Introduction to the `ScottKnott` Package

Multiple Comparisons Using the Scott & Knott Algorithm

Authors

J. C. Faria
E. G. Jelihovschi
I. B. Allaman

Institution

Universidade Estadual
de Santa Cruz — UESC
Ilhéus, Bahia, Brasil

May 24, 2026

Contents

Overview

The **ScottKnott** package implements the Scott & Knott (1974) clustering algorithm as a multiple comparison method in the context of Analysis of Variance (ANOVA). Unlike classic procedures such as Tukey, Duncan, and Newman-Keuls, the Scott & Knott method forms **non-overlapping groups** of treatment means: each mean belongs to exactly one group, eliminating the ambiguity that arises when groups share members.

The algorithm proceeds by sorting the observed treatment means in decreasing order and then recursively partitioning them into two sub-groups, applying a likelihood-ratio test at each split. The process stops when no further significant partition is found. The result is a complete, disjoint labelling of the treatment means that is easy to interpret regardless of the number of treatments.

```
library(ScottKnott)
```

1 Quick Start — Completely Randomized Design (CRD)

`CRD1` contains simulated data for a balanced CRD with **4 treatment levels** and **6 replicates** per treatment. The main function `SK()` accepts a model formula, an `aov` object, or an `lm` object. The `which` argument names the factor to be compared.

```
data(CRD1)

sk1 <- with(CRD1,
            SK(y ~ x,
               data = dfm,
               which = 'x'))
summary(sk1)
```

```
#> Goups of means at sig.level = 0.05
#>      Means G1 G2
#> tr-2 59.82  a
#> tr-3 54.11  a
#> tr-1 52.02  a
#> tr-4 40.19     b
```

The summary shows, for each level, the mean and the group letter assigned by the algorithm. Levels sharing the same letter do not differ significantly at the default 5 % level.

A single call to `plot()` produces the canonical dot plot with group letters displayed above each point:

```
plot(sk1,
     dispersion = 'mm',
     d.col = 'steelblue')
```

2 Accepted Input Classes

`SK()` dispatches on the class of its first argument. The same grouping can be obtained from a `formula`, an `aov` object, or an `lm` object.

```
## From: aov
av1 <- with(CRD1, aov(y ~ x, data = dfm))
sk2 <- SK(av1, which = 'x')
summary(sk2)
```

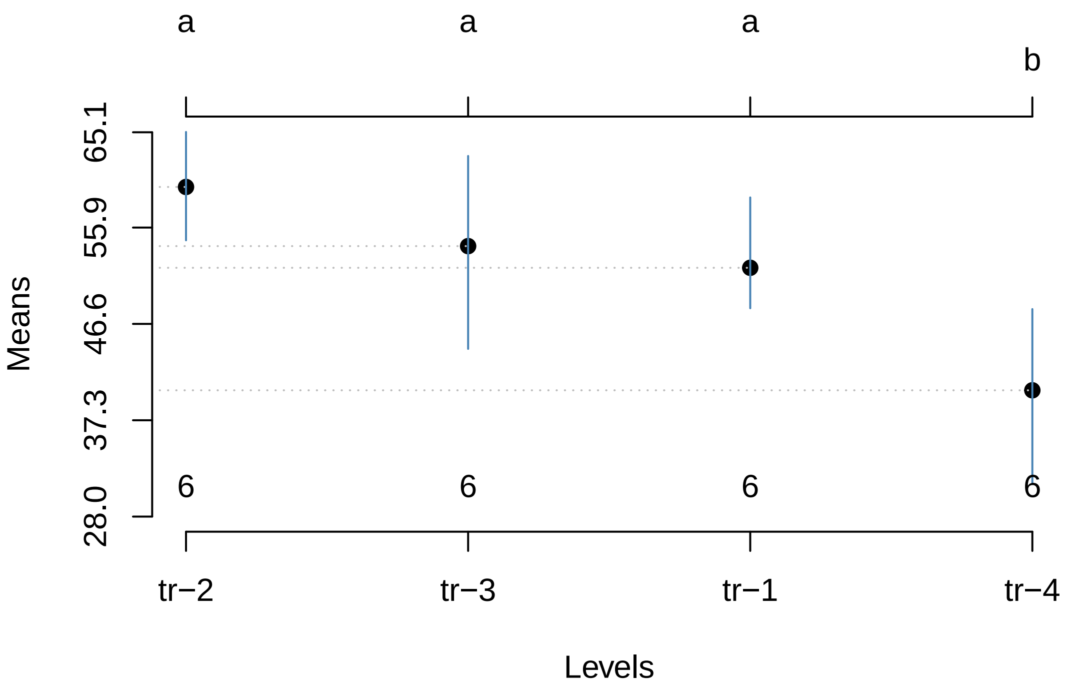

Figure 1: CRD1: treatment means with min-max dispersion bars and SK groups.

```
#> Goups of means at sig.level = 0.05
#>      Means G1 G2
#> tr-2 59.82  a
#> tr-3 54.11  a
#> tr-1 52.02  a
#> tr-4 40.19     b


## From: lm
lm1 <- with(CRD1, lm(y ~ x, data = dfm))
sk3 <- SK(lm1, which = 'x')
summary(sk3)
```

```
#> Goups of means at sig.level = 0.05
#>      Means G1 G2
#> tr-2 59.82  a
#> tr-3 54.11  a
#> tr-1 52.02  a
#> tr-4 40.19     b
```

3 Unbalanced Data

When observations are missing, `SK()` automatically adjusts the means using the Least-Squares Means methodology (via the **emmeans** package). The analysis proceeds identically to the balanced case.

```
## Remove the first observation to create an unbalanced dataset
u_sk1 <- with(CRD1,
```

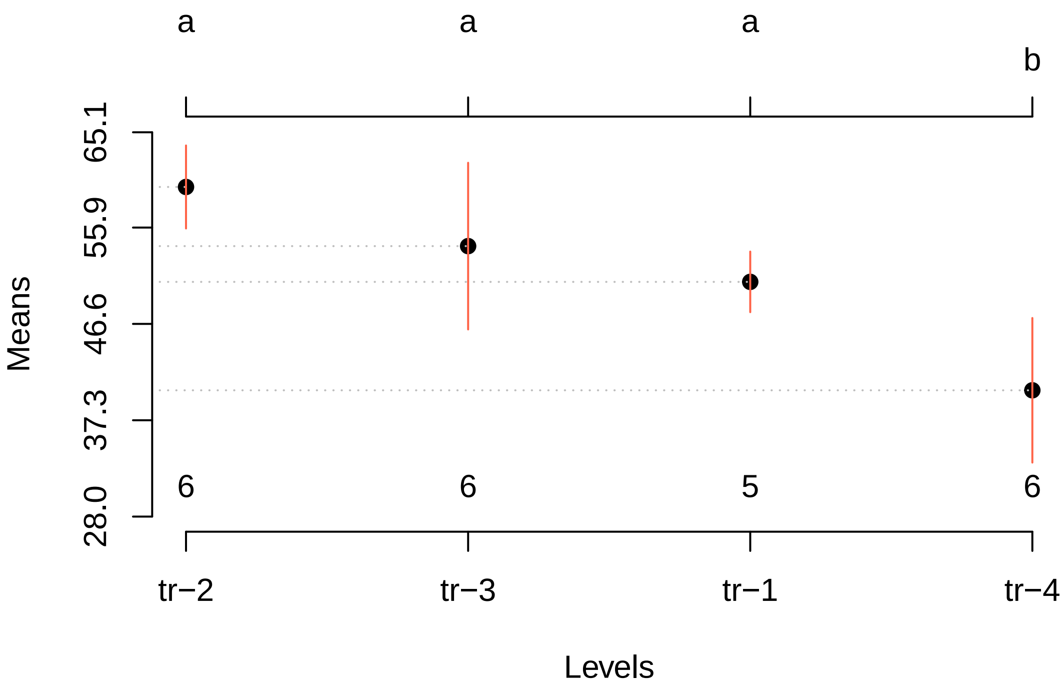

Figure 2: CRD1 (unbalanced): adjusted means with SD bars.

```
              SK(y ~ x,
                 data = dfm[-1, ],
                 which = 'x'))
summary(u_sk1)
```

```
#> Goups of means at sig.level = 0.05
#>      Means G1 G2
#> tr-2 59.82  a
#> tr-3 54.11  a
#> tr-1 50.66  a
#> tr-4 40.19     b
```

The number of replicates shown at the bottom of the plot reflects the actual (unequal) sample sizes:

```
plot(u_sk1, dispersion = 'sd', d.col = 'tomato')
```

4 Randomized Complete Block Design (RCBD)

`RCBD` contains simulated data for a design with **5 treatment levels** and **4 blocks**. The blocking factor `blk` is included in the formula; `which` selects the factor of interest for the comparison.

```
data(RCBD)

sk4 <- with(RCBD,
            SK(y ~ blk + tra,
               data = dfm,
               which = 'tra'))
```

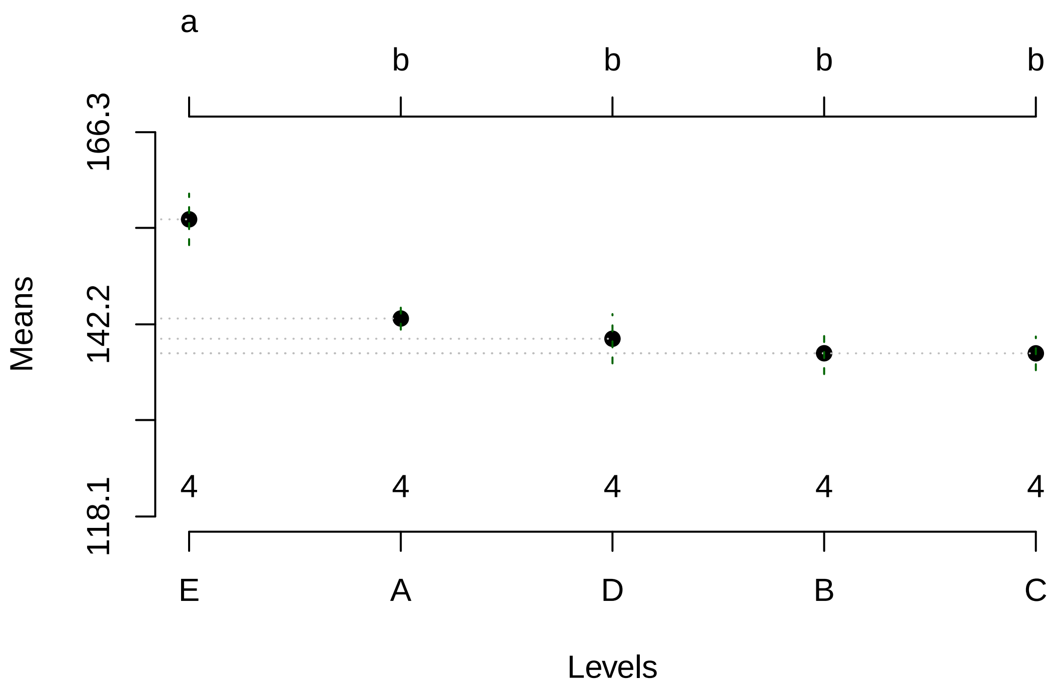

Figure 3: RCBD: treatment means with individual CI bars.

```
summary(sk4)

#> Goups of means at sig.level = 0.05
#>     Means G1 G2
#> E 155.37  a
#> A 142.93     b
#> D 140.39     b
#> B 138.57     b
#> C 138.56     b
```

```
plot(sk4,
     dispersion = 'ci',
     d.col = 'darkgreen',
     d.lty = 2)
```

5 Significance Level

The default significance level is `sig.level = 0.05`. Stricter or looser levels lead to fewer or more groups, respectively.

```
## alpha = 0.01 (stricter)
sk_01 <- with(RCBD,
              SK(y ~ blk + tra,
                 data = dfm,
                 which = 'tra',
                 sig.level = 0.01))

## alpha = 0.10 (looser)
```

```
sk_10 <- with(RCBD,
             SK(y ~ blk + tra,
                data = dfm,
                which = 'tra',
                sig.level = 0.10))

cat('--- sig.level = 0.01 ---\n')

#> --- sig.level = 0.01 ---

summary(sk_01)

#> Goups of means at sig.level = 0.01
#>    Means G1
#> E 155.37  a
#> A 142.93  a
#> D 140.39  a
#> B 138.57  a
#> C 138.56  a


cat('--- sig.level = 0.10 ---\n')

#> --- sig.level = 0.10 ---

summary(sk_10)

#> Goups of means at sig.level = 0.1
#>    Means G1 G2
#> E 155.37  a
#> A 142.93     b
#> D 140.39     b
#> B 138.57     b
#> C 138.56     b
```

6 Factorial Experiment (FE)

`FE` contains simulated data for a **3-factor factorial** design (N, P, K), each at 2 levels, in 4 blocks. `SK()` supports both main-effect and nested comparisons using colon notation in `which` and the `fl1` / `fl2` arguments to select the level of the nesting factor.

```
data(FE)

## Main effect: factor N
sk5 <- with(FE,
            SK(y ~ blk + N*P*K,
               data = dfm,
               which = 'N'))
summary(sk5)

#> Goups of means at sig.level = 0.05
#>    Means G1 G2
#> n1  2.75  a
#> n0  2.31     b
```

```
## Nested: levels of N within level 1 of P
sk6 <- with(FE,
            SK(y ~ blk + N*P*K,
               data = dfm,
               which = 'P:N',
               fl1 = 1))
summary(sk6)
```

```
#> Goups of means at sig.level = 0.05
#>       Means G1
#> p0/n1  2.60  a
#> p0/n0  2.41  a
```

```
## Nested: levels of N within level 2 of P
sk7 <- with(FE,
            SK(y ~ blk + N*P*K,
               data = dfm,
               which = 'P:N',
               fl1 = 2))
summary(sk7)
```

```
#> Goups of means at sig.level = 0.05
#>       Means G1 G2
#> p1/n1  2.90  a
#> p1/n0  2.20     b
```

7 Split-Plot Experiment (SPE)

`SPE` contains simulated data for a design with **3 whole plots** (factor P) and **4 sub-plot treatments** (factor SP). When testing the whole-plot factor, the appropriate error term must be specified via the `error` argument.

```
data(SPE)

## Sub-plot factor SP (residual error, default)
sk8 <- with(SPE,
            SK(y ~ blk + P*SP + Error(blk/P),
               data = dfm,
               which = 'SP'))
summary(sk8)
```

```
#> Goups of means at sig.level = 0.05
#>     Means G1 G2
#> sp1 17.72  a
#> sp4 16.78  a
#> sp3 16.24  a
#> sp2 13.34     b
```

```
## Whole-plot factor P (must specify the blk:P error term)
sk9 <- with(SPE,
            SK(y ~ blk + P*SP + Error(blk/P),
               data = dfm,
```

```
               which = 'P',
               error = 'blk:P'))
summary(sk9)
```

```
#> Goups of means at sig.level = 0.05
#>    Means G1
#> p1 16.70  a
#> p2 15.80  a
#> p3 15.56  a
```

8 Visualisation Options

8.1 Dispersion bars

Four dispersion options are available for `plot.SK()`, as summarised in Table 1.

Table 1: Dispersion options available for `plot.SK()`.

Option	Description
`'mm'`	Min-max range (default)
`'sd'`	±1 standard deviation
`'ci'`	Individual 95 % confidence interval
`'cip'`	Pooled 95 % confidence interval (uses MSE)

`CRD2` provides a more visually rich example with **45 treatment levels**:

```
data(CRD2)

sk10 <- with(CRD2,
             SK(y ~ x,
                data = dfm,
                which = 'x'))

col=c(rep(2, 6),
      rep(3, 36),
      rep(4, 1),
      rep(5, 2))

plot(sk10,
     dispersion='cip',
     yl=FALSE,
     id.las=2,
     col=col,
     d.col=col)
```

8.2 Comparing all four options

```
op <- par(mfrow = c(2, 2), mar = c(4, 3, 4, 1))

plot(sk1, dispersion = 'mm',  d.col = 'steelblue')
mtext('(A)', side = 3, adj = 0, line = 2, font = 2)
```

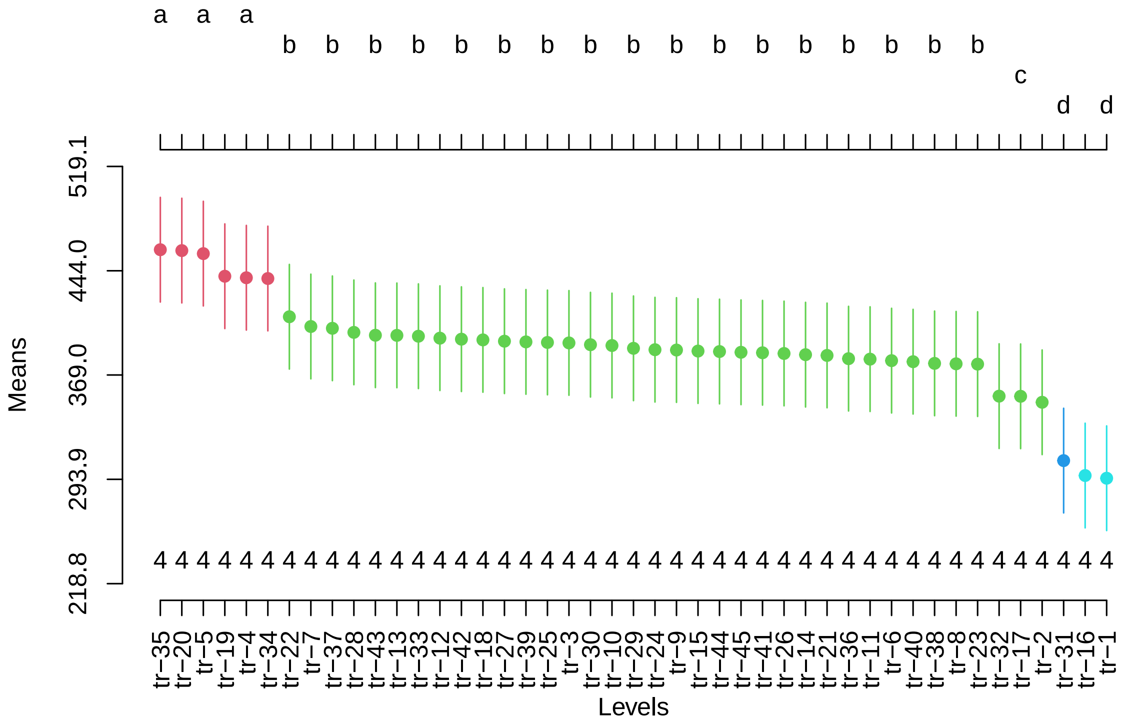

Figure 4: CRD2: 45 treatment means with pooled CI bars.

```
plot(sk1, dispersion = 'sd',  d.col = 'tomato')
mtext('(B)', side = 3, adj = 0, line = 2, font = 2)

plot(sk1, dispersion = 'ci',  d.col = 'darkgreen')
mtext('(C)', side = 3, adj = 0, line = 2, font = 2)

plot(sk1, dispersion = 'cip', d.col = 'purple')
mtext('(D)', side = 3, adj = 0, line = 2, font = 2)
```

```
par(op)
```

8.3 Boxplot

`boxplot.SK()` extends the standard boxplot by overlaying the SK group letters above the frame and drawing the treatment mean inside each box.

```
## boxplot.SK re-evaluates the data argument from the original call;
## pass CRD1$dfm directly so it is findable in any environment.
sk1_bp <- SK(y ~ x,
             data = CRD1$dfm,
             which = 'x')

boxplot(sk1_bp,
        mean.col = 'red',
        mean.lwd = 2,
        args.legend = list(x = 'topright'))
```

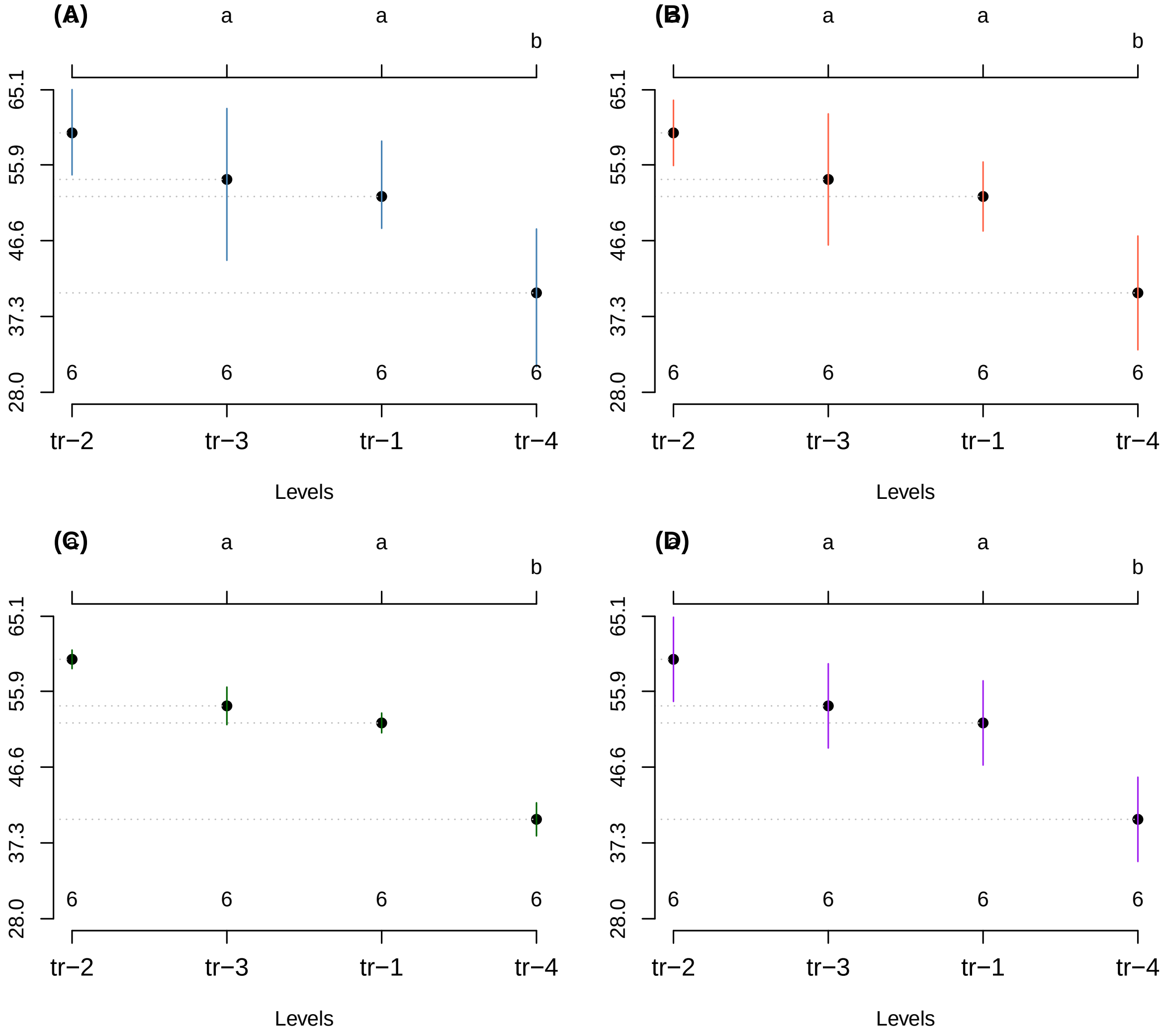

Figure 5: The four dispersion options applied to CRD1. (A) mm: min-max range; (B) sd: standard deviation; (C) ci: individual confidence interval; (D) cip: pooled confidence interval.

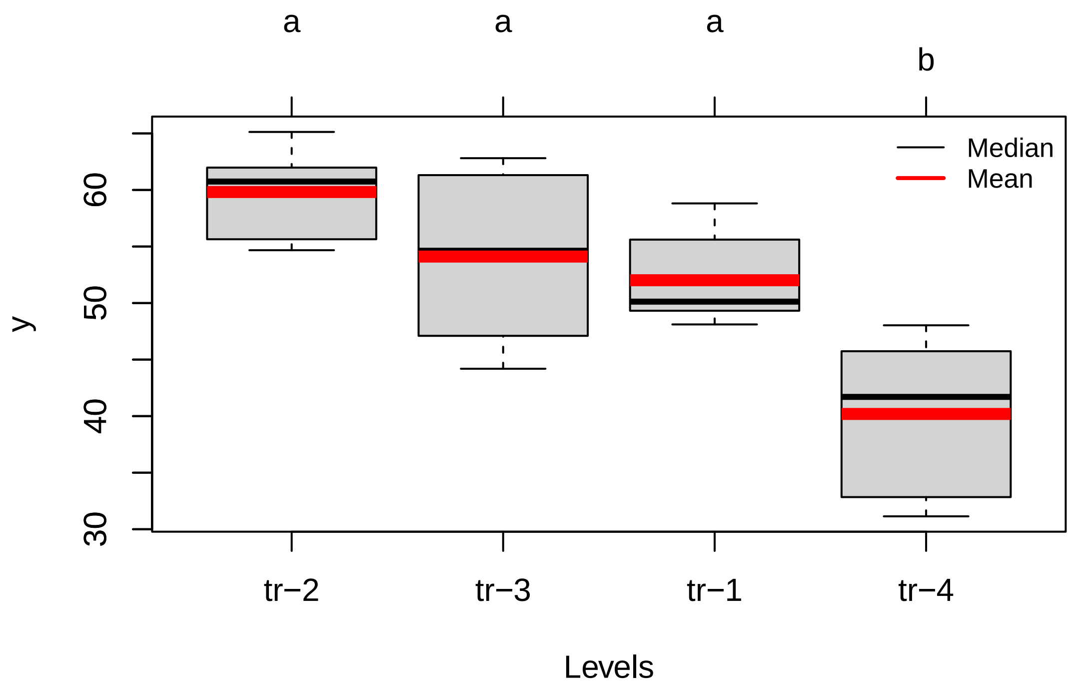

Figure 6: CRD1: boxplot with SK group labels and means (red line).

9 Tabular Output

`xtable()` converts an `SK` result to an `xtable` object for inclusion in LaTeX or HTML documents. Table 2 shows the Scott & Knott grouping for the RCBD example.

```
library(xtable)

tb <- xtable(sk4,
             caption = 'RCBD: Scott \\& Knott grouping of treatment means.',
             label = 'tab:rcbd',
             digits = 3)
print(tb,
      type = 'latex',
      caption.placement = 'top',
      include.rownames = FALSE,
      booktabs = TRUE,
      table.placement = 'H')
```

Table 2: RCBD: Scott & Knott grouping of treatment means.

Treatment	Means	G1	G2	Sig.level
E	155.37	a		0.050
A	142.93		b	
D	140.39		b	
B	138.57		b	
C	138.56		b	

10 Mixed Models with lme4

`SK()` also accepts `lmerMod` objects from the **lme4** package, useful when random effects need to be modelled explicitly.

```
library(lme4)

data(RCBD)

lmer1 <- with(RCBD,
              lmer(y ~ (1|blk) + tra,
                   data = dfm))

#> boundary (singular) fit:  see help('isSingular')


sk11 <- SK(lmer1, which = 'tra')
summary(sk11)

#> Goups of means at sig.level = 0.05
#>    Means G1 G2
#> E 155.37  a
#> A 142.93     b
#> D 140.40     b
#> B 138.57     b
#> C 138.56     b
```

References

References

[1] Scott, R. J. and Knott, M. (1974). A cluster analysis method for grouping means in the analysis of variance. *Biometrics*, **30**, 507–512.

[2] Jelihovschi, E. G., Faria, J. C., and Allaman, I. B. (2014). ScottKnott: A package for performing the Scott-Knott clustering algorithm in R. *Trends in Applied and Computational Mathematics*, **15**(1), 3–17.

[3] Conrado, T. V., Ferreira, D. F., Scapim, C. A., and Maluf, W. R. (2017). Adjusting the Scott-Knott cluster analyses for unbalanced designs. *Crop Breeding and Applied Biotechnology*, **17**(1), 1–9. doi:10.1590/1984-70332017v17n1a1