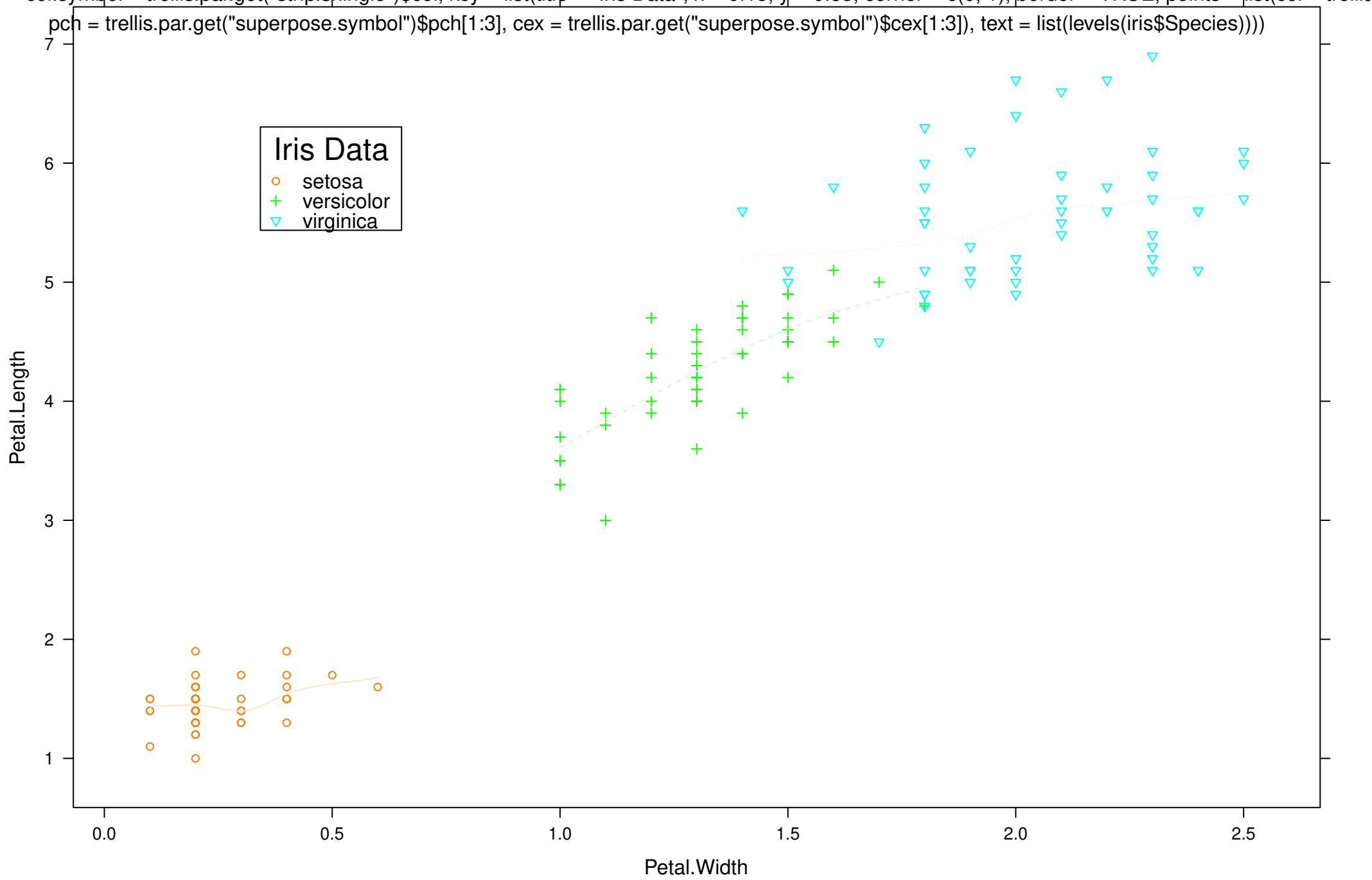
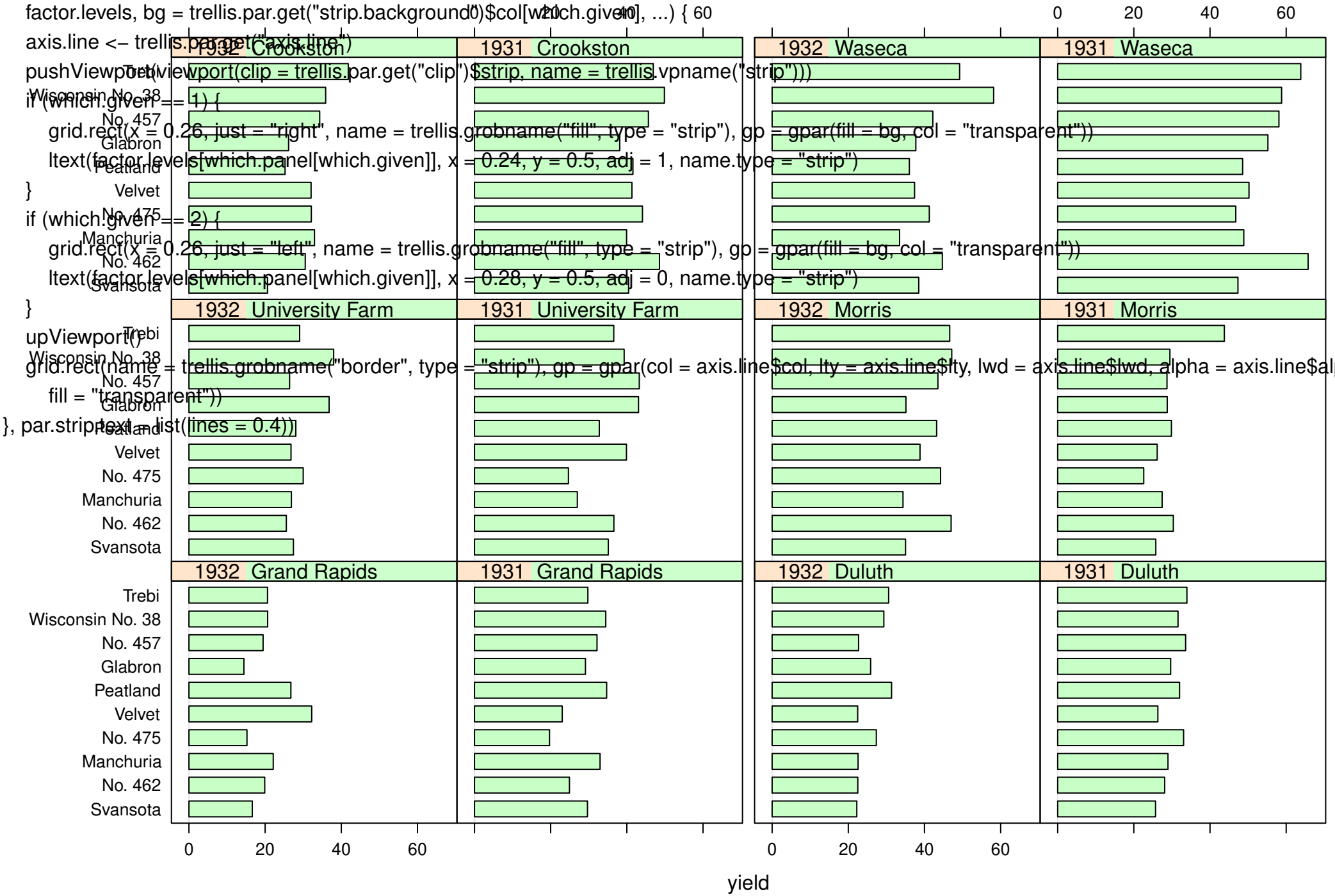

levelplot(volcano, colorkey = list(space = "top"), sub = "Maunga Whau volcano", aspect = "iso")
100
120
140
160
180
200
60
50
40
30
20
10
column
20
40
60
80
row
Maunga Whau volcano

wireframe(volcano, shade = TRUE, aspect = c(61/87, 0.4), screen = list(z = −120, x = −45), light.source = c(0, 0, 10), distance =
ref, height, w = 0.5) grey(w * irr + (1 − w) * (1 − (1 − ref)^0.4)))



wireframe(zzz ~ xx * yy, shade = TRUE, light.source = c(3, 3, 3))




xyplot(Petal.Length ~ Petal.Width, data = iris, groups = Species, panel = panel.superpose, type = c("p", "smooth"), span = 0.75, col.line = trellis.par.get(
col.symbol = trellis.par.get("strip.shingle")$col, key = list(title = "Iris Data", x = 0.15, y = 0.85, corner = c(0, 1), border = TRUE, points = list(col = trellis
pch = trellis.par.get("superpose.symbol")$pch[1:3], cex = trellis.par.get("superpose.symbol")$cex[1:3]), text = list(levels(iris$Species))))
Iris Data
setosa
versicolor
virginica
Petal.Length
Petal.Width
1
2
3
4
5
6
7
0.0
0.5
1.0
1.5
2.0
2.5

```
barchart(variety ~ yield | year * site, barley, origin = 0, layout = c(4, 3), between = list(x = c(0, 0.5, 0)), strip = function(which.given, which.panel,
    factor.levels, bg = trellis.par.get("strip.background")$col[which.given], ...) {
    axis.line <- trellis.par.get("axis.line")
    pushViewport(viewport(clip = trellis.par.get("clip")$strip, name = trellis.vpname("strip")))
    if (which.given == 1) {
        grid.rect(x = 0.26, just = "right", name = trellis.grobname("fill", type = "strip"), gp = gpar(fill = bg, col = "transparent"))
        ltext(factor.levels[which.panel[which.given]], x = 0.24, y = 0.5, adj = 1, name.type = "strip")
    }
    if (which.given == 2) {
        grid.rect(x = 0.26, just = "left", name = trellis.grobname("fill", type = "strip"), gp = gpar(fill = bg, col = "transparent"))
        ltext(factor.levels[which.panel[which.given]], x = 0.28, y = 0.5, adj = 0, name.type = "strip")
    }
    upViewport()
    grid.rect(name = trellis.grobname("border", type = "strip"), gp = gpar(col = axis.line$col, lty = axis.line$lty, lwd = axis.line$lwd, alpha = axis.line$alpha
        fill = "transparent"))
}, par.strip.text = list(lines = 0.4))
```