Some examples using the BHH2 package[*]

Ernesto Barrios
University of Wisconsin-Madison

March, 2005

Contents

[*]BHH2 current version: 0.2-1

1 Introduction

Many of the calculations and figures in *Statistics for Experimenters II* (Box, Hunter, and Hunter (2005)) can be done calling only a few number of R functions. There are some plots and computations more complex though. The purpose of the functions in the BHH2 package is to make easier some of these more elaborated procedures. We have left apart, however, a set of functions for Bayesian screening and follow-up designs, discussed in chapter 7 of the book, that we collected in the BsMD package (Barrios (2005)).

Most of the functions in the package are still under construction and can be improved for computational efficiency and to extend their applicability. It is assumed that the reader knows the basics of R, some elements of plotting and the use of the function `lm` for the fitting of linear models. Various documents for the understanding and exploitation of the R language are available at CRAN (`http://cran.r-project.org`). In particular, *An Introduction to R* by Venables et˜al. (2005), is enough to understand all the functions in the BHH2 package and if necessary to modify them to better fit your needs.

In this document we introduce most of the functions included in the BHH2 package: `dotPlot`, `anovaPlot`, `lambdaPlot`, `permtest` and `ddDesMatrix`. A list of all the functions and data sets can be found in the help pages. To illustrate the use of the functions just mentioned, we work out some of the examples in *Statistics for Experimenters II*, referred here after as BHH2.

2 Permutation Test

Function `permtest`

In the two-samples problem ($(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_m)$), the statistics t and F are computed for location and dispersion comparisons, and their p-values determined. Spooled variance is used in the equal means test. Next, the observations are combined in different samples as

$$(x_1^{(i)}, \ldots, x_n^{(i)}) \text{ and } (y_1^{(i)}, \ldots, y_m^{(i)}), \qquad i = 1, \ldots, N$$

where $N = \binom{n+m}{n}$ is the total number of possible combinations. For each samples pair, $t^{(i)}$ and $F^{(i)}$ statistics are computed and then it is determined how many $t^{(i)} \le t$ and $F^{(i)} \le F$, to calculate the corresponding percentiles from the permutation (randomization) distribution.

In the one-sample case, all the $N = 2^n$ samples $(\pm x_1, \ldots, \pm x_n)$ are generated and later determine how many $t^{(i)} \le t$, to calculate the sample's percentile from the permutation distribution.

Function `permtest` computes the observed t and F statistics and reports the corresponding p-values from the theoretical and permutation distributions.

The following example corresponds to the Modified Fertilizer Mixtures for Tomato Plants example (BHH2, Ch.˜3) . Samples from 2 different fertilizer mixtures are compared. In addition to the statistics and percentiles, the function `permtest` reports `N`, the number of different samples generated to build the permutation distribution.

```
> data(tomato.data)
> attach(tomato.data)
> a <- pounds[fertilizer == "A"]
> b <- pounds[fertilizer == "B"]
> permtest(b, a)
```

```
            N           t.obs   t-Dist:P(>t) PermDist:P(>t)          F.obs
  462.0000000       0.4436848      0.3338727      0.3290043      0.5620532
 F-Dist:P(>F) PermDist:P(>F)
    0.7299968       0.7835498

> detach()
```

The Boy's Shoes example is a randomized paired comparison of materials A and B (BHH2, Ch.~3). It is of interest to test whether there is a significant difference between materials or not.

```
> data(shoes.data)
> permtest(shoes.data$matB - shoes.data$matA)

            N           t.obs   t-Dist-P(>t) PermDist-P(>t)
 1.024000e+03    3.348877e+00   4.269390e-03   4.882812e-03
```

The function is limited to small sample sizes ($N < 20$). See also exactRankTests and DAAG packages for other functions for permutation tests.

3 Dot Diagrams

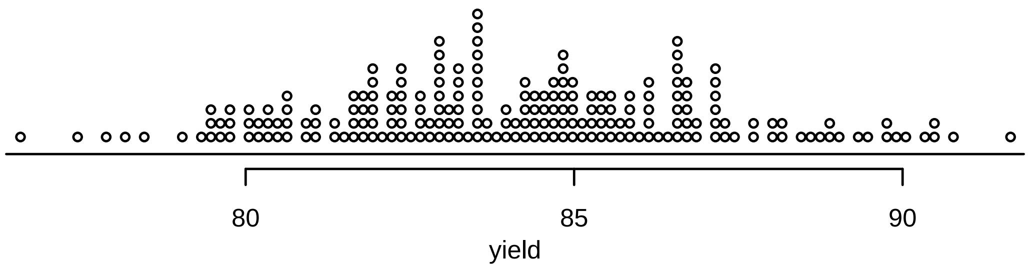
Function dotPlot

The dotPlot function produces a scatter plot similar to a stem-and-leaf plot or a histogram. Stacked or partially overlapped dots are used to display the frequency distribution. The dot plot has the nice property of displaying all the individual observations for moderated size data sets, useful to visualize reference distributions.

In this example we display the dot plot of the yield data in the Industrial Process Example.

```
> par(mar = c(4, 1, 2, 1), mgp = c(2, 1, 0), cex = 0.7)
> data(tab03B1)
> dotPlot(tab03B1$yield, main = "Dot plot: Industrial Process Example",
+     xlab = "yield")
```

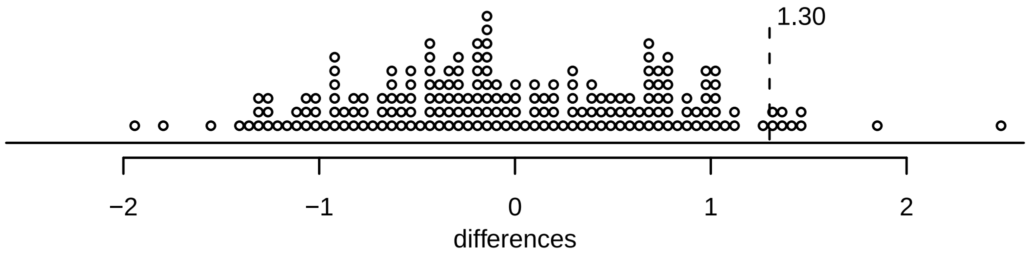
Dot plot: Industrial Process Example



The following plot represents the reference distribution of 191 differences between averages of disjoint sets of 10 observations of the Industrial Process Example. See figures 3.3 and 3.5a and tables 3B of BHH2.

```
> par(mar = c(4, 1, 2, 1), mgp = c(2, 1, 0), cex = 0.7)
> data(tab03B2)
> plt <- dotPlot(tab03B2$diff10, xlim = 2.55 * c(-1, +1), xlab = "differences",
+     main = "Dot plot: Reference Distribution of Differences")
> segments(1.3, 0, 1.3, max(plt$y), lty = 2)
> text(1.3, max(plt$y), labels = " 1.30", adj = 0)
```

Dot plot: Reference Distribution of Differences



Function `dots`

The function `dots` is used by the functions `dotPlot` and `anovaPlot` to display the dots. See the help pages for details.

4 Graphical Anova

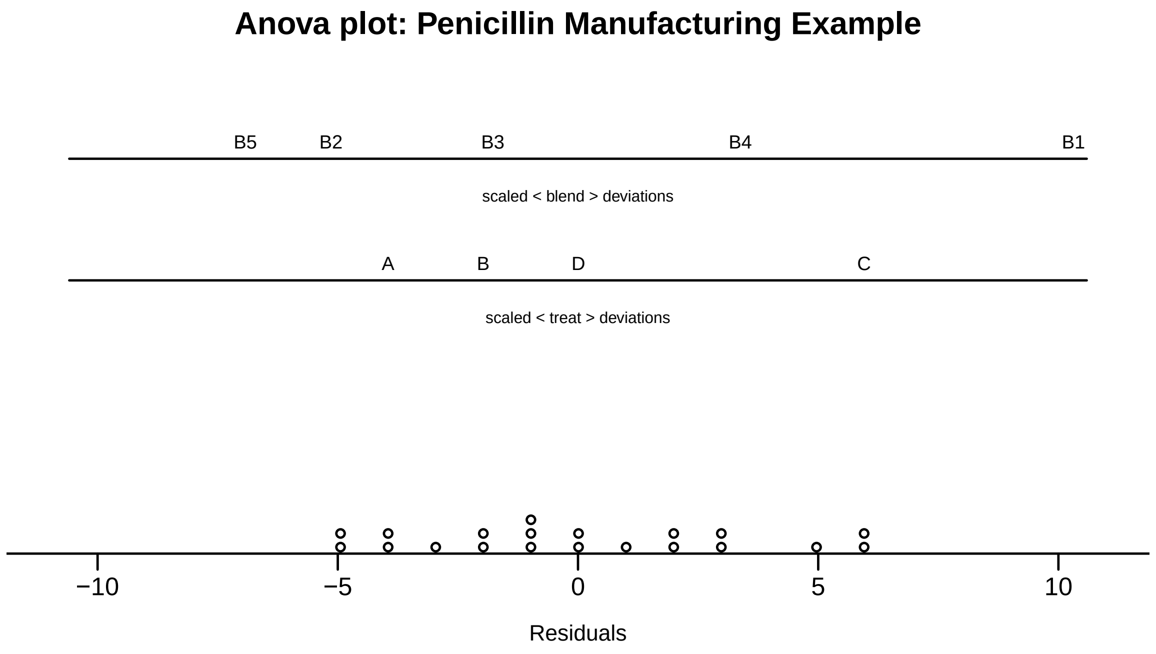
Function `anovaPlot`

The *anova plots* display in the same plot scatter diagrams of the scaled factor effects and the residuals, as reference distribution, so visual inspection of the plot would suggest which factor effects may be different from the experimental and model error.

To make a fair comparison between the factor effects and the residuals distribution, it is necessary to scale the effects properly. For example, for the factor A, the effect is multiplied by $\sqrt{n/k}$, where k and n are the degrees of freedom of factor A and the residuals respectively, taken from the table of analysis of variance.

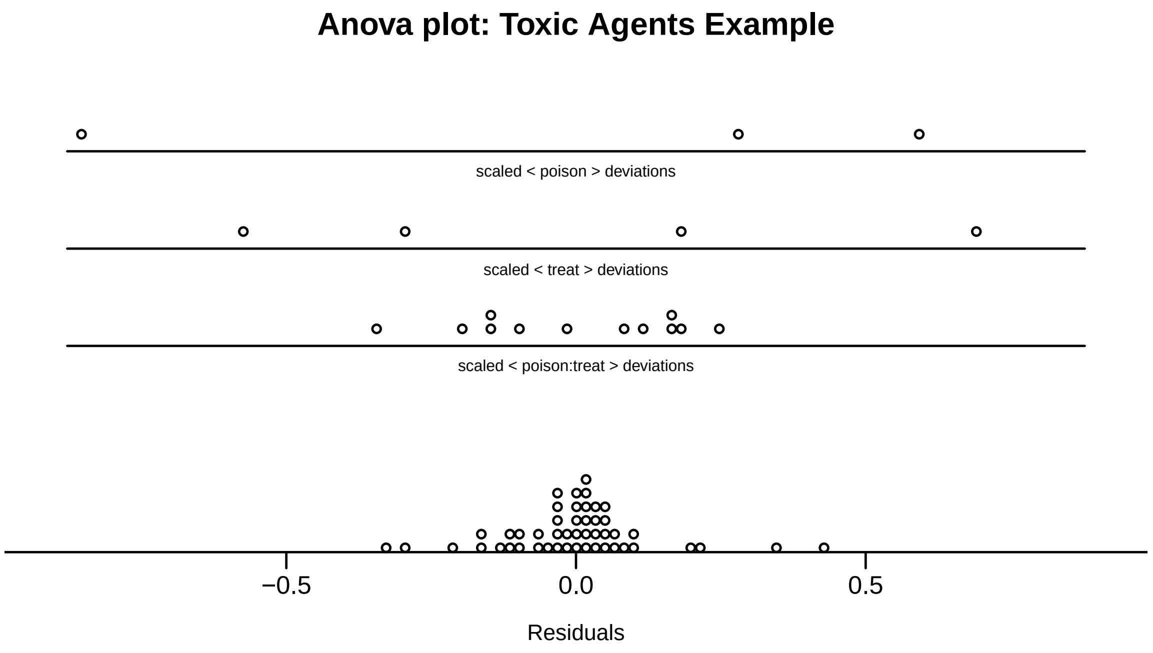
The next example is the Penicillin Yield experiment, in BHH2, Ch.˜4. A randomized block experiment is considered in the study of the process of manufacturing penicillin under different treatments (A, ..., D). The yield of the process is the response and the different product blends are used as blocking factor. A anova model (`aov`) is fitted and its anova plot displayed.

```
> par(mfrow = c(1, 1), mar = c(3, 1, 2, 1), cex = 0.7)
> data(penicillin.data)
> penicillin.aov <- aov(yield ~ blend + treat, data = penicillin.data)
> anovaPlot(penicillin.aov, main = "Anova plot: Penicillin Manufacturing Example",
+     labels = TRUE, cex.lab = 0.6)
```



The following example corresponds to the Toxic Agents example in BHH2, chapter 8, taken from Box and Cox (1964). The response is the survival time of groups of animals. The kind of poison and the type of treatment used with the animals are the experimental factors. The anova plot, similar to figure 8.5a. in BHH2 is generated.

```
> par(mfrow = c(1, 1), mar = c(3, 1, 2, 1), cex = 0.7)
> data(poison.data)
> poison.lm <- lm(y ~ poison * treat, data = poison.data)
> anovaPlot(poison.lm, main = "Anova plot: Toxic Agents Example",
+     cex.lab = 0.6)
```

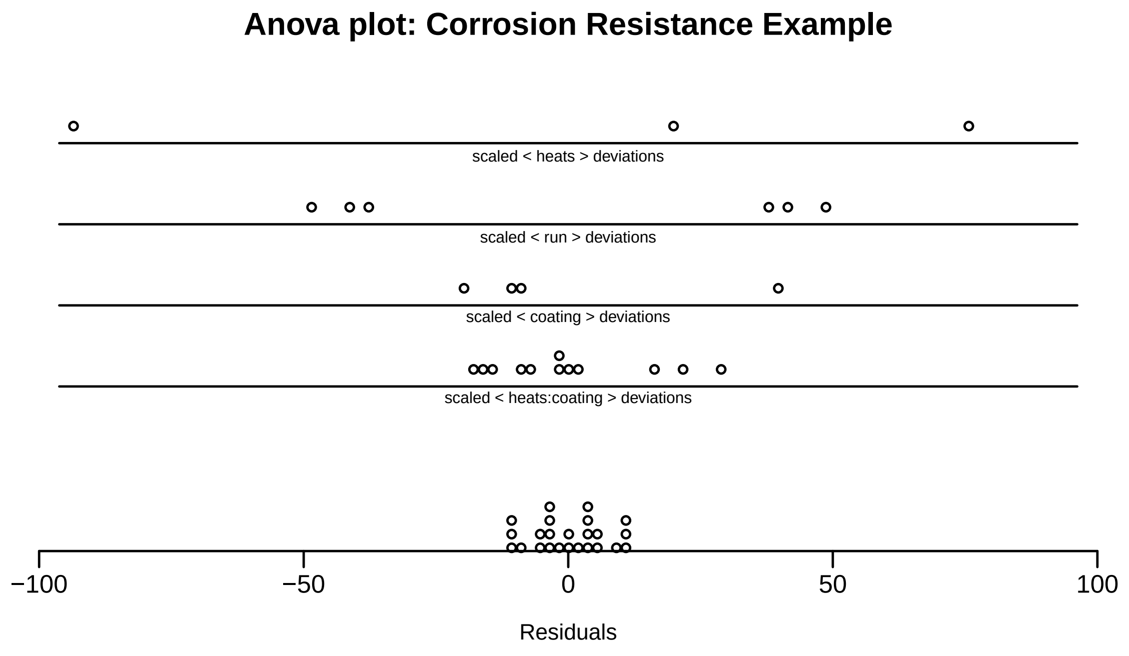


The function `anovaPlot` can be used in simple cases of split-plot experiments. Consider for example the Corrosion Resistance Study in BHH2, Ch 9. The response is the resistance against corrosion of steel bars treated under different temperatures (`heats`) and with distinct coatings (`coating`) allocated in 6 castings (`run`). See table 9.1.

The factor `heats` is allocated in the whole plot while the `coatings` factor and the interaction `heats:coating` in the sub-plot. The proper comparison of `heats` levels should be done with respect to the castings (`run`) as whole-plot errors. The factors `coating` and `heats:coating` should be compared against the residuals distribution (the sub-plot errors).

Compare the plot below with figure 9.1 in BHH2.

```
> par(mfrow = c(1, 1), mar = c(3, 1, 2, 1), cex = 0.7)
> data(corrosion.data)
> corrosion.aov <- aov(resistance ~ heats + run + coating +
+     heats:coating, data = corrosion.data)
> anovaPlot(corrosion.aov, main = "Anova plot: Corrosion Resistance Example",
+     cex.lab = 0.6)
```



5 Lambda Plot

Function `lambdaPlot`

The one-parameter Box-Cox transformation of the response y is defined as

$$Y = \frac{y^\lambda - 1}{\lambda}$$

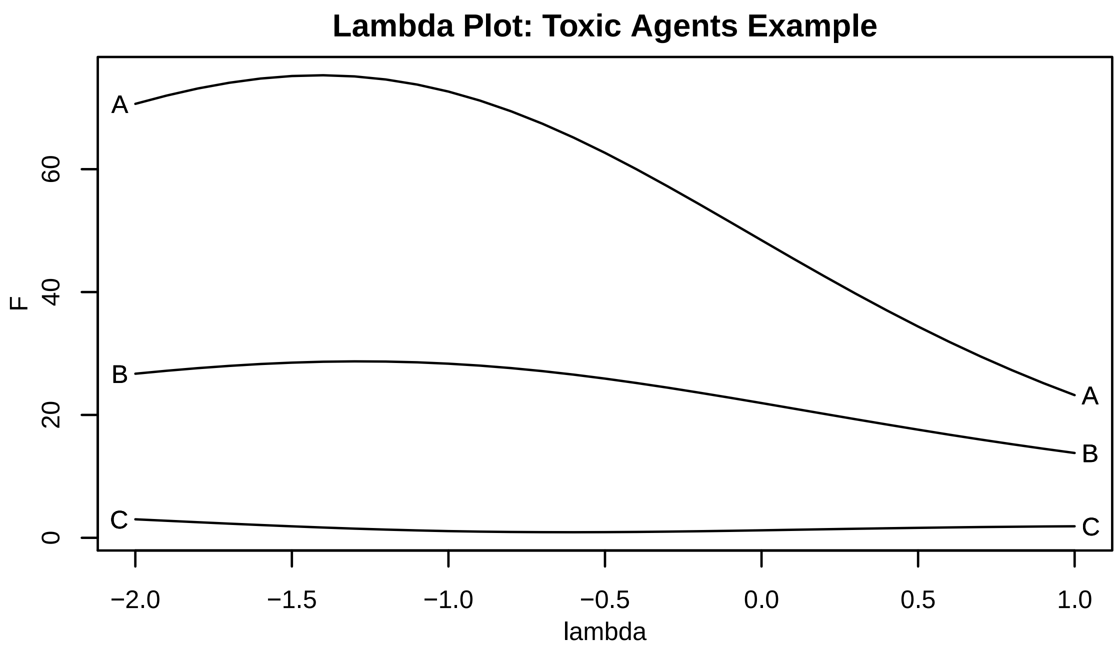
An empirical determination of the parameter λ is presented in chapter 8 of BHH2. *Lambda plots* are also introduced as an alternative way for finding appropriate values of λ. Basically, a lambda plot of a linear model is the trace over various values of λ of the relevant statistics: coefficients' t or F statistics, or the model's global F-ratio.

In the next example it is constructed the lambda plot for the model fitted for the Toxic Agents example, introduced in the last section. In this case, the coefficients' F-statistic are traced for $-2 \leq \lambda \leq 1$.

```
> par(mar = c(4, 3, 2, 1), mgp = c(2, 1, 0), cex = 0.7)
> data(poison.data)
> lambdaPlot(poison.lm, lambda = seq(-2, 1, by = 0.1), stat = "F",
+     global = FALSE, cex = 0.6, main = "Lambda Plot: Toxic Agents Example")
```

```
[1] "y"
          term label
1       poison     A
```

```
2        treat     B
3 poison:treat     C
```

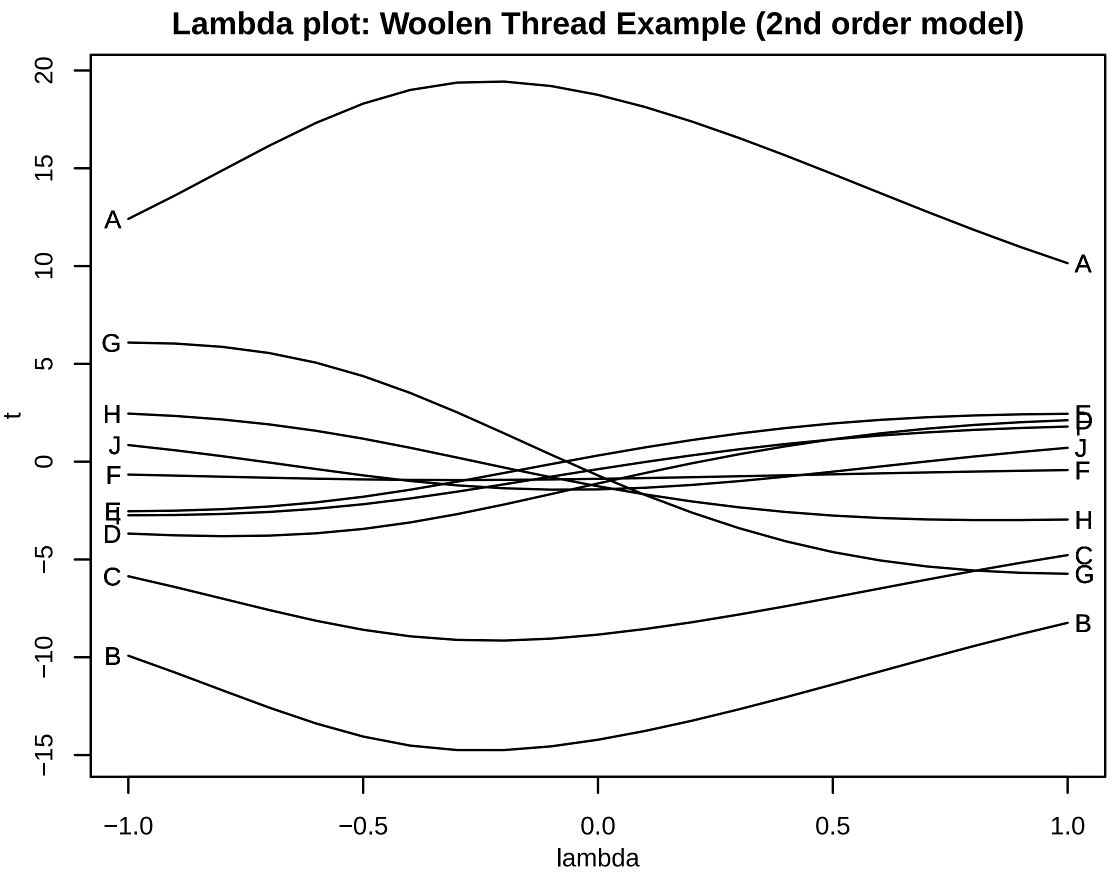


To illustrate the lambda plots when tracing the coefficients' *t*-statistic, consider the Woolen Thread experiment, presented in chapter 12 and used previously in Box and Cox (1964). A 3^3 factorial design was run with lifetime as response (`y`) affected by 3 factors: length (`x1`), amplitude (`x2`) and load (`x3`). A second degree model was fitted to the original response and its Box-Cox transformation for various values of λ ($-1 \leq \lambda \leq 1$). Compare the plot below with figure 12.7 in BHH2. The printed table will allow you to match the labels in the figure with the terms in the model.

```
> par(mar = c(4, 3, 2, 1), mgp = c(2, 1, 0), cex = 0.7)
> data(woolen.data)
> woolen.lm <- lm(y ~ x1 + x2 + x3 + I(x1^2) + I(x2^2) + I(x3^2) +
+     I(x1 * x2) + I(x1 * x3) + I(x2 * x3) + I(x1 * x2 * x3),
+     data = woolen.data)
> lambdaPlot(woolen.lm, main = "Lambda plot: Woolen Thread Example (2nd order model)",
+     stat = "t", cex = 0.6)
```

```
                term label
1        (Intercept)
2                 x1     A
3                 x2     B
4                 x3     C
```

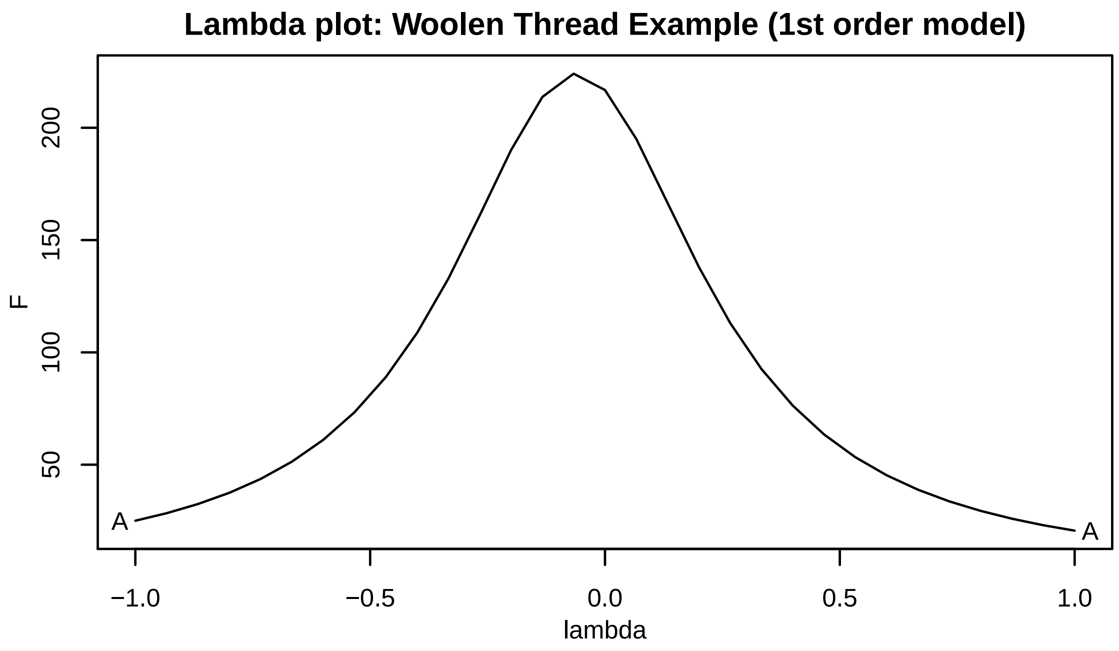
```
5          `I(x1^2)`     D
6          `I(x2^2)`     E
7          `I(x3^2)`     F
8       `I(x1 * x2)`     G
9       `I(x1 * x3)`     H
10      `I(x2 * x3)`     I
11 `I(x1 * x2 * x3)`     J
```



The "global-F" lambda plot for the corresponding first order model in the Woolen Thread example is shown next. (See BHH2, figure 12.8.)

```
> par(mar = c(4, 3, 2, 1), mgp = c(2, 1, 0), cex = 0.7)
> lambdaPlot(lm(y ~ x1 + x2 + x3, data = woolen.data), lambda = seq(-1,
+     1, length = 31), main = "Lambda plot: Woolen Thread Example (1st order model)",
+     stat = "F", global = TRUE)
```

```
[1] "y"
   term label
1 Model     A
```



6 Design Matrices

Function `ffDesMatrix`

Function `ffDesMatrix` is used to build 2-level (-1,+1) factorial designs. If generators different from `NULL` are provided to the function, the corresponding fractional factorial is built. (See BHH2, Table 6.22, p 272.) For example, the design matrix of a 2^{5-1} fractional factorial design with defining relation $I = 12345$, is built by

```
> print(X <- ffDesMatrix(5, gen = list(c(-5, 1, 2, 3, 4))))
```

```
      [,1] [,2] [,3] [,4] [,5]
 [1,]   -1   -1   -1   -1   -1
 [2,]    1   -1   -1   -1    1
 [3,]   -1    1   -1   -1    1
 [4,]    1    1   -1   -1   -1
 [5,]   -1   -1    1   -1    1
 [6,]    1   -1    1   -1   -1
 [7,]   -1    1    1   -1   -1
 [8,]    1    1    1   -1    1
 [9,]   -1   -1   -1    1    1
[10,]    1   -1   -1    1   -1
[11,]   -1    1   -1    1   -1
[12,]    1    1   -1    1    1
```

```
[13,]   -1   -1    1    1   -1
[14,]    1   -1    1    1    1
[15,]   -1    1    1    1    1
[16,]    1    1    1    1   -1
```

See the `ffDesMatrix` help pages for details.

Function `ffFullMatrix`

The function `ffFullMatrix` is a nice companion to `ffDesMatrix`. Provided what factors to consider and the desired interaction order, from a given design matrix, the function `ffFullMatrix` builds the corresponding model matrix including columns for the constant term and the blocking factors. For example, from the fractional design just built above, we construct the matrix for a model on factors 1, 2, 3 and 4, with main effects and second order interactions and the 5th factor as blocking factor.

```
> ffFullMatrix(X, x = c(1, 2, 3, 4), maxInt = 2, blk = X[, 5])$Xa

      one bk1 x1 x2 x3 x4 x1*x2 x1*x3 x1*x4 x2*x3 x2*x4 x3*x4
 [1,]   1  -1 -1 -1 -1 -1     1     1     1     1     1     1
 [2,]   1   1  1 -1 -1 -1    -1    -1    -1     1     1     1
 [3,]   1   1 -1  1 -1 -1    -1     1     1    -1    -1     1
 [4,]   1  -1  1  1 -1 -1     1    -1    -1    -1    -1     1
 [5,]   1   1 -1 -1  1 -1     1    -1     1    -1     1    -1
 [6,]   1  -1  1 -1  1 -1    -1     1    -1    -1     1    -1
 [7,]   1  -1 -1  1  1 -1    -1    -1     1     1    -1    -1
 [8,]   1   1  1  1  1 -1     1     1    -1     1    -1    -1
 [9,]   1   1 -1 -1 -1  1     1     1    -1     1    -1    -1
[10,]   1  -1  1 -1 -1  1    -1    -1     1     1    -1    -1
[11,]   1  -1 -1  1 -1  1    -1     1    -1    -1     1    -1
[12,]   1   1  1  1 -1  1     1    -1     1    -1     1    -1
[13,]   1  -1 -1 -1  1  1     1    -1    -1    -1    -1     1
[14,]   1   1  1 -1  1  1    -1     1     1    -1    -1     1
[15,]   1   1 -1  1  1  1    -1    -1    -1     1     1     1
[16,]   1  -1  1  1  1  1     1     1     1     1     1     1
```

The function `ffFullMatrix` is useful for identifying confounding patterns in a design.

7 Combinations

Function `subsets`

The function `subsets`, by Bill Venables, generates all the different subsets of size r chosen from n different elements v. It is used by the `permtest` and `ffFullMatrix` functions. For example,

```
> subsets(n = 5, r = 3, v = c("x", "y", "z", "A", "B"))
```

```
      [,1] [,2] [,3]
 [1,] "x"  "y"  "z"
 [2,] "x"  "y"  "A"
 [3,] "x"  "y"  "B"
 [4,] "x"  "z"  "A"
 [5,] "x"  "z"  "B"
 [6,] "x"  "A"  "B"
 [7,] "y"  "z"  "A"
 [8,] "y"  "z"  "B"
 [9,] "y"  "A"  "B"
[10,] "z"  "A"  "B"
```

Function `subsets` is a nice example of a recursive function and worth to understand. See also function `combinations` of the gtools package.

8 The BsMD package

The package BsMD for Bayesian screening and follow-up designs should be thought as part of BHH2. It includes functions for the generation of Daniel and Lenth's plots, presented in chapters 5 and 6 of BHH2. It also includes functions for the Bayesian analysis of factorial designs and the problem of adding extra runs to resolve unclear factor activities. These topics are presented in chapter 7, *Additional Fractional Factorials and Analyses.*

The main functions in BsMD are:

Function `DanielPlot`

Displays the normal plot of effects from a two level factorial experiment.

Function `LenthPlot`

Plots the factor effects with significance levels based on robust estimation of contrast standard errors.

Function `BsProb`

The marginal factor posterior probabilities and model posterior probabilities from designed screening experiments are calculated according to Box and Meyer's Bayesian procedure. Methods functions `plot`, `print` and `summary` are available for function `BsProb`.

Function `MD`

Best follow-up experiments based on the MD criterion are suggested to discriminate between competing models. Methods functions `print` and `summary` are available for function `MD`.

Please refer to the help pages of BsMD for a complete list of the functions and data sets included. The package also includes the document *Using BsMD for Bayesian Screening and Model Discrimination* with various examples on the use of the functions.

9 Summary

Most of the computations in Box, Hunter, and Hunter (2005) can be done in R with only a small number of function calls. For some of the more elaborated procedures, we have included in the BHH2 package the plotting functions: `anovaPlot`, `dotPlot`, and `lambdaPlot`; the function `permtest` for randomization tests and `ddDesMatrix` for the construction of 2-levels factorial designs. The calculations involved in Bayesian screening and follow-up designs, presented in chapter 7, are more computing intensive and the dedicated package BsMD is available. For details of the functions in BHH2 please see the help pages of the package.

References

E.~Barrios. BsMD: *Bayesian Screening and MD Follow-up Designs.* Contributed Packages, (v 0.6-5), March 2005. URL `http://cran.r-project.org`.

G.~E.~P. Box and D.~R. Cox. An Analysis of Transformations (with discussion). *Journal of the Royal Statistical Society, Series B*, 26:211–246, 1964.

G.~E.~P Box, W.~G. Hunter, and J.~S. Hunter. *Statistics for Experimenters II.* Wiley, New York, 2005.

W.~N. Venables, D.~M. Smith, and the R Development Core Team. *An Introduction to R. Notes on R: A Programming Environment for Data Analysis and Graphics*, 2005. URL `http://cran.r-project.org`. ISBN: 3-900051-05-4. The document is distributed as one of the R manuals.