The R Journal

Volume 2/1, June 2010

A peer-reviewed, open-access publication of the R Foundation for Statistical Computing

Contents

The R Journal is a peer-reviewed publication of the R Foundation for Statistical Computing. Communications regarding this publication should be addressed to the editors. All articles are copyrighted by the respective authors.

Prospective authors will find detailed and up-to-date submission instructions on the Journal's homepage.

Editorial

by Peter Dalgaard

Welcome to the 1st issue of the 2nd volume of The R Journal.

I am writing this after returning from the NORDSTAT 2010 conference on mathematical statistics in Voss, Norway, followed by co-teaching a course on *Statistical Practice in Epidemiology* in Tartu, Estonia.

In Voss, I had the honour of giving the opening lecture entitled "R: a success story with challenges". I shall spare you the challenges here, but as part of the talk, I described the amazing success of R, and a show of hands in the audience revealed that only about 10% of the audience was not familiar with R. I also got to talk about the general role of free software in science and I think my suggestion that closed-source software is "like a mathematician hiding his proofs" was taken quite well.

R 2.11.1 came out recently. The 2.11.x series displays the usual large number of additions and corrections to R, but if a single major landmark is to be pointed out, it must be the availability of a 64-bit version for Windows. This has certainly been long awaited, but it was held back by the lack of success with a free software 64-bit toolchain (a port using a commercial toolchain was released by REvolution Computing in 2009), despite attempts since 2007. On January 4th this year, however, Gong Yu sent a message to the R-devel mailing list that he had succeeded in building R using a version of the MinGW-w64 tools. On January 9th, Brian Ripley reported that he was now able to build a version as well. During Winter and Spring this developed into almost full-blown platform support in time for the release of R 2.11.0 in April. Thanks go to Gong Yu and the "R Windows Trojka", Brian Ripley, Duncan Murdoch, and Uwe Ligges, but the groundwork by the MinGW-w64 team should also be emphasized. The MinGW-w64 team leader, Kai Tietz, was also very helpful in the porting process.

The transition from R News to The R Journal was always about enhancing the journal's scientific credibility, with the strategic goal of allowing researchers, especially young researchers, due credit for their work within computational statistics. The R Journal is now entering a consolidation phase, with a view to becoming a "listed journal". To do so, we need to show that we have a solid scientific standing with good editorial standards, giving submissions fair treatment and being able to publish on time. Among other things, this has taught us the concept of the "healthy backlog": You should not publish so quickly that there might be nothing to publish for the next issue!

We are still aiming at being a relatively fast-track publication, but it may be too much to promise publication of even uncontentious papers within the next two issues. The fact that we now require two reviewers on each submission is also bound to cause some delay.

Another obstacle to timely publication is that the entire work of the production of a new issue is in the hands of the editorial board, and they are generally four quite busy people. It is not good if a submission turns out to require major copy editing of its LaTeX markup and there is a new policy in place to require up-front submission of LaTeX sources and figures. For one thing, this allows reviewers to advise on the LaTeX if they can, but primarily it gives better time for the editors to make sure that an accepted paper is in a state where it requires minimal copy editing before publication. We are now able to enlist student assistance to help with this. Longer-term, I hope that it will be possible to establish a front-desk to handle submissions.

Finally, I would like to welcome our new Book Review editor, Jay Kerns. The first book review appears in this issue and several more are waiting in the wings.

IsoGene: An R Package for Analyzing Dose-response Studies in Microarray Experiments

by Setia Pramana, Dan Lin, Philippe Haldermans, Ziv Shkedy, Tobias Verbeke, Hinrich Göhlmann, An De Bondt, Willem Talloen and Luc Bijnens.

Abstract **IsoGene** is an R package for the analysis of dose-response microarray experiments to identify gene or subsets of genes with a monotone relationship between the gene expression and the doses. Several testing procedures (i.e., the likelihood ratio test, Williams, Marcus, the *M*, and Modified *M*), that take into account the order restriction of the means with respect to the increasing doses are implemented in the package. The inference is based on resampling methods, both permutations and the Significance Analysis of Microarrays (SAM).

Introduction

The exploration of dose-response relationship is important in drug-discovery in the pharmaceutical industry. The response in this type of studies can be either the efficacy of a treatment or the risk associated with exposure to a treatment. Primary concerns of such studies include establishing that a treatment has some effect and selecting a dose or doses that appear efficacious and safe (Pinheiro et al., 2006). In recent years, dose-response studies have been integrated with microarray technologies (Lin et al., 2010). Within the microarray setting, the response is gene expression measured at a certain dose level. The aim of such a study is usually to identify a subset of genes with expression levels that change with experimented dose levels.

One of four main questions formulated in dose-response studies by Ruberg (1995a, 1995b) and Chuang-Stein and Agresti (1997) is whether there is any evidence of the drug effect. To answer this question, the null hypothesis of homogeneity of means (no dose effect) is tested against ordered alternatives. Lin et al. (2007, 2010) discussed several testing procedures used in dose-response studies of microarray experiments. Testing procedures which take into account the order restriction of means with respect to the increasing doses include Williams (Williams, 1971 and 1972), Marcus (Marcus, 1976), the likelihood ratio test (Barlow et al. 1972, and Robertson et al. 1988), the *M* statistic (Hu et al., 2005) and the modified *M* statistic (Lin et al., 2007).

To carry out the analysis of dose-response microarray experiments discussed by Lin et al. (2007, 2010), an R package called **IsoGene** has been developed. The **IsoGene** package implements the testing procedures described by Lin et al. (2007) to identify a subset of genes where a monotone relationship between gene expression and dose can be detected. In this package, the inference is based on resampling methods, both permutations (Ge et al., 2003) and the Significance Analysis of Microarrays (SAM), Tusher et al., 2001. To control the False Discovery Rate (FDR) the Benjamini Hochberg (BH) procedure (Benjamini and Hochberg, 1995) is implemented.

This paper introduces the **IsoGene** package with background information about the methodology used for analysis and its main functions. Illustrative examples of analysis using this package are also provided.

Testing for Trend in Dose Response Microarray Experiments

In a microarray experiment, for each gene, the following ANOVA model is considered:

$$Y_{ij} = \mu(d_i) + \varepsilon_{ij},\ i = 0,1,\ldots,K,\ j = 1,2,\ldots,n_i, \quad (1)$$

where Y_{ij} is the jth gene expression at the ith dose level, d_i ($i = 0,1,\ldots,K$) are the K**+1** dose levels, $\mu(d_i)$ is the mean gene expression at each dose level, and $\varepsilon_{ij} \sim N(0,\sigma^2)$. The dose levels $d_0,...,d_K$ are strictly increasing.

The null hypothesis of homogeneity of means (no dose effect) is given by

$$H_0 : \mu(d_0) = \mu(d_1) = \cdots = \mu(d_K). \quad (2)$$

where $\mu(d_i)$ is the mean response at dose d_i with $i = 0,..,K$, where $i = 0$ indicates the control. The alternative hypotheses under the assumption of monotone increasing and decreasing trend of means are respectively specified by

$$H_1^{Up} : \mu(d_0) \leq \mu(d_1) \leq \cdots \leq \mu(d_K), \quad (3)$$

$$H_1^{Down} : \mu(d_0) \geq \mu(d_1) \geq \cdots \geq \mu(d_K). \quad (4)$$

For testing H_0 against H_1^{Down} or H_1^{Up}, estimation of the means under both the null and alternative hypotheses is required. Under the null hypothesis, the estimator for the mean response $\hat{\mu}$ is the overall sample mean. Let $\hat{\mu}_0^\star, \hat{\mu}_1^\star, \ldots, \hat{\mu}_K^\star$ be the maximum likelihood estimates for the means (at each dose level) under the order restricted alternatives. Barlow et al. (1972) and Robertson et al. (1988) showed that

$\hat{\mu}_0^\star, \hat{\mu}_1^\star, \ldots, \hat{\mu}_K^\star$ are given by the isotonic regression of the observed means.

In order to test H_0 against H_1^{Down} or H_1^{Up}, Lin et al. (2007) discussed five testing procedures shown in Figure 1. The likelihood ratio test ($\bar{E}_{01}^2$) (Bartholomew 1961 , Barlow et al. 1972, and Robertson et al. 1988) uses the ratio between the variance under the null hypothesis ($\hat{\sigma}_{H_0}^2$) and the variance under order restricted alternatives ($\hat{\sigma}_{H_1}^2$):

$$\Lambda_{01}^{\frac{2}{N}} = \frac{\hat{\sigma}_{H_1}^2}{\hat{\sigma}_{H_0}^2} = \frac{\sum_{ij}(y_{ij} - \hat{\mu}_j^\star)^2}{\sum_{ij}(y_{ij} - \hat{\mu})^2}. \tag{5}$$

Here, $\hat{\mu} = \sum_{ij} y_{ij} / \sum_i n_i$ is the overall mean and $\hat{\mu}_i^\star$ is the isotonic mean of the ith dose level. The null hypothesis is rejected for a "small" value of $\Lambda_{01}^{\frac{2}{N}}$. Hence, H_0 is rejected for a large value of $\bar{E}_{01}^2 = 1 - \Lambda_{01}^{\frac{2}{N}}$.

Test statistic	Formula
Likelihood Ratio Test (LRT)	$\bar{E}_{01}^2 = \frac{\sum_{ij}(y_{ij}-\hat{\mu})^2 - \sum_{ij}(y_{ij}-\hat{\mu}_i^\star)^2}{\sum_{ij}(y_{ij}-\hat{\mu})^2}$
Williams	$t = (\hat{\mu}_K^\star - \bar{y}_0)/s$
Marcus	$t = (\hat{\mu}_K^\star - \hat{\mu}_0^\star)/s$
M	$M = (\hat{\mu}_K^\star - \hat{\mu}_0^\star)/\tilde{s}$
Modified M (M′)	$M' = (\hat{\mu}_K^\star - \hat{\mu}_0^\star)\tilde{s}'$

Figure 1: Test statistics for trend test, where $s = \sqrt{2 \times \sum_{i=0}^{K} \sum_{j=1}^{n_i} (y_{ij} - \hat{\mu}_i)^2 / (n_i(n-K))}$, $\tilde{s} = \sqrt{\sum_{i=0}^{K} \sum_{j=1}^{n_i} (y_{ij} - \hat{\mu}_i^\star)^2 / (n-K)}$, $\tilde{s}' = \sqrt{\sum_{i=0}^{K} \sum_{j=1}^{n_i} (y_{ij} - \hat{\mu}_i^\star)^2 / (n-I)}$, and I is the unique number of isotonic means.

Williams (1971, 1972) proposed a step-down procedure to test for the dose effect. The tests are performed sequentially from the comparison between the isotonic mean of the highest dose and the sample mean of the control, to the comparison between the isotonic mean of the lowest dose and the sample mean of the control. The procedure stops at the dose level where the null hypothesis (of no dose effect) is not rejected. The test statistic is shown in Figure 1, where $\bar{y}_0$ is the sample mean at the first dose level (control), $\hat{\mu}_i^\star$ is the estimate for the mean at the ith dose level under the ordered alternative, n_i is the number of replications at each dose level, and s^2 is an estimate of the variance. A few years later, Marcus (1976) proposed a modification to Williams's procedure by replacing the sample mean of the control dose ($\bar{y}_0$) with the isotonic mean of the control dose ($\hat{\mu}_0^\star$).

Hu et al. (2005) proposed a test statistic (denoted by M) that was similar to Marcus' statistic, but with the standard error estimator calculated under the ordered alternatives. This is in contrast to Williams' and Marcus' approaches that used the within group sum of squares under the unrestricted means. Moreover, Hu et al. (2005) used $n - K$ as the degrees of freedom. However, the unique number of isotonic means is not fixed, but changes across genes. For that reason, Lin et al. (2007) proposed a modification to the standard error estimator used in the M statistic by replacing it with $(n - I)$ as the degrees of freedom, where I is the unique number of isotonic means for a given gene. The five test statistics discussed above are based on the isotonic regression of the observed means. Estimation of isotonic regression requires the knowledge of the direction of the trend (increasing/decreasing). In practice, the direction is not known in advance. Following Barlow et al. (1972), the **IsoGene** package calculates the likelihood of the isotonic trend for both directions and chooses the direction for which the likelihood is maximized.

The Significance Analysis of Microarrays procedure (SAM, Tusher et al., 2001) can be also adapted to the five test statistics described above. The generic algorithm of SAM discussed by Chu et al. (2001) is implemented in this package. For the t-type test statistics (i.e., Williams, Marcus, the M, and the M'), a fudge factor is added in the standard error of the mean difference. For example, the SAM regularized test statistic for the M' is given by,

$$M'^{SAM} = \frac{\hat{\mu}_K^\star - \hat{\mu}_0^\star}{\tilde{s}' + s_0}, \tag{6}$$

where s_0 is the fudge factor and is estimated from the percentiles of standard errors of all genes which minimize the Coefficient of Variation (CV) of the Median Absolute Deviation (MAD) of the SAM regularized test statistics. For the F-type test statistic, such as $\bar{E}_{01}^2$, the SAM regularized test statistic is defined by,

$$\bar{E}_{01}^{2SAM} = \frac{\sqrt{\hat{\sigma}_{H_0}^2 - \hat{\sigma}_{H_1}^2}}{\sqrt{\hat{\sigma}_{H_0}^2} + s_0}. \tag{7}$$

Multiplicity

In this study, a gene-by-gene analysis is carried out. When many hypotheses are tested, the probability of making the type I error increases sharply with the number of hypotheses tested. Hence, multiplicity adjustments need to be performed. Dudoit et al. (2003) and Dudoit and van der Laan (2008) provided extensive discussions on the multiple adjustment procedures in genomics. Lin et al. (2007) compared several approaches for multiplicity adjustments, and showed the application in dose-response microarray experiments. In the **IsoGene** package the inference is based on resampling-based methods. The Benjamini Hochberg (BH) procedure is used to control the FDR . We use the definition of FDR from Benjamini and Hochberg (1995). The Significance Analysis of Microarrays (SAM, Tusher et al., 2001) approach is also considered, which estimates the FDR by using permutations, where the FDR is computed as median of the number of falsely called genes divided by the number of genes called significant.

Introduction to IsoGene Package

The **IsoGene** package provides the estimates and *p*-values of five test statistics for testing for monotone trends discussed in the previous section. The *p*-values are calculated based on a resampling procedure in which the distribution of the statistic under the null hypothesis is approximated using permutations.

The main functions of the **IsoGene** package are `IsoRawp()` and `IsoTestBH()` which calculate the raw *p*-values using permutations and adjust them using the Benjamini-Hochberg (BH-FDR, Benjamini and Hochberg, 1995) and Benjamini-Yekutieli (BY-FDR, Benjamini and Yekutieli, 2001) procedures. The **IsoGene** package also implements the Significance Analysis of Microarrays (SAM) by using function `IsoTestSAM()`.

Function	Description
`IsoRawp()`	Calculates raw p-values for each test statistic using permutations
`IsoTestBH()`	Adjusts raw p-values of the five test statistics using the BH- or BY-FDR procedure
`IsoGene1()`	Calculates the values of the five test statistics for a single gene
`IsoGenem()`	Calculates the values of the five test statistics for all genes
`IsoPlot()`	Plots the data points, sample means at each dose and an isotonic regression line (optional)
`IsopvaluePlot()`	Plots the p^{Up} and p^{Down}-values of a gene for a given test
`IsoBHplot()`	Plots the raw p-values and adjusted BH- and BY-FDR *p*-values
`IsoGenemSAM()`	Calculates the values for the five SAM regularized test statistics
`IsoTestSAM()`	Obtains the list of significant genes using the SAM procedure

Figure 2: The main **IsoGene** package functions.

The supporting functions are `IsoGenem()` and `IsoGenemSAM()`, which calculate the values for the five test statistics and for five SAM regularized test statistics, respectively. The functions `IsopvaluePlot()`, `IsoBHplot()`, and `IsoPlot()` can be used to display the data and show the results of the testing procedures. The summary of the functions and their descriptions are presented in Figure 2.

The **IsoGene** package can be obtained from CRAN: http://cran.r-project.org/package=IsoGene. The **IsoGene** package requires the **ff** and **Iso** packages.

Example 1: Data Exploration

To illustrate the analysis of dose-response in microarray using **IsoGene** package, we use the dopamine data. The data were obtained from a pre-clinical evaluation study of an active compound (Göhlmann and Talloen, 2009). In this study the potency of the compound was investigated. The compound had 6 dose levels (0, 0.01, 0.04, 0.16, 0.63, 2.5 mg/kg) and each dose was given to four to five independent rats. The experiment was performed using Affymetrix whole-human genome chip. There are 26 chips/arrays and each chip/array contains 11,562 probe sets (for simplicity, we refer to the probe sets as genes). The dopamine data set with 1000 genes is provided inside the package as example data. The complete data set can be obtained on request upon the first author. For this paper, the analysis is based on the original data set (using all genes).

The example data is in an `ExpressionSet` object called `dopamine`. More detailed explanation of the `ExpressionSet` object is discussed by Falcon et al. (2007). In order to load the object into **R**, the following code can be used:

```
> library(affy)
> library(IsoGene)
> data(dopamine)
> dopamine
ExpressionSet(storageMode:lockedEnvironment)
assayData: 11562 features, 26 samples
  element names: exprs
phenoData
  sampleNames: X1, X2, ..., X26 (26 total)
  varLabels and varMetadata description:
    dose: Dose Levels
featureData
  featureNames: 201_at,202_at,...,11762_at
   (11562 total)
  fvarLabels and fvarMetadata
    description: none
experimentData: use 'experimentData(object)'
```

The **IsoGene** package requires the information of dose levels and gene expression as input. The information of dose levels and the log2 transformed gene intensities can be extracted using the following code:

```
> express <- data.frame(exprs(dopamine))
> dose <- pData(dopamine)$dose
```

For data exploration, the function `IsoPlot()` can be used to produce a scatterplot of the data. The function `IsoPlot()` has two options to specify the dose levels, i.e., `type ="ordinal"` or `"continuous"`. By default, the function produces the plot with continuous dose levels and data points. To add an isotonic regression line, one can specify `add.curve=TRUE`.

Plots of the data and an isotonic regression line for one of the genes in the data set with dose on the continuous and ordinal scale can be produced by using the following code:

```
# Figure 3
> par(mfrow=c(1,2))
> IsoPlot(dose,express[56,],type="continuous",
+ add.curve=TRUE)
> IsoPlot(dose,express[56,],type="ordinal",
+ add.curve=TRUE)
```

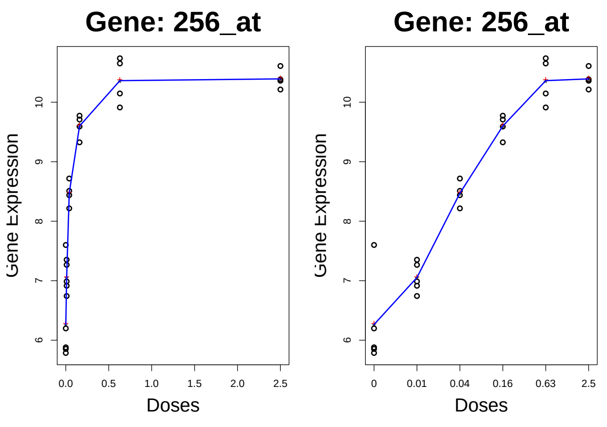


Figure 3: *The plots produced by* `IsoPlot()` *with dose as continuous variable (left panel) and dose as ordinal variable (right panel and the real dose level is presented on the* x*-axis). The data points are plotted as circles, while the sample means as red crosses and the fitted increasing isotonic regression model as a blue solid line.*

Example 2: Resampling-based Multiple Testing

In this section, we illustrate an analysis to test for monotone trend using the five statistics presented in section 2 using the **IsoGene** package. The function `IsoRawp()` is used to calculate raw p-values based on permutations. The dose levels, the data frame of the gene expression, and the number of permutations used to approximate the null distribution need to be specified in this function. Note that, since the permutations are generated randomly, one can use function `set.seed` to obtain the same random number. To calculate raw p-values for the `dopamine` data using 1000 permutations with `seed=1234`, we can use the following code:

```
> set.seed(1234)
> rawpvalue <- IsoRawp(dose, express,
+ niter=1000)
```

The output object `rawpvalue` is a list with four components containing the p-values for the five test statistics: two-sided p-values, one-sided p-values, p^{Up}-values and p^{Down}-values. The codes to extract the component containing two-sided p-values for the first four genes are presented below. In a similar way, one can obtain other components.

```
> twosided.pval <- rawpvalue[[2]]
> twosided.pval[1:4, ]
  Probe.ID    E2 Williams Marcus     M  ModM
1   201_at 1.000    0.770  0.996 0.918 0.946
2   202_at 0.764    0.788  0.714 0.674 0.700
3   203_at 0.154    0.094  0.122 0.120 0.134
4   204_at 0.218    0.484  0.378 0.324 0.348
```

Once the raw p-values are obtained, one needs to adjust these p-values for multiple testing. The function `IsoTestBH()` is used to adjust the p-values while controlling for the FDR. The raw p-values (e.g., `rawp`), FDR level, type of multiplicity adjustment (BH-FDR or BY-FDR) and the statistic need to be specified in following way:

```
IsoTestBH(rawp, FDR=c(0.05,0.1),
type=c("BH","BY"), stat=c("E2",
"Williams","Marcus","M","ModifM"))
```

`IsoTestBH()` produces a list of genes, which have a significant increasing/decreasing trend. The following code can be used to adjust the two-sided p-values of the $\bar{E}^2_{01}$ using the BH-FDR adjustment:

```
> E2.BH <- IsoTestBH(twosided.pval,
+ FDR = 0.05, type = "BH", stat ="E2")
> dim(E2.BH)
[1] 250   4
```

The object `E2.BH` contains a list of significant genes along with the probe ID, the corresponding row numbers of the gene in the original data set, the unadjusted (raw), and the BH-FDR adjusted p-values of the $\bar{E}^2_{01}$ test statistic. In the `dopamine` data, there are 250 genes tested with a significant monotone trend using the likelihood ratio test ($\bar{E}^2_{01}$) at the 0.05 FDR level. The first four significant genes are listed below:

```
> E2.BH[1:4, ]
  Probe.ID row.num raw p-values BH adj.p-values
1 256_at    56            0              0
2 260_at    60            0              0
3 280_at    80            0              0
4 283_at    83            0              0
```

One can visualize the number of significant findings for the BH-FDR and BY-FDR procedures for a given test statistic using the `IsoBHPlot()` function.

```
> # Figure 4
> IsoBHPlot(twosided.pval,FDR=0.05,stat="E2")
```

Figure 4 shows the raw p-values (solid blue line), the BH-FDR adjusted p-values (dotted and dashed red line) and BY-FDR (dashed green line) adjusted p-values for the $\bar{E}^2_{01}$.

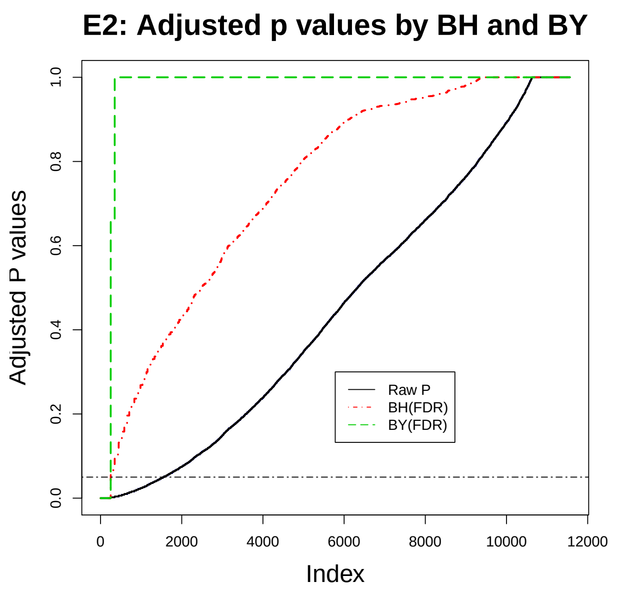


Figure 4: *Plot of unadjusted, BH-FDR and BY-FDR adjusted p-values for* $\bar{E}^2_{01}$

Example 3: Significance Analysis of Dose-response Microarray Data (SAM)

The Significance Analysis of Microarrays (SAM) for testing for the dose-response relationship under order restricted alternatives is implemented in the **IsoGene** package as well. The function `IsoTestSAM()` provides a list of significant genes based on the SAM procedure.

```
> IsoTestSAM(x, y, fudge=c("none","pooled"),
+ niter=100, FDR=0.05, stat=c("E2",
+ "Williams","Marcus","M","ModifM"))
```

The input for this function is the dose levels (`x`), gene expression (`y`), number of permutations (`niter`), the FDR level, the test statistic, and the option of using fudge factor. The option `fudge` is used to specify the calculation the fudge factor in the SAM regularized test statistic. If the option `fudge ="pooled"` is used, the fudge factor will be calculated using the method described in the SAM manual (Chu et al., 2001). If we specify `fudge ="none"` no fudge factor is used.

The following code is used for analyzing the `dopamine` data using the SAM regularized M'-test statistic:

```
> set.seed(1235)
> SAM.ModifM <-  IsoTestSAM(dose, express,
+ fudge="pooled", niter=100,
+ FDR=0.05, stat="ModifM")
```

The resulting object `SAM.ModifM`, contains three components:

1. `sign.genes1` contains a list of genes declared significant using the SAM procedure.
2. `qqstat` gives the SAM regularized test statistics obtained from permutations.
3. `allfdr` provides a delta table in the SAM procedure for the specified test statistic.

To extract the list of significant gene, one can do:

```
> SAM.ModifM.result <- SAM.ModifM[[1]]
> dim(SAM.ModifM.result)
[1] 151   6
```

The object `SAM.ModifM.result`, contains a matrix with six columns: the Probe IDs, the corresponding row numbers of the genes in the data set, the observed SAM regularized M' test statistics, the q-values (obtained from the lowest False Discovery Rate at which the gene is called significant), the raw p-values obtained from the joint distribution of permutation test statistics for all of the genes as described by Storey and Tibshirani (2003), and the BH-FDR adjusted p-values.

For `dopamine` data, there are 151 genes found to have a significant monotone trend based on the SAM regularized M' test statistic with the FDR level of 0.05. The SAM regularized M' test statistic values along with q-values for the first four significant genes are presented below.

```
> SAM.ModifM.result[1:4,]

  Probe.ID row.num  stat.val qvalue    pvalue adj.pvalue
1  4199_at    3999 -4.142371 0.0000 3.4596e-06 0.0012903
2  4677_at    4477 -3.997728 0.0000 6.9192e-06 0.0022857
3  7896_at    7696 -3.699228 0.0000 1.9027e-05 0.0052380
4  9287_at    9087 -3.324213 0.0108 4.4974e-05 0.0101960
```

Note that genes are sorted increasingly based on the SAM regularized M' test statistic values (i.e., `stat.val`).

In the **IsoGene** package, the function `IsoSAMPlot()` provides graphical outputs of the SAM procedure. This function requires the SAM regularized test statistic values and the delta table in the SAM procedure, which can be obtained from the resulting object of the `IsoTestSAM()` function, which in this example data is called `SAM.ModifM`. To extract the objects we can use the following code:

```
# Obtaining SAM regularized test statistic
qqstat <- SAM.ModifM[[2]]
# Obtaining delta table
allfdr <- SAM.ModifM[[3]]
```

We can also obtain the `qqstat` and `allfdr` from the functions `Isoqqstat()` and `Isoallfdr()`, respectively. The code for the two functions are:

```
Isoqqstat(x, y, fudge=c("none","pooled"),niter)
Isoallfdr(qqstat, ddelta, stat=c("E2",
"Williams","Marcus","M","ModifM"))
```

The examples of using the functions `Isoqqstat()` and `Isoallfdr()` are as follows:

```
# Obtaining SAM regularized test statistic
> qqstat <- Isoqqstat(dose, express,
+ fudge="pooled", niter=100)
# Obtaining delta table
> allfdr <- Isoallfdr(qqstat, ,stat="ModifM")
```

Note that in `Isoallfdr()`, `ddelta` is left blank, with default values taken from the data, i.e., all the percentiles of the standard errors of the M' test statistic.

The two approaches above will give the same result. Then to produces the SAM plot for the SAM regularized M' test statistic, we can use the function `IsoSAMPlot`:

```
# Figure 5
> IsoSAMPlot(qqstat, allfdr, FDR = 0.05,
+ stat = "ModifM")
```

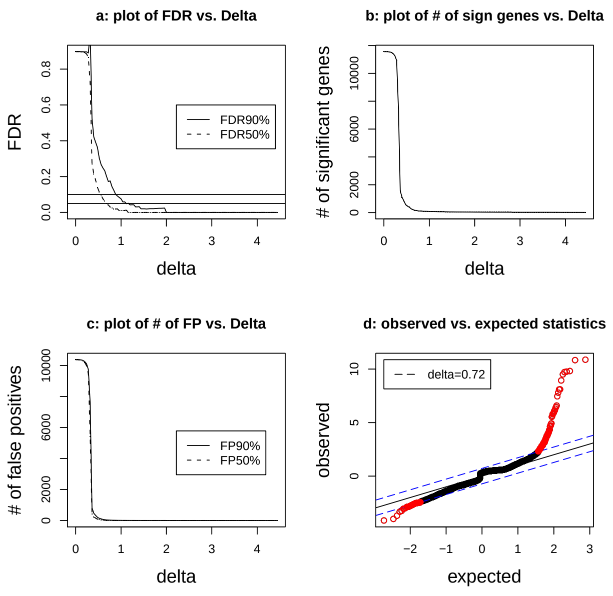


Figure 5: *The SAM plots: a.Plot of the FDR vs.* Δ*; b. Plot of number of significant genes vs.* Δ*; c. Plot of number of false positives vs.* Δ*; d. Plot of the observed vs. expected test statistics.*

Panel *a* of Figure 5 shows the FDR (either 50% or 90% (more stringent)) vs. Δ, from which, user can choose the Δ value with the corresponding desired FDR. The FDR 50% and FDR 90% are obtained from, respectively, the median and 90th percentile of the number of falsely called genes (number of false positives) divided by the number of genes called significant, which are estimated by using permutations (Chu et al., 2001). Panel *b* shows the number of significant genes vs. Δ, and panel *c* shows the number of false positives (either obtained from 50th or 90th percentile) vs. Δ. Finally, panel *d* shows the observed vs. the expected (obtained from permutations) test statistics, in which the red dots are those genes called differentially expressed.

Comparison of the Results of Resampling-based Multiple Testing and the SAM

In the previous sections we have illustrated analysis for testing a monotone trend for dose-response microarray data by using permutations and the SAM. The same approach can also be applied to obtain a list of significant genes based on other statistics and other multiplicity adjustments. Figure 6 presents the number of genes that are found to have a significant monotone trend using five different statistics with the FDR level of 0.05 using the two approaches.

It can be seen from Figure 6 that the $\bar{E}^2_{01}$ gives a higher number of significant genes than other t-type statistics. Adding the fudge factor to the statistics leads to a smaller number of significant genes using the SAM procedure for the t-type test statistics as compared to the BH-FDR procedure.

	Number of significant genes		
Test statistic	BH-FDR	SAM	# common genes
$\bar{E}^2_{01}$	250	279	200
Williams	186	104	95
Marcus	195	117	105
M	209	158	142
M'	203	151	134

Figure 6: Number of significant genes for each test statistic with BH-FDR and SAM.

In the inference based on the permutations (BH-FDR) and the SAM, most of the genes found by the five statistics are in common (see Figure 6). The plots of the samples and the isotonic trend of four best genes that are found in all statistics and in both the permutations and the SAM approaches are presented in Figure 7, which have shown a significant increasing monotone trend.

Discussion

We have shown the use of the **IsoGene** package for dose-response studies within a microarray setting along with the motivating examples. For the analysis using the SAM procedure, it should be noted that the fudge factor in the $\bar{E}^2_{01}$ is obtained based on the approach for F-type test statistic discussed by Chu et al. (2001) and should be used with caution. The performance of such an adjustment as compared to the t-type test statistics has not yet been investigated in terms of power and control of the FDR. Therefore, it is advisable to use the fudge factor in the t-type test statistics, such as the M and modified M test statistics.

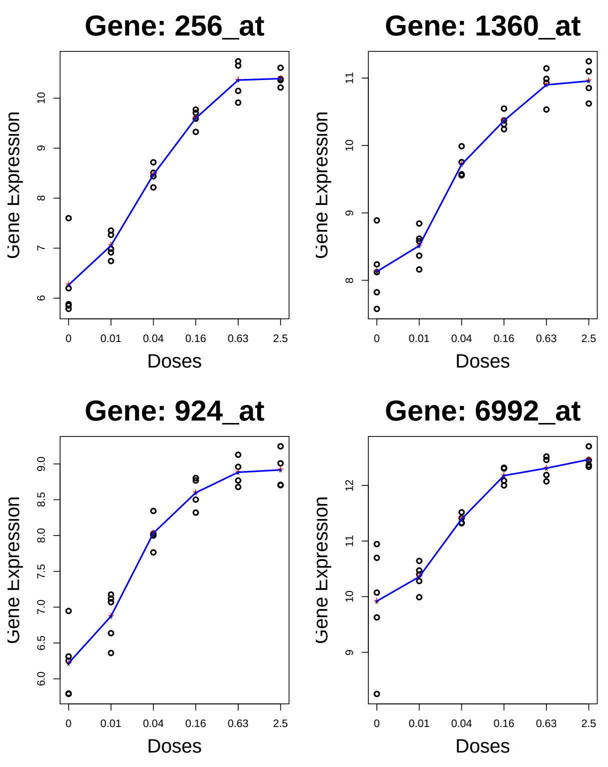


Figure 7: *Plots of the samples (circles) and the isotonic trend (solid blue line) for the four best genes with a significant monotone trend.*

The calculation of the raw *p*-values based on permutations using the `IsoRawp()` function is computationally intensive. For example, when we used 100 permutations for analyzing 10,000 genes, it takes around 30 minutes. When we use 1,000 permutations, the elapsed time increases around 10 times. Usually to approximate the null distribution of the test statistics, a large number of permutations is needed, which leads to the increase of computation time. Considering the computation time, we also provide the *p*-values obtained from joint distribution of permutation statistics for all genes which is implemented in function `IsoTestSAM()`. This approach is sufficient to obtain small *p*-values using a small number of permutations. Alternatively, one can also use the SAM procedure which does not require a large number of permutations and takes less computation time. However caution should be drawn to the comparison of the SAM procedure with and without the fudge factor.

A further development of this package is the Graphical User Interface (GUI) using `tcl/tk` for users without knowledge of R programming.

Acknowledgement

Financial support from the IAP research network nr P5/24 of the Belgian Government (Belgian Science Policy) is gratefully acknowledged.

Bibliography

R. Barlow, D. Bartholomew, M. Bremner, and H. Brunk. *Statistical Inference Under Order Restriction*. New York: Wiley, 1972.

D. Bartholomew. Ordered tests in the analysis of variance. *Biometrika*, 48:325–332, 1961.

Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. R. Statist. Soc. B*, 57:289–300, 1995.

Y. Benjamini and D. Yekutieli. The control of the false discovery rate in multiple testing under dependency. *ANN STAT*, 29 (4):1165–1188, 2001.

G. Chu, B. Narasimhan, R. Tibshirani, and V. Tusher. SAM "Significance Analysis of Microarrays" users guide and technical document, 2001. URL `http://www-stat.stanford.edu/~tibs/SAM/`.

C. Chuang-Stein and A. Agresti. Tutorial in biostatistics: A review of tests for detecting a monotone dose-response relationship with ordinal response data. *Statistics in Medicine*, 16:2599–2618, 1997.

S. Dudoit and M. J. van der Laan. *Multiple Testing Procedures with Applications to Genomics*. Springer, 2008.

S. Dudoit, J. P. Shaffer, and J. Boldrick. Multiple hypothesis testing in microarray experiments. *Statistical Science*, 18 (1):71–103, 2003.

S. Falcon, M. Morgan, and R. Gentleman. An introduction to BioConductor's ExpressionSet class. a BioConductor vignette. *Bioconductor Vignettes*, URL `http://bioconductor.org/packages/2.3/bioc/vignettes/Biobase/inst/doc/ExpressionSetIntroduction.pdf`, 2007.

Y. Ge, S. Dudoit, and T. P. Speed. Resampling- based multiple testing for microarray data analysis. *Test*, 12 (1):1–77, 2003.

H. Göhlmann and W. Talloen. *Gene Expression Studies Using Affymetrix Microarrays*. Chapman & Hall/CRC, 2009.

J. Hu, M. Kapoor, W. Zhang, S. Hamilton, and K. Coombes. Analysis of dose response effects on gene expression data with comparison of two microarray platforms. *Bioinformatics*, 21(17):3524–3529, 2005.

D. Lin, Z. Shkedy, D. Yekutieli, T. Burzykowski, H. Göhlmann, A. De Bondt, T. Perera, T. Geerts, and L. Bijnens. Testing for trends in dose-response microarray experiments: A comparison of several testing procedures, multiplicity and resampling-based inference. *Statistical Applications in Genetics and Molecular Biology*, 6(1):Article 26, 2007.

D. Lin, Z. Shkedy, D. Yekutieli, D. Amaratunga, and L. Bijnens., editors. *Modeling Dose-response Microarray Data in Early Drug Development Experiments Using R*. Springer, (to be published in 2010).

R. Marcus. The powers of some tests of the quality of normal means against an ordered alternative. *Biometrika*, 63:177–83, 1976.

J. Pinheiro, F. Bretz, and M. Branson. *Analysis of dose response studies: Modeling approaches. pp. 146-171 in Ting, N. (ed.) Dose Finding in Drug Development*. Springer, New York, 2006.

T. Robertson, F. Wright, and R. Dykstra. *Order Restricted Statistical Inference*. Wiley, 1988.

S. Ruberg. Dose response studies. i. some design considerations. *J. Biopharm. Stat*, 5(1):1–14, 1995a.

S. Ruberg. Dose response studies. ii. analysis and interpretation. *J. Biopharm. Stat*, 5(1):15–42, 1995b.

J. D. Storey and R. Tibshirani. Statistical significance for genomewide studies. *Proceedings of the National Academy of Sciences*, 100 no. 16:9440–9445, 2003.

V. Tusher, R. Tibshirani, and G. Chu. Significance analysis of microarrays applied to the ionizing radiation response. *Proceedings of the National Academy of Sciences*, 98:5116–5121, 2001.

D. Williams. A test for differences between treatment means when several dose levels are compared with a zero dose control. *Biometrics*, 27:103–117, 1971.

D. Williams. The comparison of several dose levels with a zero dose control. *Biometrics*, 28:519–531, 1972.

Setia Pramana, Dan Lin, Philippe Haldermans and Ziv Shkedy
Interuniversity Institute for Biostatistics and Statistical Bioinformatics, CenStat, Universiteit Hasselt,
Agoralaan 1, B3590 Diepenbeek, Belgium
setia.pramana@uhasselt.be
dan.lin@uhasselt.be
philippe.haldermans@uhasselt.be
ziv.shkedy@uhasselt.be

Tobias Verbeke
OpenAnalytics BVBA
Kortestraat 30A, 2220 Heist-op-den-Berg, Belgium
tobias.verbeke@openanalytics.be

Hinrich Göhlmann, An De Bondt, Willem Talloen and Luc Bijnens
Johnson and Johnson Pharmaceutical Research and Development, a division of Janssen Pharmaceutica,
Beerse, Belgium
HGOEHLMA@PRDBE.jnj.com
ADBONDT@its.jnj.com
WTALLOEN@its.jnj.com
LBIJNENS@its.jnj.com

MCMC for Generalized Linear Mixed Models with glmmBUGS

by Patrick Brown and Lutong Zhou

Abstract The **glmmBUGS** package is a bridging tool between Generalized Linear Mixed Models (GLMMs) in R and the BUGS language. It provides a simple way of performing Bayesian inference using Markov Chain Monte Carlo (MCMC) methods, taking a model formula and data frame in R and writing a BUGS model file, data file, and initial values files. Functions are provided to reformat and summarize the BUGS results. A key aim of the package is to provide files and objects that can be modified prior to calling BUGS, giving users a platform for customizing and extending the models to accommodate a wide variety of analyses.

Introduction

There are two steps in model fitting that are much more time consuming in WinBUGS than in R. First, unbalanced multi-level data needs to be formatted in a way that BUGS can handle, including changing categorical variables to indicator variables. Second, a BUGS model file must be written. The **glmmBUGS** package addresses these issues by allowing users to specify models as formulas in R, as they would in the `glm` function, and provides everything necessary for fitting the model with WinBUGS or OpenBUGS via the **R2WinBUGS** package (Sturtz et al., 2005).

The **glmmBUGS** package creates the necessary BUGS model file, starting value function, and suitably formatted data. Improved chain mixing is accomplished with a simple reparametrization and the use of sensible starting values. Functions are provided for formatting and summarizing the results. Although a variety of models can be implemented entirely within **glmmBUGS**, the package intends to provide a basic set of data and files for users to modify as necessary. This allows the full flexibility of BUGS model specification to be exploited, with much of the initial "grunt work" being taken care of by **glmmBUGS**.

Examples

Independent random effects

Consider the bacteria data from the **MASS** package:

```
> library(MASS)
> data(bacteria)
> head(bacteria)
```

```
  y ap hilo week  ID     trt
1 y  p   hi    0 X01 placebo
2 y  p   hi    2 X01 placebo
3 y  p   hi    4 X01 placebo
4 y  p   hi   11 X01 placebo
5 y  a   hi    0 X02   drug+
6 y  a   hi    2 X02   drug+
```

The variables to be considered are: `y`, the presence or absence of bacteria in a sample coded as `"y"` and `"n"` respectively; `week`, the time of the observation; `ID`, the subject identifier; and `trt` giving the treatment group as `"placebo"`, `"drug"`, or `"drug+"`.

A generalized linear mixed model is applied to the data with:

$$\begin{aligned} Y_{ij} &\sim \text{Bernoulli}(p_{ij}) \\ \text{logit}(p_{ij}) &= \mu + x_{ij}\beta + V_i \\ V_i &\sim iid\ \text{N}(0,\sigma^2) \end{aligned} \tag{1}$$

where: Y_{ij} indicates the presence or absence of bacteria of the ith person at week j; covariates x_{ij} are week and indicator variables for treatment; p_{ij} denotes the probability of bacteria presence; V_i is the random effect for the ith patient, assumed to be normal with mean 0 and variance σ^2. To improve the mixing of the Markov chain, a reparametrized model is fitted with:

$$\begin{aligned} Y_{ij} &\sim \text{Bernoulli}(p_{ij}) \\ \text{logit}(p_{ij}) &= R_i + w_{ij}\gamma \\ R_i &\sim \text{N}(\mu + g_i\alpha,\sigma^2). \end{aligned} \tag{2}$$

Here g_i is the (indicator variable for) treatment group for subject i and w_{ij} is the week that observation j was taken. Note that the two models are the same, with $V_i = R_i - \mu - g_i\alpha$, $\beta = (\gamma,\alpha)$ and $x_{ij} = (w_{ij},g_i)$. The model in (1) has strong negative dependence between the posterior samples of V_i and μ, whereas R_i and μ in (2) are largely independent.

As BUGS only allows numeric data, and cannot have the '+' sign in variable names, the data are recoded as follows:

```
> bacterianew <- bacteria
> bacterianew$yInt = as.integer(bacterianew$y ==
+     "y")
> levels(bacterianew$trt) <- c("placebo",
+     "drug", "drugplus")
```

The primary function in the package is `glmmBUGS`, which does the preparatory work for fitting the model in (2) with:

```
> library(glmmBUGS)
> bacrag <- glmmBUGS(formula = yInt ~
+     trt + week, data = bacterianew,
+     effects = "ID", modelFile = "model.bug",
+     family = "bernoulli")
```

This specifies yInt as a Bernoulli-valued response, trt and week as fixed-effect covariates, and the "ID" column for the random effects. The result is a list with three elements:

ragged is a list containing the data to be passed to WinBUGS;

pql is the result from fitting the model with the glmmPQL function in **MASS** package.

startingValues is a list of starting values for the parameters and random effects, which is obtained from the glmmPQL result.

In addition, two files are written to the working directory

'model.bug' is the BUGS model file

'getInits.R' contains the R code for a function to generate random starting values.

To accommodate unbalanced designs, the data are stored as ragged arrays, as described in the section "Handling unbalanced datasets" of the WinBUGS manual (Spiegelhalter et al., 2004). The ragged result has a vector element "SID" indicating the starting position of each individual's observations in the dataset. The covariates are split into elements "XID" and "Xobservations" for the individual-level and observation-level covariates respectively:

```
> names(bacrag$ragged)

[1] "NID"          "SID"
[3] "yInt"         "XID"
[5] "Xobservations"
```

The model file consists of an outer loop over ID levels and an inner loop over observations. The details of how the model is implemented are best understood by examining the 'model.bug' file and consulting the WinBUGS manual.

At this stage the user is expected to modify the files and, if necessary, the data to refine the model, set appropriate priors, and ensure suitability of starting values. Before using the bugs function in the **R2WinBUGS** package to sample from the posterior, the starting value function getInits must be prepared. First, the file 'getInits.R' (which contains the source for getInits) should be edited and sourced into R. Next, the list containing the PQL-derived starting values must be assigned the name startingValues, as getInits will be accessing this object every time it is run.

```
> source("getInits.R")
> startingValues = bacrag$startingValues
> library(R2WinBUGS)
```

```
> bacResult = bugs(bacrag$ragged, getInits,
+     model.file = "model.bug", n.chain = 3,
+     n.iter = 2000, n.burnin = 100,
+     parameters.to.save = names(getInits()),
+     n.thin = 10)
```

Post WinBUGS commands

After running WinBUGS, a series of functions in the **glmmBUGS** package can be used to manipulate and check the simulation results. The restoreParams function is used to restore the original parametrization from (1) and assign the original names to the group or subject identifiers.

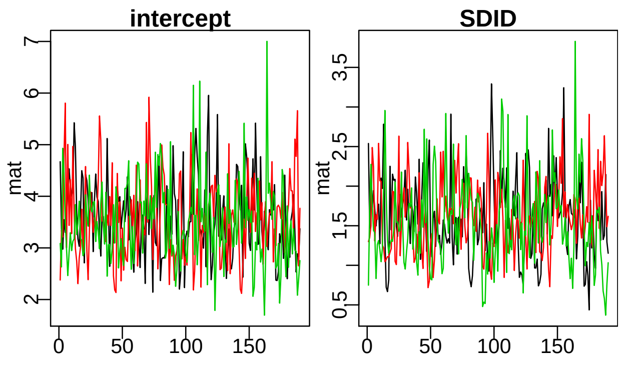
```
> bacParams = restoreParams(bacResult,
+     bacrag$ragged)
> names(bacParams)

[1] "intercept" "SDID"      "deviance"
[4] "RID"       "betas"
```

The result is a set of posterior samples for μ (intercept), σ (SDID), all the V_i (RID), and β (betas). Posterior means and credible intervals are obtained with the summaryChain function.

```
> bacsummary = summaryChain(bacParams)
> names(bacsummary)

[1] "scalars" "RID"     "betas"

> signif(bacsummary$betas[, c("mean",
+     "2.5%", "97.5%")], 3)

               mean   2.5%   97.5%
observations -0.159 -0.263 -0.0614
trtdrug      -1.500 -3.240  0.0164
trtdrugplus  -0.920 -2.740  0.6060

> bacsummary$scalars[, c("mean", "2.5%",
+     "97.5%")]

              mean     2.5%    97.5%
intercept 3.547556 2.243375 5.219125
SDID      1.597761 0.737010 2.716975
```

In order to check the convergence of the simulations, we use the checkChain function to produce trace plots of the intercept and σ parameters.

```
> checkChain(bacParams, c("intercept",
+     "SDID"))
```



A spatial example

The `ontario` dataset contains expected and observed cases of molar cancer by census sub-division in Ontario, Canada.

```
> data(ontario)
> head(ontario)
```

```
          CSDUID observed logExpected
3501005 3501005       61    2.118865
3501011 3501011       50    1.971265
3501012 3501012      117    3.396444
3501020 3501020       65    1.919814
3501030 3501030       37    1.779957
3501042 3501042       16    1.182329
```

Consider the following model (see Rue and Held, 2005):

$$\begin{aligned} Y_i &\sim \text{Poisson}(\lambda_i E_i) \\ \log(\lambda_i) &= \mu + U_i + V_i \\ U_i &\sim \text{GMRF}(\sigma_U^2) \\ V_i &\sim \text{iid N}(0, \sigma_V^2) \end{aligned} \tag{3}$$

where Y_i and E_i are the observed and expected number of cancer cases in census division i; λ_i is the cancer incidence rate; the U_i are spatial random effects, a Gaussian Markov random field with variance σ_U^2; and V_i are spatially independent random effects with variance σ_V^2.

This model is reparametrised with $R_i = U_i + \mu + V_i$ in place of V_i. An adjacency matrix is required for the spatial random effect. This was computed from the spatial boundary files with the `poly2nb` function from the **spdep** package and is stored as the object `popDataAdjMat`. The model can be fitted with `glmmBUGS` as follows:

```
> data(popDataAdjMat)
> forBugs = glmmBUGS(formula = observed +
+     logExpected ~ 1, spatial = popDataAdjMat,
+     effects = "CSDUID", family = "poisson",
+     data = ontario)
```

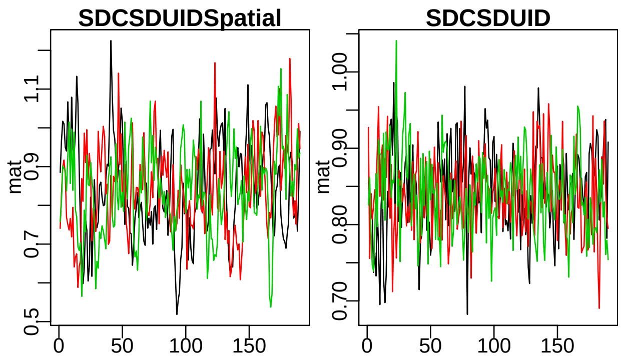
Note that the expected counts are needed on the log scale, and are passed as a second argument on the left side of the model equation to denote their being offset parameters without a coefficient. The random effect is at the census sub-division level (`CSDUID`), with `popDataAdjMat` giving the dependency structure. Posterior samples can be generated as follows:

```
> startingValues = forBugs$startingValues
> source("getInits.R")
> onResult = bugs(forBugs$ragged, getInits,
+     model.file = "model.bug", n.chain = 3,
+     n.iter = 2000, n.burnin = 100,
+     parameters.to.save = names(getInits()),
+     n.thin = 10)
> ontarioParams = restoreParams(onResult,
+     forBugs$ragged)
> names(ontarioParams)
```

```
[1] "intercept"          "SDCSDUID"
[3] "SDCSDUIDSpatial"    "deviance"
[5] "RCSDUID"            "RCSDUIDSpatial"
[7] "FittedRateCSDUID"
```

There are posterior simulations for two variance parameters, σ_U (`SDCSDUIDSPATIAL`) and σ_V (`SDCSDUID`). Samples for the random effects are given for U_i (`RCSDUIDSpatial`), $U_i + V_i$ (`RCSDUID`), and λ_i (`FittedRateCSDUID`). Trace plots are shown below for σ_U and σ_V, the standard deviations of the spatial and non-spatial random effects.

```
> checkChain(ontarioParams, c("SDCSDUIDSpatial",
+     "SDCSDUID"))
```



These trace plots show poor mixing, and the Markov chain should be re-run with longer chains and more thinning.

The posterior means for the relative risk λ_i for Toronto and Ottawa (Census Subdivisions 3506008 and 3530005) are extracted with:

```
> ontarioSummary = summaryChain(ontarioParams)
> ontarioSummary$FittedRateCSDUID[c("3506008",
+     "3520005"), "mean"]
```

```
    3506008     3520005
0.119725007 0.008148696
```

The posterior probability of each region having a relative risk twice the provincial average is computed with

```
> postProb = apply(ontarioParams$RCSDUID,
+     3, function(x) mean(x > log(2)))
```

The `"RCSDUID"` element of `ontarioParams` is a three dimensional array, with entry i,j,k being iteration i of chain j for region k. The `apply` statement computes the proportion of log-relative risks over log(2) for each region (the third dimension).

The results can be merged into the original spatial dataset, as described in the documentation for the **diseasemapping** package. Figure 1 shows a map of the posterior probabilities.

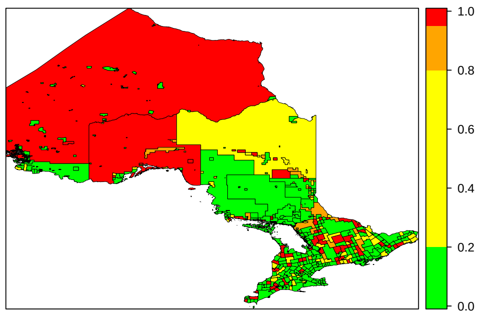


Figure 1: Posterior probabilities of Ontario census sub-divisions having twice the provincial average risk of molar cancer.

Additional features

The `reparam` argument to the `glmmBUGS` function reparametrizes the intercept, and a set of covariates for a new baseline group is provided. Recall that in the bacteria example the `ragged` object passed to WinBUGS divides covariates into observation and ID level effects. `ragged$Xobservations` is a vector and `ragged$XID` is a matrix of two columns, which contains the week variable and indicator variables for the treatments respectively. The `reparam` argument should accordingly be a list with one or more of elements `"observations"` containing the week or `"ID"` containing the two treatment indicators. Setting `reparam=list(observations=2)` would make week 2 the baseline week, rather than week zero. This argument is useful when using spline functions as covariates, as the intercept parameter which WinBUGS updates can be set to the curve evaluated in the middle of the dataset.

The `prefix` argument allows a character string to precede all data and variable names. This can be used to combine multiple model files and datasets into a single BUGS model, and modifying these files to allow dependence between the various components. For instance, one random effects model for blood pressure could be given `prefix='blood'`, and a model for body mass given `prefix='mass'`. To allow for dependence between the individual-level random effects for blood pressure and mass, the two data sets and starting values produced by the `glmmBUGS` function can be combined, and the two model files concatenated and modified. Were it not for the prefix, some variable names would be identical in the two datasets and combining would not be possible.

Finally, multi-level models can be accommodated by passing multiple variable names in the `effects` argument. If the bacteria data were collected from students nested within classes nested within schools, the corresponding argument would be `effects = c("school","class","ID")`.

Currently unimplemented features that would enhance the package include: random coefficients (specified with an `lmer`-type syntax); survival distributions (with `"Surv"` objects as responses); and geostatistical models for the random effects. As WinBUGS incorporates methods for all of the above, these tasks are very feasible and offers of assistance from the community would be gratefully accepted.

Summary

There are a number of methods and R packages for performing Bayesian inference on generalized linear mixed models, including **geoRglm** (Christensen and Ribeiro, 2002), **MCMCglmm** (Hadfield, 2009) and **glmmGibbs** (Myles and Clayton, 2001). Particularly noteworthy is the **INLA** package which avoids the use of MCMC by using Laplace approximations, as described in Rue et al. (2009). Despite this, many researchers working with R (including the authors) continue to use WinBUGS for model fitting. This is at least partly due to the flexibility of the BUGS language, which gives the user nearly full control over specification of distributions and dependence structures. This package aims to retain this flexibility by providing files to be edited before model fitting is carried out, rather than providing a large number of options for, i.e. prior distributions. More model-specific packages such as **INLA** are entirely adequate (and perhaps preferable) for the two examples presented here. Those who wish to have full control over priors or build more complex models, for example multivariate or errors-in-covariate models, will likely appreciate the head start which the **glmmBUGS** package can provide for these tasks.

Acknowledgements

Patrick Brown holds a Discovery Grant from the Natural Sciences and Engineering Research Council of Canada.

Bibliography

O. Christensen and P. Ribeiro. geoRglm - a package for generalised linear spatial models. *R-NEWS*, 2(2):26–28, 2002. URL `http://cran.R-project.org/doc/Rnews`.

J. D. Hadfield. MCMC methods for multi-response generalised linear mixed models: The MCMCglmm R package. URL `http://cran.r-project.org/web/packages/MCMCglmm/vignettes/Tutorial.pdf`, 2009.

J. Myles and D. Clayton. Package GLMMGibbs. URL `http://cran.r-project.org/src/contrib/Archive/GLMMGibbs`, 2002.

H. Rue and L. Held. *Gaussian Markov Random Fields: Theory and Applications*, volume 104 of *Monographs on Statistics and Applied Probability*. Chapman & Hall, London, 2005.

H. Rue, S. Martino, and N. Chopin. Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the Royal Statistical Society — Series B*, 71 (2):319 – 392, 2009.

D. Spiegelhalter, A. Thomas, N. Best, and D. Lunn. WinBUGS user manual, 2004. URL http://mathstat.helsinki.fi/openbugs/data/Docu/WinBugs%20Manual.html.

S. Sturtz, U. Ligges, and A. Gelman. R2WinBUGS: A package for running WinBUGS from R. *Journal of Statistical Software*, 12(3):1–16, 2005. URL http://www.jstatsoft.org.

Patrick Brown
Dalla Lana School of Public Health,
University of Toronto, and
Cancer Care Ontario, Canada
patrick.brown@utoronto.ca

Lutong Zhou
Cancer Care Ontario, Canada
carly_zhou@hotmail.com

Mapping and Measuring Country Shapes

The cshapes Package

by Nils B. Weidmann and Kristian Skrede Gleditsch

Abstract The article introduces the **cshapes** R package, which includes our CShapes dataset of contemporary and historical country boundaries, as well as computational tools for computing geographical measures from these maps. We provide an overview of the need for considering spatial dependence in comparative research, how this requires appropriate historical maps, and detail how the **cshapes** associated R package **cshapes** can contribute to these ends. We illustrate the use of the package for drawing maps, computing spatial variables for countries, and generating weights matrices for spatial statistics.

Introduction

Our CShapes project (Weidmann et al., 2010) provides electronic maps of the international system for the period 1946–2008 in a standard Geographic Information System (GIS) format. Most existing GIS maps consider only contemporary country configurations and do not include information on changes in countries and borders over time. Beyond the value of more accurate historical maps in their own right for visualization, such maps can also help improve statistical analyses. Comparative research in the social sciences often considers country-level characteristics such as total population, economic performance, or the degree to which a country's political institutions can be characterized as democratic. Many theories relate such outcomes to other characteristics of individual countries, and typically ignore how possible interdependence between countries may also determine outcomes. However, over recent years there has been more attention to the spatial relationship between observations. For example, the level of democracy in a country can be strongly influenced by how democratic its neighbors are (Gleditsch, 2002). Similarly, conflict diffusion across national boundaries implies that the risk of civil war in a country will be partly determined by conflict in its immediate proximity (Gleditsch, 2007). Tobler's (1970, 236) *First Law of Geography* states that "everything is related to everything else, but near things are more related than distant things". To the extent that this applies to the international system, we need to–at a theoretical level–specify the transnational mechanisms that cause these interdependencies, and–at an analytical level–our empirical techniques must properly incorporate spatial interdependence among observations.

The CShapes dataset and the R package **cshapes** presented in this article help address the second task. Assessing spatial dependence in the international system requires measures of the connections and distances between our units of analysis, i.e., states. There exist various data collections that provide different indicators of spatial interdependence: a contiguity dataset (Gochman, 1991), the minimum-distance database by Gleditsch and Ward (2001), a dataset of shared boundary lengths (Furlong and Gleditsch, 2003), and distances between capitals (Gleditsch, 2008). However, although all of these collections entail indicators from a map of the international system at a given point in time, these databases provide the final indicators in fixed form rather than directly code the information from the underlying maps.

Our CShapes project takes a different approach and provides electronic maps of the international system for the period 1946–2008 from which such geographical indicators can be coded directly. The R package **cshapes** allows computing important distance measures directly from these maps. This approach has several advantages over the existing spatial indicators mentioned above. First, it ensures that the computed indicators are comparable, since they all refer to the same geopolitical entities. Second, maintaining and updating the dataset becomes much more convenient: If a change occurs in the international system – such as for example the independence of Montenegro in 2006 – we only need to update the electronic maps and re-run the computations in order to get up-to-date distance measures as of 2006. Third, the availability of electronic maps opens up new possibilities for visualization of country-level variables.

In the remainder of this article we provide a brief overview of the underlying spatial dataset of country boundaries, describing our coding approach and the data format. We then turn to the **cshapes** package for R, and describe three main applications the package can be used for: (i) visualizing country data as maps, (ii) computing new spatial indicators for countries, and (iii) creating weights matrices for spatial regression models.

The CShapes Dataset

Coding Criteria

Creating the CShapes dataset requires us to define coding rules with respect to three questions: (i) what constitutes an independent state, (ii) what is its spatial extent, and (iii) what constitutes a territorial change to a state. The following paragraphs summarize our approach. For a more detailed discussion, we refer the reader to Weidmann et al. (2010).

What is a state? There are various criteria for defining states. Research in Political Science often relies on the list of independent states provided by the *Correlates of War, 2008* (COW) project. This list relies on several criteria, including recognition by the UK and France, membership in the League of Nations and the United Nations, as well as other ad-hoc criteria. An alternative *Gleditsch and Ward* (GW) (1999) list of independent states addresses some limitations of the COW list, and has also gained a wide acceptance in the research community. As a result, we have made CShapes compatible with both the COW and GW state lists.

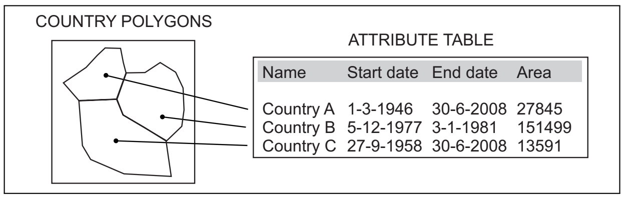
What is the spatial extent of a state? In many cases, the territory of a state comprises several (and possibly disjoint) regions with different status. For colonial states, an obvious distinction can be made between the "homeland" of a state and its dependent territories or colonies. Territory can also be disputed between states, such as for example the Golan Heights in the border region between Syria and Israel, or territory claimed may not be directly controlled or accepted by other states, as in the case of claims in Antarctica. As there are many such cases it is often impossible to allocate particular territories to states. To avoid this difficulty, we only code the core territories of state, without dependent territories or disputed regions.

What constitutes a territorial change? Given the spatial extent of a state, we need to define the territorial changes that should be reflected in the dataset. Our data encompass three types of changes are given in the dataset. First, new states can become independent, as in the case of the the secession of Eritrea from Ethiopia in 1993. Second, states can merge, as in the case of the dissolution of the (Eastern) German Democratic Republic in 1990 and its ascension to the (Western) German Federal Republic. Third, we may have territorial changes to a state that do not involve the emergence or disappearance of states.

CShapes includes all changes of the first two types. To keep the effort manageable, we only code changes of the third type that (i) affect the core territory of a state, and (ii) affect an area of at least 10,000 square kilometers. Our coding here is based on an existing dataset of territorial changes by Tir et al. (1998).

Implementation

Geographic data are typically stored in one of two formats: raster and vector data. Raster data represent continuous spatial quantities, such as territorial elevation, while vector data are used to capture discrete spatial entities ("features"), such as cities, rivers or lakes. There are three basic types of vector data: points, lines and polygons. The latter are the most appropriate for representing state boundaries. Apart from the spatial features themselves, vector data formats store non-spatial information related to them in an *attribute table*. For example, in a dataset of point features representing cities, we could store the name and population of each city in the attribute table.



Name	Start date	End date	Area
Country A	1-3-1946	30-6-2008	27845
Country B	5-12-1977	3-1-1981	151499
Country C	27-9-1958	30-6-2008	13591

Figure 1: CShapes vector data representation as polygons with lifetimes (from Weidmann et al., 2010).

In CShapes, we use polygons to represent state boundaries at a given time, and supplement the temporal information for these polygons in the attribute table. This is done by assigning a lifetime (i.e., a start day and an end day) to each polygon. By filtering the "active" polygons for a given date, it is possible to get a snapshot of the international system for this date. Figure 1 illustrates the CShapes data representation with each polygon's lifetime in the attribute table. The attribute table also contains the country identifier according to the COW and GW lists, as well as the location of the state capital in latitude/longitude coordinates. This information is required for the calculation of capital distances between states (see below). The CShapes vector dataset is provided in the shapefile format (ESRI, 1998), which can be used directly in standard GIS applications. However, in order to provide a more accessible interface to the data and to enable additional computations, we have developed the **cshapes** R package introduced in the next section.

The cshapes R Package and Applications

The **cshapes** package for R has two main purposes: first, to make the CShapes data available for use within R, and second, to perform additional operations on the data, in particular weights matrix computation for spatial regression modeling. In the following, we introduce the basic functionality of the package and later demonstrate its use through three illustrative applications.

Accessing the Data

In R, basic support for spatial data is provided by the **sp** package, which includes a set of classes both for vector and raster data. Other packages add functionality to work with these classes, as for example the **maptools** package for reading/writing shapefiles, drawing maps, or the **spdep** package for spatial regression. For an introduction to R's spatial features, we refer to the overview in Bivand (2006), and to Bivand et al. (2008) for a detailed introduction. **cshapes** draws on these packages in making the historical country boundaries available for use within R. The `cshp()` function is a convenience method to load the spatial dataset and returns a `SpatialPolygonsDataFrame`, which can be used to access and modify both the polygons and the attached attribute table. As an example to illustrate this, we first demonstrate how to extract country boundaries for a given point in time, and later use these data to visualize data, in this case the regime type of countries.

```
# load country border as of 2002
> cmap.2002 <- cshp(date=as.Date("2002-1-1"))
```

We can now join the extracted polygons for this date with supplementary information about countries. The Polity IV database (Marshall and Jaggers, 2008) provides a summary indicator about regime type. The indicator ranges from -10 to 10, with low values corresponding to autocratic regimes, and high values corresponding to democratic systems. For the example below, the Polity dataset is downloaded and filtered to retain only the 2002 values. Furthermore, the `polity.2002` data frame contains only two columns, the COW country identifier (ccode) and the regime type indicator (polity2). In order to match this information, we first need to match country identifiers from cshapes with those in the Polity dataset. As the Polity data is not available for all countries in 2002, we do not get a regime type value for each country in `cmap.2002`.

```
# download dataset, filter scores for 2002
> polity.file <- paste(tempdir(),
+ "p4v2008.sav",  sep="/")
> download.file("http://www.systemicpeace.org/
+ inscr/p4v2008.sav", polity.file)
> polity <- read.spss(polity.file, to.data.frame=T)
> polity.2002 <- polity[polity$year==2002,]
> polity.2002 <- subset(polity.2002,
+ !is.na(polity2), select=c(ccode, polity2))

# match country identifiers from both datasets
> o <- match(cmap.2002$COWCODE, polity.2002$ccode)

# order polity dataset accordingly
polity.2002 <- polity.2002[o,]

# set row names, required for spCbind function
row.names(polity.2002) <- cmap.2002$FEATUREID

# append using spCbind
cmap.2002.m <- spCbind(cmap.2002, polity.2002)
```

Drawing Maps

The above created dataset with appended regime type scores for 2002 can now be used for visualizing regime type across space. As described in Bivand et al. (2008, ch. 3), the **classInt** package is useful for creating maps with color shadings. The literature commonly distinguishes between three regime types: autocracies (Polity scores -10 through -7), anocracies (scores -6 through 6) and democracies (scores 7 through 10), a typology we employ for our example. Using the `classIntervals()` and `findColors()` functions, we divide regime type scores into three different classes and assign colors of different intensity, where more autocratic countries are displayed using darker colors. The following example shows the code necessary to generate the map in Figure 2.

```
# generate a color palette
> pal <- grey.colors(3, 0.25, 0.95)

# find the class intervals and colors
> breaks <- classIntervals(cmap.2002.m$polity2,
+ n=3, style="fixed", fixedBreaks=c(-10, -6, 7, 10))
> colors <- findColours(breaks, pal)

# create plot and add legend
> plot(cmap.2002.m, bty="n", col=colors)
> legend(x="bottomleft",
+ legend=c("Autocracy", "Anocracy", "Democracy"),
+ fill = attr(colors, "palette"),
+ bty = "n", title="Regime type")
```

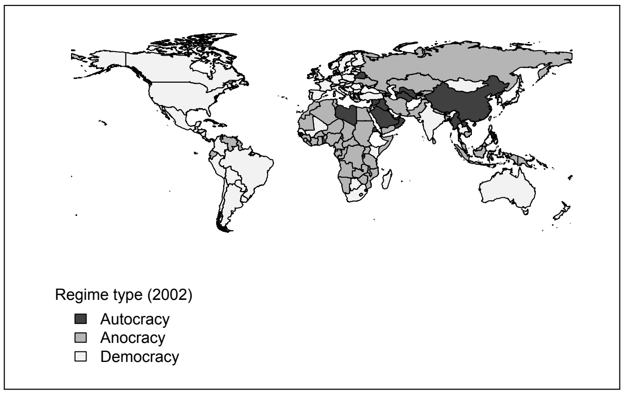


Figure 2: Mapping regime type scores according to the Polity dataset for 2002.

Spatial Variables for Countries

As an example of spatial characteristics that can be calculated using **cshapes** we consider the distance between a country's capital and its geographical center. The extent to which a country contains extensive peripheries or hinterlands is often held to be an important influence on state consolidation and the risk of conflict (Herbst, 2000; Buhaug et al., 2008). A simple measure of unfavorable geography can be derived from comparing the position of a state capital relative to the geographic center of a country. Figure 3 shows the borders of two African countries, the Democratic Republic of the Congo and Nigeria, as well as their capitals and their polygon centroids. In the Democratic Republic of the Congo (left), the capital Kinshasa is located in the far West of the country, which creates large hinterlands in the East. Nigeria (right) has a more balanced configuration, with the capital Abuja located close to the centroid.

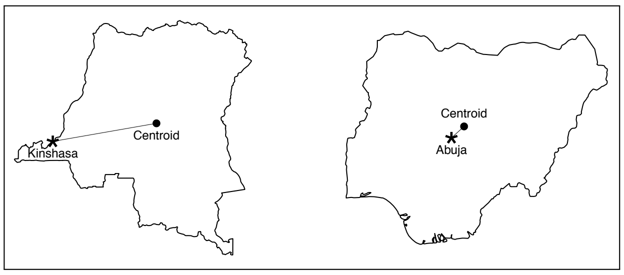


Figure 3: Illustration of the measure of deviation between the geographic centroid of a polygon and the capital. Left: Democratic Republic of Congo, right: Nigeria (from Weidmann et al., 2010).

For a systematic statistical test, we compute the capital–centroid deviation from the capital to the polygon centroid for each country/year. We do not show the complete regression analysis here, but focus on the computation of this indicator. We first obtain the centroids for all polygons using the `coordinates()` function in the **sp** package. We then apply a newly created function called `distance()` to each polygon. This function is essentially a wrapper around the `spDistsN1` function from the **sp** package and computes the distance between each polygons's centroid and its capital. The capital longitude and latitude is obtained directly from the respective polygon's attributes (the CAPLONG and CAPLAT fields). Note that since the CShapes dataset is unprojected, we need to set the `longlat=TRUE` parameter in the `spDistsN1` function in order to obtain great circle distances.

```
> cmap <- cshp()
> centroids <- coordinates(cmap)
> distance <- function(from.x, from.y, to.x, to.y)
+ from <- matrix(c(from.x, from.y), ncol=2, nrow=1)
+ spDistsN1(from, c(to.x, to.y), longlat = TRUE)
+
> cmap$capcentdist <- mapply(distance, centroids[,1],
+ centroids[,2], cmap$CAPLONG, cmap$CAPLAT)
```

Computing Country Distances

Spatial statistical applications typically require that we first specify spatial structure. One way to represent this is by means of a matrix indicating adjacency or distance between any two observations i, j. A matrix of this kind can then be transformed into a weights matrix that assigns a higher weight to spatially close observations. As discussed above, there are many advantages to computing these distances directly from a spatial dataset such as CShapes, which is why we provide a dynamic way of distance computation in our package, rather than offering pre-computed distances. This also makes it possible to compute these matrices for arbitrary points in time.

With countries as the unit of analysis, distance matrices are typically based on one of three types of distance (Ward and Gleditsch, 2008). First, we can consider the distance between the capital cities of two countries. Second, rather than capitals, we can use the geographical center point of a country for distance computation. This center point or centroid is the country's geometric center of gravity. Third, we can measure the minimum distance between two countries, i.e. the distance between two points, one from each country, that are closest to each other. Gleditsch and Ward (2001) discuss the relative advantages of different measures in greater detail. Computing distances between predetermined points as in the first two alternatives is straightforward; we only need to take into account that the CShapes dataset is unprojected, which is why we compute great circle distances. Finding the minimum distance, however, is more complicated as we need to consider the shape and position for two countries to find the closest points.

The `distmatrix()` function. **cshapes** computes distance matrices of all three types with its `distmatrix()` function. Capital and centroid distances are easy to compute, as there is only one point per polygon used for the computation. Obtaining the minimum distance is more difficult. Given two countries, we need to compute the distance of each point on one of the polygons to all lines in the other polygon. This procedure must be repeated for each pair of countries. Because the computational costs of this procedure, the **cshapes** package implements three ways for speeding up this computation. First, we implement the distance computation in Java, using the **rJava** package for interfacing and the Java Topology Suite[1] (JTS) for handling spatial data in Java. Second, we resort to computing distances between pairs of points in the two polygons, rather than between points and lines. Third, we reduce the number of points in a polygon, which results in fewer comparisons to be made. The polygon sim-

[1]http://www.vividsolutions.com/jts/jtshome.htm

plification is readily implemented in JTS according to the algorithm proposed by Douglas and Peucker (1973). The accuracy of the simplification can be controlled by a threshold parameter that indicates the maximum allowed deviation of the simplified shape from the original one. Figure 4 shows an example of the shapes generated by the polygon simplification, using our default threshold (0.1).

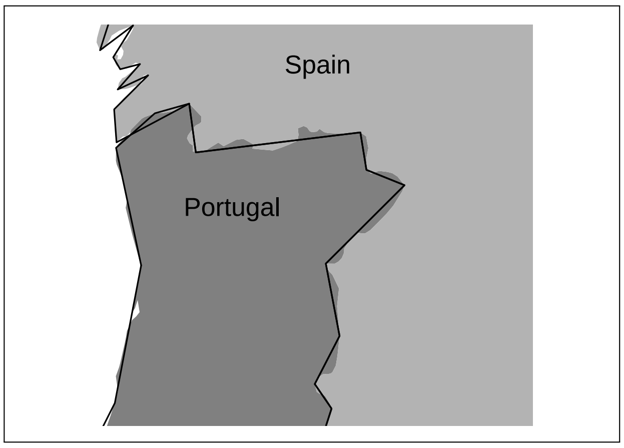


Figure 4: Result of the polygon simplification procedure using the Douglas-Peucker algorithm. The shaded areas show the borders between Portugal and Spain according to the original, detailed dataset. The solid line shows these boundaries after the simplification process, where twisted segments have been replaced by straight ones, while preserving the basic shape of a country.

`distmatrix()` is the **cshapes** function to generate distance matrices. It returns a symmetric matrix $D_{i,j}$, where the entries $d_{i,j}$ correspond to the distance between countries i and j in kilometers, and all $d_{i,j} = 0$. The rows and columns are labeled by the respective country identifier, according to the COW or GW coding system. Since the layout of the international system changes over time, the first parameter of the function specifies the date at which the matrix should be computed. The country coding system to be used is set by the `useGW` parameter, which defaults to the Gleditsch and Ward system. The distance type is specified by the `type` parameter that has three possible values: `capdist`, `centdist` and `mindist`. Furthermore, the `tolerance` parameter sets the threshold for the Douglas-Peucker polygon simplification described above. This parameter only affects minimum distance computation. The default value of this parameter is 0.1, and a value of 0 disables polygon simplification. Below, we compare the execution time obtained with default polygon simplification to no simplification.[2]

```
> system.time(dmat.simple <- distmatrix(
+ as.Date("2002-1-1"), type="mindist"))
   user  system elapsed
223.286   0.842 224.523

> system.time(dmat.full <- distmatrix(
+ as.Date("2002-1-1"), type="mindist",
+ tolerance=0))
    user   system  elapsed
12452.40    48.67 12686.44

> cor(as.vector(dmat.simple),
+ as.vector(dmat.full))
[1] 0.9999999
```

Because of the quadratic scaling of the minimum distance computation, the reduction in computational costs achieved by the simplification procedure is extremely high. Whereas default polygon simplification allows us to compute the complete matrix in less than four minutes, the same procedure on the full dataset takes about four hours. Still, the inaccuracies introduces by the simplification are very small; the computed distance matrices correlate at 0.999.

Computing spatial weights. Once a distance matrix has been computed, we can transform it to different weights matrices, depending on the respective application. Weights matrices based on spatial adjacency assign a weight of 1 to all entries with $d_{i,j} \leq t$ for a distance threshold t. Empirical research suggests that it is helpful for many applications to have a relatively inclusive threshold t (for example, 900 km) to determine relevant neighboring states, in order to prevent excluding states that are not directly contiguous yet in practice often have high levels of interaction (Gleditsch, 2002). Alternatively, we can transform a distance matrix into a continuous weights matrix by inverting the $d_{i,j}$. The following code illustrates how to generate weights from **cshapes** distance matrices.

```
> dmat <- distmatrix(as.Date("2002-1-1"),
+ type="mindist")

# adjacency matrix, 900 km threshold
> adjmat <- ifelse(dmat > 900, 0, 1)
> diag(adjmat) <- 0

# inverse distance
> invdmat <- 1/dmat
> diag(invdmat) <- 0

# save matrix for later use
> write.table(adjmat, "adjmat2002.txt",
+ row.names=T, col.names=T)

# load matrix
> adjmat <- read.table("adjmat2002.txt",
+ header=T, check.names=F)
```

Integration with spdep. **spdep** is one of R's most advanced packages for spatial statistical analysis.

[2]Hardware: Macbook 2.4 GHz Intel Core 2 Duo, 4 GB RAM.

It relies on two basic representations for spatial structure: neighbor lists for discrete neighborhood relationships, and weights lists for storing spatial weights between observations. Using functions from the **spdep** package, it is possible to convert the matrices created by **cshapes**. The following code demonstrates how to do this.

```
# neighbor list from adjacency matrix:
# create a weights list first...
> adjmat.lw <- mat2listw(adjmat,
+ row.names=row.names(dmat))

# ... and retrieve neighbor list
> adjmat.nb <- adjmat.lw$neighbours

# weights list from weights matrix
> invdmat.lw <- mat2listw(invdmat,
+ row.names=row.names(invdmat))
```

The neighbor lists obtained from **cshapes** can then be used to fit spatial regression models using **spdep**.

Conclusions

Accurate historical maps are essential for comparative cross-national research, and for incorporating spatial dependence into applied empirical work. Our CShapes dataset provides such information in a standard GIS format for the period 1946–2008. In this paper, we have introduced our **cshapes** package that provides a series of tools for useful operations on these data within R. Our package supplements the existing valuable packages for spatial analysis in R, and we hope that the package may facilitate greater attention to spatial dependence in cross-national research.

Bibliography

R. S. Bivand. Implementing spatial data analysis software tools in R. *Geographical Analysis*, 38(2006):23–40, 2006.

R. S. Bivand, E. J. Pebesma, and V. Gómez-Rubio. *Applied Spatial Data Analysis with R*. Springer, New York, 2008.

H. Buhaug, L.-E. Cederman, and J. K. Rød. Disaggregating ethnic conflict: A dyadic model of exclusion theory. *International Organization*, 62(3):531–551, 2008.

Correlates of War Project. State system membership list, v2008.1, 2008. Available online at http://correlatesofwar.org.

D. Douglas and T. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10(2):112–122, 1973.

ESRI. ESRI shapefile technical description, 1998. Available at `http://esri.com/library/whitepapers/pdfs/shapefile.pdf`.

K. Furlong and N. P. Gleditsch. The boundary dataset. *Conflict Management and Peace Science*, 29 (1):93–117, 2003.

K. S. Gleditsch. *All Politics is Local: The Diffusion of Conflict, Integration, and Democratization*. University of Michigan Press, Ann Arbor, MI, 2002.

K. S. Gleditsch. Transnational dimensions of civil war. *Journal of Peace Research*, 44(3):293–309, 2007.

K. S. Gleditsch. Distances between capital cities, 2008. Available at `http://privatewww.essex.ac.uk/~ksg/data-5.html`.

K. S. Gleditsch and M. D. Ward. Measuring space: A minimum distance database and applications to international studies. *Journal of Peace Research*, 38 (6):749–768, December 2001.

K. S. Gleditsch and M. D. Ward. Interstate system membership: A revised list of the independent states since 1816. *International Interactions*, 25:393–413, 1999.

C. S. Gochman. Interstate metrics: Conceptualizing, operationalizing, and measuring the geographic proximity of states since the congress of Vienna. *International Interactions*, 17(1):93–112, 1991.

J. Herbst. *States and Power in Africa: Comparative Lessons in Authority and Control*. Princeton University Press, Princeton, NJ, 2000.

M. G. Marshall and K. Jaggers. Polity IV project: Political regime characteristics and transitions, 1800-2007, 2008. Available at http://www.systemicpeace.org/polity/polity4.htm.

J. Tir, P. Schafer, P. F. Diehl, and G. Goertz. Territorial changes, 1816-1996. *Conflict Management and Peace Science*, 16:89–97, 1998.

W. R. Tobler. A computer movie simulating urban growth in the Detroit region. *Economic Geography*, 46:234–240, 1970.

M. D. Ward and K. S. Gleditsch. *Spatial Regression Models*. Sage, Thousand Oaks, CA, 2008.

N. B. Weidmann, D. Kuse, and K. S. Gleditsch. The geography of the international system: The CShapes dataset. *International Interactions*, 36(1), 2010.

Nils B. Weidmann
Woodrow Wilson School
Princeton University, USA
`nils.weidmann@gmail.com`

Kristian Skrede Gleditsch
Department of Government
University of Essex, UK
`ksg@essex.ac.uk`

tmvtnorm: A Package for the Truncated Multivariate Normal Distribution

by Stefan Wilhelm and B. G. Manjunath

Abstract In this article we present **tmvtnorm**, an R package implementation for the truncated multivariate normal distribution. We consider random number generation with rejection and Gibbs sampling, computation of marginal densities as well as computation of the mean and covariance of the truncated variables. This contribution brings together latest research in this field and provides useful methods for both scholars and practitioners when working with truncated normal variables.

Introduction

The package **mvtnorm** is the first choice in R for dealing with the Multivariate Normal Distribution (Genz et al., 2009). However, frequently one or more variates in a multivariate normal setting $\mathbf{x} = (x_1, \dots, x_m)^T$ are subject to one-sided or two-sided truncation ($a_i \leq x_i \leq b_i$). The package **tmvtnorm** (Wilhelm and Manjunath (2010)) is an extension of the **mvtnorm** package and provides methods necessary to work with the truncated multivariate normal case defined by $TN(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{a}, \mathbf{b})$.
The probability density function for $TN(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{a}, \mathbf{b})$ can be expressed as:

$$f(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{a}, \mathbf{b}) = \frac{\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}}{\int_{\mathbf{a}}^{\mathbf{b}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\} d\mathbf{x}}$$

for $\mathbf{a} \leq \mathbf{x} \leq \mathbf{b}$ and 0 otherwise. We will use the following bivariate example with $\boldsymbol{\mu} = (0.5, 0.5)^T$ and covariance matrix $\boldsymbol{\Sigma}$

$$\boldsymbol{\Sigma} = \begin{pmatrix} 1 & 0.8 \\ 0.8 & 2 \end{pmatrix}$$

as well as lower and upper truncation points $\mathbf{a} = (-1, -\infty)^T, \mathbf{b} = (0.5, 4)^T$, i.e. x_1 is doubly, while x_2 is singly truncated. The setup in R code is:

```
> library(tmvtnorm)
> mu <- c(0.5, 0.5)
> sigma <- matrix(c(1, 0.8, 0.8, 2), 2, 2)
> a <- c(-1, -Inf)
> b <- c(0.5, 4)
```

The following tasks for dealing with the truncated multinormal distribution are described in this article:

- generation of random numbers
- computation of marginal densities
- computation of moments (mean vector, covariance matrix)
- estimation of parameters

Generation of random numbers

Rejection sampling

The first idea to generate variates from a truncated multivariate normal distribution is to draw from the untruncated distribution using `rmvnorm()` in the **mvtnorm** package and to accept only those samples inside the support region (i.e., rejection sampling). The different algorithms used to generate samples from the multivariate normal distribution have been presented for instance in Mi et al. (2009) and in Genz and Bretz (2009).

The following R code can be used to generate $N = 10000$ samples using rejection sampling:

```
> X <- rtmvnorm(n=10000, mean=mu,
>      sigma=sigma, lower=a, upper=b,
>      algorithm="rejection")
```

The parameter `algorithm="rejection"` is the default value and may be omitted.

This approach works well, if the acceptance rate α is reasonably high. The method `rtmvnorm()` first calculates the acceptance rate α and then iteratively generates N/α draws in order to get the desired number of remaining samples N. This typically needs just a few iterations and is fast. Figure 1 shows random samples obtained by rejection sampling. Accepted samples are marked as black, discarded samples are shown in gray. In the example, the acceptance rate is $\alpha = 0.432$.

```
> alpha <- pmvnorm(lower=a, upper=b, mean=mu,
>           sigma=sigma)
```

However, if only a small fraction of samples is accepted, as is the case in higher dimensions, the number of draws per sample can be quite substantial. Then rejection sampling becomes very inefficient and finally breaks down.

Gibbs sampling

The second approach to generating random samples is to use the Gibbs sampler, a Markov Chain Monte Carlo (MCMC) technique. The Gibbs sampler samples from conditional univariate distributions $f(x_i|\mathbf{x}_{-i}) = f(x_i|x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_m)$ which are themselves truncated univariate normal distributions (see for instance Horrace (2005) and Kotecha and Djuric (1999)). We implemented the algorithm presented in the paper of Kotecha and Djuric (1999) in the **tmvtnorm** package with minor improvements. Using `algorithm="gibbs"`, users can sample with the Gibbs sampler.

```
> X <- rtmvnorm(n=10000, mean=mu,
>      sigma=sigma, lower=a, upper=b,
>      algorithm="gibbs")
```

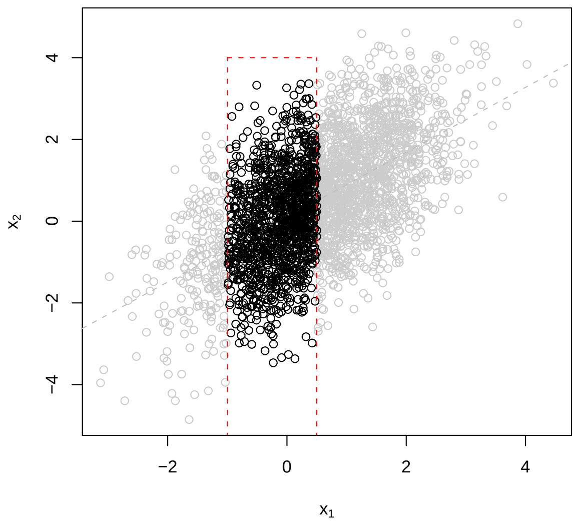


Figure 1: Random Samples from a truncated bivariate normal distribution generated by `rtmvnorm()`

Gibbs sampling produces a Markov chain which finally converges to a stationary target distribution. Generally, the convergence of MCMC has to be checked using diagnostic tools (see for instance the **coda** package from Plummer et al. (2009)). The starting value of the chain can be set as an optional parameter `start.value`. Like all MCMC methods, the first iterates of the chain will not have the exact target distribution and are strongly dependent on the start value. To reduce this kind of problem, the first iterations are considered as a burn-in period which a user might want to discard. Using `burn.in.samples=100`, one can drop the first 100 samples of a chain as burn-in samples.

The major advantage of Gibbs sampling is that it accepts all proposals and is therefore not affected by a poor acceptance rate α. As a major drawback, random samples produced by Gibbs sampling are not independent, but correlated. The degree of correlation depends on the covariance matrix $\boldsymbol{\Sigma}$ as well on the dimensionality and should be checked for using autocorrelation or cross-correlation plots (e.g. `acf()` or `ccf()` in the **stats** package).

```
> acf(X)
```

Taking only a nonsuccessive subsequence of the Markov chain, say every k-th sample, can greatly reduce the autocorrelation among random points as shown in the next code example. This technique called "thinning" is available via the `thinning=k` argument:

```
> X2 <- rtmvnorm(n=10000, mean=mu,
>       sigma=sigma, lower=a, upper=b,
>       algorithm="gibbs", burn.in.samples=100,
>       thinning = 5)
> acf(X2)
```

While the samples `X` generated by Gibbs sampling exhibit cross-correlation for both variates up to lag 1, this correlation vanished in `X2`.

Table 1 shows a comparison of rejection sampling and Gibbs sampling in terms of speed. Both algorithms scale with the number of samples N and the number of dimensions m, but Gibbs sampling is independent from the acceptance rate α and even works for very small α. On the other hand, the Gibbs sampler scales linearly with `thinning` and requires extensive loops which are slow in an interpreted language like R. From package version 0.8, we reimplemented the Gibbs sampler in Fortran. This compiled code is now competitive with rejection sampling. The deprecated R code is still accessible in the package via `algorithm="gibbsR"`.

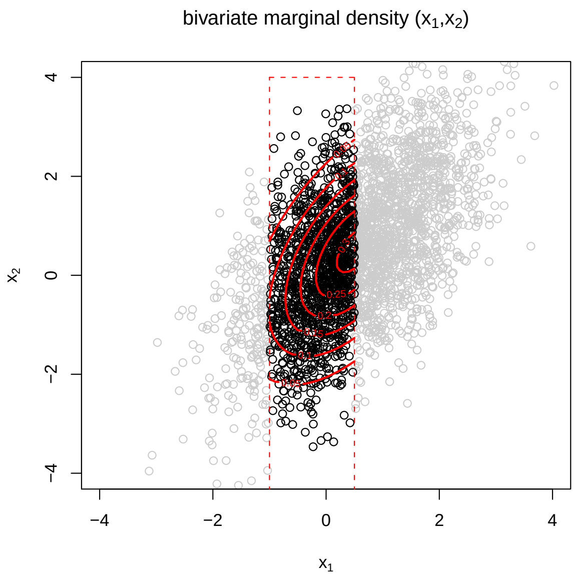


Figure 2: Contour plot for the bivariate truncated density function obtained by `dtmvnorm()`

Marginal densities

The joint density function $f(\mathbf{x}, \mu, \boldsymbol{\Sigma}, \mathbf{a}, \mathbf{b})$ for the truncated variables can be computed using `dtmvnorm()`. Figure 2 shows a contour plot of the bivariate density in our example.

Algorithm	Samples	$m=2$		$m=10$			
		$R_s=\prod_{i=1}^{2}(-1,1]$ $\alpha=0.56$		$R_s=\prod_{i=1}^{10}(-1,1]$ $\alpha=0.267$		$R_s=\prod_{i=1}^{10}(-4,-3]$ $\alpha=5.6\cdot 10^{-6}$	
Rejection Sampling	$N=10000$	5600	0.19 sec	2670	0.66 sec	0	∞
	$N=100000$	56000	1.89 sec	26700	5.56 sec	1	∞
Gibbs Sampling (R code, no thinning)	$N=10000$	10000	0.75 sec	10000	3.47 sec	10000	3.44 sec
	$N=100000$	100000	7.18 sec	100000	34.58 sec	100000	34.58 sec
Gibbs Sampling (Fortran code, no thinning)	$N=10000$	10000	0.03 sec	10000	0.13 sec	10000	0.12 sec
	$N=100000$	100000	0.22 sec	100000	1.25 sec	100000	1.21 sec
Gibbs Sampling (Fortran code and `thinning=10`)	$N=10000$	10000	0.22 sec	10000	1.22 sec	10000	1.21 sec
	$N=100000$	100000	2.19 sec	100000	12.24 sec	100000	12.19 sec

Table 1: Comparison of rejection sampling and Gibbs sampling for 2-dimensional and 10-dimensional TN with support $R_s=\prod_{i=1}^{2}(-1,1]$, $R_s=\prod_{i=1}^{10}(-1,1]$ and $R_s=\prod_{i=1}^{10}(-4,-3]$ respectively: number of generated samples, acceptance rate α and computation time measured on an Intel® Core™Duo CPU@2.67 GHz

However, more often marginal densities of one or two variables will be of interest to users. For multinormal variates the marginal distributions are again normal. This property does not hold for the truncated case. The marginals of truncated multinormal variates are not truncated normal in general. An explicit formula for calculating the one-dimensional marginal density $f(x_i)$ is given in the paper of Cartinhour (1990). We have implemented this algorithm in the method `dtmvnorm.marginal()` which, for convenience, is wrapped in `dtmvnorm()` via a `margin=i` argument. Figure 3 shows the Kernel density estimate and the one-dimensional marginal for both x_1 and x_2 calculated using the `dtmvnorm(...,  margin=i)` $i=1,2$ call:

```
> x  <- seq(-1, 0.5, by=0.1)
> fx <- dtmvnorm(x, mu, sigma,
>       lower=a, upper=b, margin=1)
```

The bivariate marginal density $f(x_q,x_r)$ for x_q and x_r ($m>2, q\neq r$) is implemented based on the works of Leppard and Tallis (1989) and Manjunath and Wilhelm (2009) in method `dtmvnorm.marginal2()` which, again for convenience, is just as good as `dtmvnorm(..., margin=c(q,r))`:

```
> fx <- dtmvnorm(x=c(0.5, 1),
>       mu, sigma,
>       lower=a, upper=b, margin=c(1,2))
```

The help page for this function in the package contains an example of how to produce a density contour plot for a trivariate setting similar to the one shown in figure 2.

Computation of moments

The computation of the first and second moments (mean vector $\mu^*=E[\mathbf{x}]$ and covariance matrix Σ^* respectively) is not trivial for the truncated case, since they are obviously not the same as μ and Σ from the parametrization of $TN(\mu,\Sigma,\mathbf{a},\mathbf{b})$. The moment-generating function has been given first by Tallis (1961), but his formulas treated only the special case of a singly-truncated distribution with a correlation matrix $\boldsymbol{R}$. Later works, especially Lee (1979) generalized the computation of moments for a covariance matrix Σ and gave recurrence relations between moments, but did not give formulas for the moments of the double-truncated case. We presented the computation of moments for the general double-truncated case in a working paper Manjunath and Wilhelm (2009) and implemented the algorithm in the method `mtmvnorm()`, which can be used as

```
> moments <- mtmvnorm(mean=mu, sigma=sigma,
>             lower=a, upper=b)
```

The moment calculation for our example results in $\mu^*=(-0.122,0)^T$ and covariance matrix

$$\Sigma^*=\begin{pmatrix} 0.165 & 0.131 \\ 0.131 & 1.458 \end{pmatrix}$$

To compare these results with the sample moments, use

```
> colMeans(X)
> cov(X)
```

It can be seen in this example that truncation can significantly reduce the variance and change the covariance between variables.

Estimation

Unlike our example, in many settings μ and Σ are unknown and must be estimated from the data. Estimation of (μ,Σ) when truncation points $\mathbf{a}$ and $\mathbf{b}$ are known can typically be done by either Maximum-Likelihood, Instrumental Variables (Amemiya (1973), Amemiya (1974), Lee (1979), Lee (1983)) or in a Bayesian context (Griffiths (2002)). We are planning to include these estimation approaches in the package in future releases.

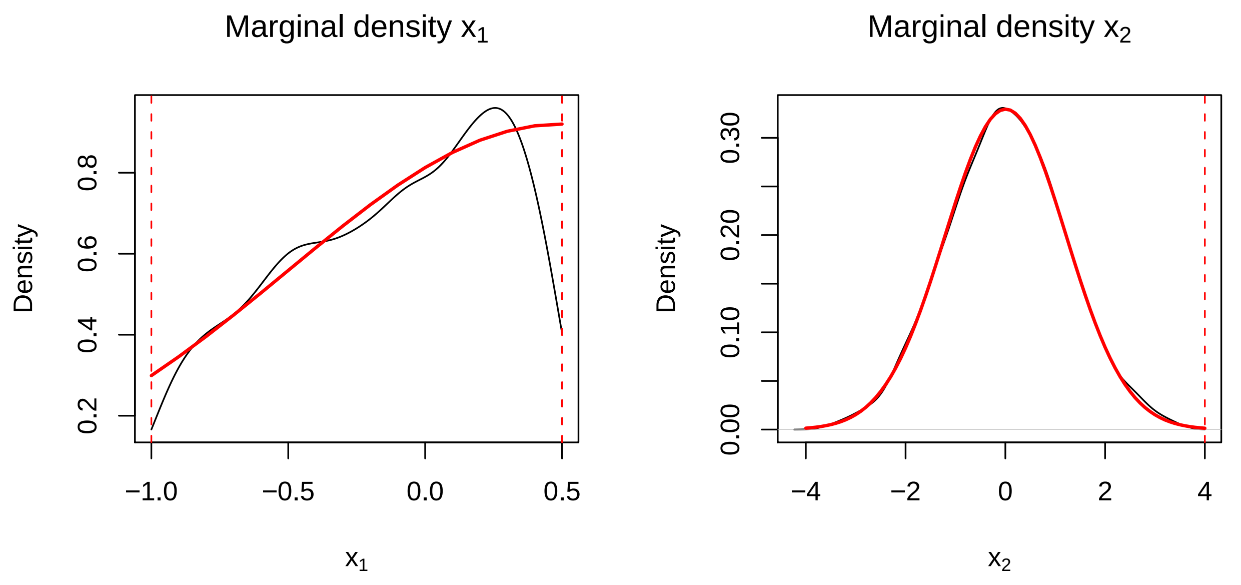


Figure 3: Marginal densities for x_1 and x_2 obtained from Kernel density estimation from random samples generated by `rtmvnorm()` and from direct calculation using `dtmvnorm(...,margin=1)` and `dtmvnorm(...,margin=2)` respectively

Summary

The package **tmvtnorm** provides methods for simulating from truncated multinormal distributions (rejection and Gibbs sampling), calculations from marginal densities and also calculation of mean and covariance of the truncated variables. We hope that many useRs will find this package useful when dealing with truncated normal variables.

Bibliography

T. Amemiya. Regression analysis when the dependent variable is truncated normal. *Econometrica*, 41: 997–1016, 1973.

T. Amemiya. Multivariate regression and simultaneous equations models when the dependent variables are truncated normal. *Econometrica*, 42:999–1012, 1974.

J. Cartinhour. One-dimensional marginal density functions of a truncated multivariate normal density function. *Communications in Statistics - Theory and Methods*, 19:197–203, 1990.

A. Genz and F. Bretz. *Computation of Multivariate Normal and t Probabilities*, volume 195 of *Lecture Notes in Statistics*. Springer-Verlag, Heidelberg, 2009.

A. Genz, F. Bretz, T. Miwa, X. Mi, F. Leisch, F. Scheipl, and T. Hothorn. *mvtnorm: Multivariate Normal and t Distributions*, 2009. URL `http://CRAN.R-project.org/package=mvtnorm`. R package version 0.9-8.

W. Griffiths. A Gibbs' sampler for the parameters of a truncated multivariate normal distribution. 2002. URL `http://ideas.repec.org/p/mlb/wpaper/856.html`.

W. C. Horrace. Some results on the multivariate truncated normal distribution. *Journal of Multivariate Analysis*, 94:209–221, 2005.

J. H. Kotecha and P. M. Djuric. Gibbs sampling approach for generation of truncated multivariate gaussian random variables. *IEEE Computer Society*, pages 1757–1760, 1999. `http://dx.doi.org/10.1109/ICASSP.1999.756335`.

L.-F. Lee. On the first and second moments of the truncated multi-normal distribution and a simple estimator. *Economics Letters*, 3:165–169, 1979.

L.-F. Lee. The determination of moments of the doubly truncated multivariate tobit model. *Economics Letters*, 11:245–250, 1983.

P. Leppard and G. M. Tallis. Evaluation of the mean and covariance of the truncated multinormal. *Applied Statistics*, 38:543–553, 1989.

B. G. Manjunath and S. Wilhelm. Moments calculation for the double truncated multivariate normal density. Working Paper, 2009. URL `http://ssrn.com/abstract=1472153`.

X. Mi, T. Miwa, and T. Hothorn. mvtnorm: New numerical algorithm for multivariate normal probabilities. *R Journal*, 1:37–39, 2009.

M. Plummer, N. Best, K. Cowles, and K. Vines. *coda: Output analysis and diagnostics for MCMC*, 2009. R package version 0.13-4.

G. M. Tallis. The moment generating function of the truncated multinormal distribution. *Journal of the Royal Statistical Society, Series B*, 23(1):223–229, 1961.

S. Wilhelm and B. G. Manjunath. *tmvtnorm: Truncated Multivariate Normal Distribution*, 2010. URL http://CRAN.R-project.org/package=tmvtnorm. R package version 0.9-2.

Stefan Wilhelm
Department of Finance
University of Basel, Switzerland
wilhelm@financial.com

B. G. Manjunath
Department of Mathematics
University of Siegen, Germany
bgmanjunath@gmail.com

neuralnet: Training of Neural Networks

by Frauke Günther and Stefan Fritsch

Abstract Artificial neural networks are applied in many situations. **neuralnet** is built to train multi-layer perceptrons in the context of regression analyses, i.e. to approximate functional relationships between covariates and response variables. Thus, neural networks are used as extensions of generalized linear models.
neuralnet is a very flexible package. The backpropagation algorithm and three versions of resilient backpropagation are implemented and it provides a custom-choice of activation and error function. An arbitrary number of covariates and response variables as well as of hidden layers can theoretically be included.
The paper gives a brief introduction to multi-layer perceptrons and resilient backpropagation and demonstrates the application of **neuralnet** using the data set `infert`, which is contained in the R distribution.

Introduction

In many situations, the functional relationship between covariates (also known as input variables) and response variables (also known as output variables) is of great interest. For instance when modeling complex diseases, potential risk factors and their effects on the disease are investigated to identify risk factors that can be used to develop prevention or intervention strategies. Artificial neural networks can be applied to approximate any complex functional relationship. Unlike generalized linear models (GLM, McCullagh and Nelder, 1983), it is not necessary to prespecify the type of relationship between covariates and response variables as for instance as linear combination. This makes artificial neural networks a valuable statistical tool. They are in particular direct extensions of GLMs and can be applied in a similar manner. Observed data are used to train the neural network and the neural network learns an approximation of the relationship by iteratively adapting its parameters.

The package **neuralnet** (Fritsch and Günther, 2008) contains a very flexible function to train feed-forward neural networks, i.e. to approximate a functional relationship in the above situation. It can theoretically handle an arbitrary number of covariates and response variables as well as of hidden layers and hidden neurons even though the computational costs can increase exponentially with higher order of complexity. This can cause an early stop of the iteration process since the maximum of iteration steps, which can be defined by the user, is reached before the algorithm converges. In addition, the package provides functions to visualize the results or in general to facilitate the usage of neural networks. For instance, the function `compute` can be applied to calculate predictions for new covariate combinations.

There are two other packages that deal with artificial neural networks at the moment: **nnet** (Venables and Ripley, 2002) and **AMORE** (Limas et al., 2007). **nnet** provides the opportunity to train feed-forward neural networks with traditional backpropagation and in **AMORE**, the TAO robust neural network algorithm is implemented. **neuralnet** was built to train neural networks in the context of regression analyses. Thus, resilient backpropagation is used since this algorithm is still one of the fastest algorithms for this purpose (e.g. Schiffmann et al., 1994; Rocha et al., 2003; Kumar and Zhang, 2006; Almeida et al., 2010). Three different versions are implemented and the traditional backpropagation is included for comparison purposes. Due to a custom-choice of activation and error function, the package is very flexible. The user is able to use several hidden layers, which can reduce the computational costs by including an extra hidden layer and hence reducing the neurons per layer. We successfully used this package to model complex diseases, i.e. different structures of biological gene-gene interactions (Günther et al., 2009). Summarizing, **neuralnet** closes a gap concerning the provided algorithms for training neural networks in R.

To facilitate the usage of this package for new users of artificial neural networks, a brief introduction to neural networks and the learning algorithms implemented in **neuralnet** is given before describing its application.

Multi-layer perceptrons

The package **neuralnet** focuses on multi-layer perceptrons (MLP, Bishop, 1995), which are well applicable when modeling functional relationships. The underlying structure of an MLP is a directed graph, i.e. it consists of vertices and directed edges, in this context called neurons and synapses. The neurons are organized in layers, which are usually fully connected by synapses. In **neuralnet**, a synapse can only connect to subsequent layers. The input layer consists of all covariates in separate neurons and the output layer consists of the response variables. The layers in between are referred to as hidden layers, as they are not directly observable. Input layer and hidden layers include a constant neuron relating to intercept synapses, i.e. synapses that are not directly influenced by any covariate. Figure 1 gives an example of a neural network with one hidden layer that consists of three hidden neurons. This neural network models the relationship between the two covariates A and B and the response variable Y. **neuralnet** theoretically allows inclusion of arbitrary numbers of covariates and response variables. However, there can occur convergence difficulties using a huge number of both covariates and response variables.

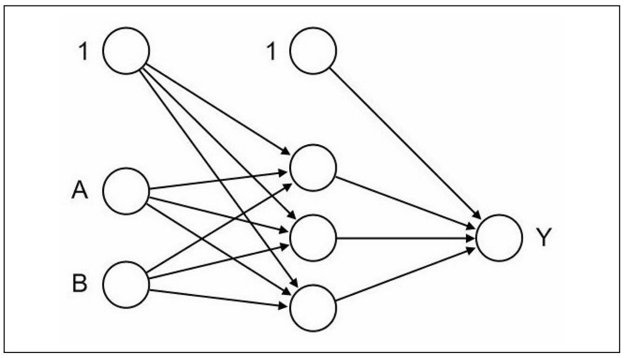


Figure 1: Example of a neural network with two input neurons (A and B), one output neuron (Y) and one hidden layer consisting of three hidden neurons.

To each of the synapses, a weight is attached indicating the effect of the corresponding neuron, and all data pass the neural network as signals. The signals are processed first by the so-called integration function combining all incoming signals and second by the so-called activation function transforming the output of the neuron.

The simplest multi-layer perceptron (also known as perceptron) consists of an input layer with n covariates and an output layer with one output neuron. It calculates the function

$$o(\mathbf{x}) = f\left(w_0 + \sum_{i=1}^{n} w_i x_i\right) = f\left(w_0 + \mathbf{w}^T\mathbf{x}\right),$$

where w_0 denotes the intercept, $\mathbf{w} = (w_1, \ldots, w_n)$ the vector consisting of all synaptic weights without the intercept, and $\mathbf{x} = (x_1, \ldots, x_n)$ the vector of all covariates. The function is mathematically equivalent to that of GLM with link function f^{-1}. Therefore, all calculated weights are in this case equivalent to the regression parameters of the GLM.

To increase the modeling flexibility, hidden layers can be included. However, Hornik et al. (1989) showed that one hidden layer is sufficient to model any piecewise continuous function. Such an MLP with a hidden layer consisting of J hidden neurons calculates the following function:

$$\begin{aligned} o(\mathbf{x}) &= f\left(w_0 + \sum_{j=1}^{J} w_j \cdot f\left(w_{0j} + \sum_{i=1}^{n} w_{ij} x_i\right)\right) \\ &= f\left(w_0 + \sum_{j=1}^{J} w_j \cdot f\left(w_{0j} + \mathbf{w_j}^T\mathbf{x}\right)\right), \end{aligned}$$

where w_0 denotes the intercept of the output neuron and w_{0j} the intercept of the jth hidden neuron. Additionally, w_j denotes the synaptic weight corresponding to the synapse starting at the jth hidden neuron and leading to the output neuron, $\mathbf{w_j} = (w_{1j}, \ldots, w_{nj})$ the vector of all synaptic weights corresponding to the synapses leading to the jth hidden neuron, and $\mathbf{x} = (x_1, \ldots, x_n)$ the vector of all covariates. This shows that neural networks are direct extensions of GLMs. However, the parameters, i.e. the weights, cannot be interpreted in the same way anymore.

Formally stated, all hidden neurons and output neurons calculate an output $f(g(z_0, z_1, \ldots, z_k)) = f(g(\mathbf{z}))$ from the outputs of all preceding neurons $z_0, z_1, \ldots, z_k$, where $g : \mathrm{R}^{k+1} \rightarrow \mathrm{R}$ denotes the integration function and $f : \mathrm{R} \rightarrow \mathrm{R}$ the activation function. The neuron $z_0 \equiv 1$ is the constant one belonging to the intercept. The integration function is often defined as $g(\mathbf{z}) = w_0 z_0 + \sum_{i=1}^{k} w_i z_i = w_0 + \mathbf{w}^\mathbf{T}\mathbf{z}$. The activation function f is usually a bounded nondecreasing nonlinear and differentiable function such as the logistic function ($f(u) = \frac{1}{1+e^{-u}}$) or the hyperbolic tangent. It should be chosen in relation to the response variable as it is the case in GLMs. The logistic function is, for instance, appropriate for binary response variables since it maps the output of each neuron to the interval $[0,1]$. At the moment, **neuralnet** uses the same integration as well as activation function for all neurons.

Supervised learning

Neural networks are fitted to the data by learning algorithms during a training process. **neuralnet** focuses on supervised learning algorithms. These learning algorithms are characterized by the usage of a given output that is compared to the predicted output and by the adaptation of all parameters according to this comparison. The parameters of a neural network are its weights. All weights are usually initialized with random values drawn from a standard normal distribution. During an iterative training process, the following steps are repeated:

- The neural network calculates an output $\mathbf{o}(\mathbf{x})$ for given inputs $\mathbf{x}$ and current weights. If the training process is not yet completed, the predicted output $\mathbf{o}$ will differ from the observed output $\mathbf{y}$.

- An error function E, like the sum of squared errors (SSE)

  $$E = \frac{1}{2} \sum_{l=1}^{L} \sum_{h=1}^{H} (o_{lh} - y_{lh})^2$$

  or the cross-entropy

  $$\begin{aligned} E = -\sum_{l=1}^{L} \sum_{h=1}^{H} & (y_{lh} \log(o_{lh}) \\ & + (1 - y_{lh}) \log(1 - o_{lh})), \end{aligned}$$

  measures the difference between predicted and observed output, where $l = 1, \ldots, L$ indexes the observations, i.e. given input-output pairs, and $h = 1, \ldots, H$ the output nodes.

- All weights are adapted according to the rule of a learning algorithm.

The process stops if a pre-specified criterion is fulfilled, e.g. if all absolute partial derivatives of the error function with respect to the weights ($\partial E/\partial w$) are smaller than a given threshold. A widely used learning algorithm is the resilient backpropagation algorithm.

Backpropagation and resilient backpropagation

The resilient backpropagation algorithm is based on the traditional backpropagation algorithm that modifies the weights of a neural network in order to find a local minimum of the error function. Therefore, the gradient of the error function ($dE/d\mathbf{w}$) is calculated with respect to the weights in order to find a root. In particular, the weights are modified going in the opposite direction of the partial derivatives until a local minimum is reached. This basic idea is roughly illustrated in Figure 2 for a univariate error-function.

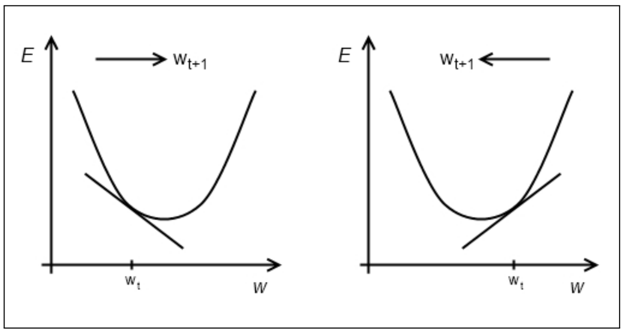


Figure 2: Basic idea of the backpropagation algorithm illustrated for a univariate error function $E(w)$.

If the partial derivative is negative, the weight is increased (left part of the figure); if the partial derivative is positive, the weight is decreased (right part of the figure). This ensures that a local minimum is reached. All partial derivatives are calculated using the chain rule since the calculated function of a neural network is basically a composition of integration and activation functions. A detailed explanation is given in Rojas (1996).

neuralnet provides the opportunity to switch between backpropagation, resilient backpropagation with (Riedmiller, 1994) or without weight backtracking (Riedmiller and Braun, 1993) and the modified globally convergent version by Anastasiadis et al. (2005). All algorithms try to minimize the error function by adding a learning rate to the weights going into the opposite direction of the gradient. Unlike the traditional backpropagation algorithm, a separate learning rate η_k, which can be changed during the training process, is used for each weight in resilient backpropagation. This solves the problem of defining an over-all learning rate that is appropriate for the whole training process and the entire network. Additionally, instead of the magnitude of the partial derivatives only their sign is used to update the weights. This guarantees an equal influence of the learning rate over the entire network (Riedmiller and Braun, 1993). The weights are adjusted by the following rule

$$w_k^{(t+1)} = w_k^{(t)} - \eta_k^{(t)} \cdot \text{sign}\left(\frac{\partial E^{(t)}}{\partial w_k^{(t)}}\right),$$

as opposed to

$$w_k^{(t+1)} = w_k^{(t)} - \eta \cdot \frac{\partial E^{(t)}}{\partial w_k^{(t)}},$$

in traditional backpropagation, where t indexes the iteration steps and k the weights.

In order to speed up convergence in shallow areas, the learning rate η_k will be increased if the corresponding partial derivative keeps its sign. On the contrary, it will be decreased if the partial derivative of the error function changes its sign since a changing sign indicates that the minimum is missed due to a too large learning rate. Weight backtracking is a technique of undoing the last iteration and adding a smaller value to the weight in the next step. Without the usage of weight backtracking, the algorithm can jump over the minimum several times. For example, the pseudocode of resilient backpropagation with weight backtracking is given by (Riedmiller and Braun, 1993)

```
for all weights{
    if (grad.old*grad>0){
        delta := min(delta*eta.plus, delta.max)
        weights := weights - sign(grad)*delta
        grad.old := grad
    }
    else if (grad.old*grad<0){
        weights := weights + sign(grad.old)*delta
        delta := max(delta*eta.minus, delta.min)
        grad.old := 0
    }
    else if (grad.old*grad=0){
        weights := weights - sign(grad)*delta
        grad.old := grad
    }
}
```

while that of the regular backpropagation is given by

```
for all weights{
    weights := weights - grad*delta
}
```

The globally convergent version introduced by Anastasiadis et al. (2005) performs a resilient backpropagation with an additional modification of one learning rate in relation to all other learning rates. It is either the learning rate associated with the smallest absolute partial derivative or the smallest learning rate (indexed with i), that is changed according to

$$\eta_i^{(t)} = -\frac{\sum_{k;k\neq i} \eta_k^{(t)} \cdot \frac{\partial E^{(t)}}{\partial w_k^{(t)}} + \delta}{\frac{\partial E^{(t)}}{\partial w_i^{(t)}}},$$

if $\frac{\partial E^{(t)}}{\partial w_i^{(t)}} \neq 0$ and $0 < \delta \ll \infty$. For further details see Anastasiadis et al. (2005).

Using neuralnet

neuralnet depends on two other packages: **grid** and **MASS** (Venables and Ripley, 2002). Its usage is leaned towards that of functions dealing with regression analyses like `lm` and `glm`. As essential arguments, a formula in terms of *response variables ~ sum of covariates* and a data set containing covariates and response variables have to be specified. Default values are defined for all other parameters (see next subsection). We use the data set `infert` that is provided by the package **datasets** to illustrate its application. This data set contains data of a case-control study that investigated infertility after spontaneous and induced abortion (Trichopoulos et al., 1976). The data set consists of 248 observations, 83 women, who were infertile (cases), and 165 women, who were not infertile (controls). It includes amongst others the variables `age`, `parity`, `induced`, and `spontaneous`. The variables `induced` and `spontaneous` denote the number of prior induced and spontaneous abortions, respectively. Both variables take possible values 0, 1, and 2 relating to 0, 1, and 2 or more prior abortions. The age in years is given by the variable `age` and the number of births by `parity`.

Training of neural networks

The function `neuralnet` used for training a neural network provides the opportunity to define the required number of hidden layers and hidden neurons according to the needed complexity. The complexity of the calculated function increases with the addition of hidden layers or hidden neurons. The default value is one hidden layer with one hidden neuron. The most important arguments of the function are the following:

- `formula`, a symbolic description of the model to be fitted (see above). No default.
- `data`, a data frame containing the variables specified in `formula`. No default.
- `hidden`, a vector specifying the number of hidden layers and hidden neurons in each layer. For example the vector (3,2,1) induces a neural network with three hidden layers, the first one with three, the second one with two and the third one with one hidden neuron. Default: 1.
- `threshold`, an integer specifying the threshold for the partial derivatives of the error function as stopping criteria. Default: 0.01.
- `rep`, number of repetitions for the training process. Default: 1.
- `startweights`, a vector containing prespecified starting values for the weights. Default: random numbers drawn from the standard normal distribution
- `algorithm`, a string containing the algorithm type. Possible values are `"backprop"`, `"rprop+"`, `"rprop-"`, `"sag"`, or `"slr"`. `"backprop"` refers to traditional backpropagation, `"rprop+"` and `"rprop-"` refer to resilient backpropagation with and without weight backtracking and `"sag"` and `"slr"` refer to the modified globally convergent algorithm (grprop). `"sag"` and `"slr"` define the learning rate that is changed according to all others. `"sag"` refers to the smallest absolute derivative, `"slr"` to the smallest learning rate. Default: `"rprop+"`
- `err.fct`, a differentiable error function. The strings `"sse"` and `"ce"` can be used, which refer to 'sum of squared errors' and 'cross entropy'. Default: `"sse"`
- `act.fct`, a differentiable activation function. The strings `"logistic"` and `"tanh"` are possible for the logistic function and tangent hyperbolicus. Default: `"logistic"`
- `linear.output`, logical. If `act.fct` should not be applied to the output neurons, `linear.output` has to be `TRUE`. Default: `TRUE`
- `likelihood`, logical. If the error function is equal to the negative log-likelihood function, `likelihood` has to be `TRUE`. Akaike's Information Criterion (AIC, Akaike, 1973) and Bayes Information Criterion (BIC, Schwarz, 1978) will then be calculated. Default: FALSE
- `exclude`, a vector or matrix specifying weights that should be excluded from training. A matrix with n rows and three columns will exclude n weights, where the first column indicates the layer, the second column the input neuron of the weight, and the third neuron the output neuron of the weight. If given as vector, the exact numbering has to be known. The numbering can be checked using the provided plot or the saved starting weights. Default: `NULL`
- `constant.weights`, a vector specifying the values of weights that are excluded from training and treated as fixed. Default: `NULL`

The usage of neuralnet is described by modeling the relationship between the case-control status (case) as response variable and the four covariates age, parity, induced and spontaneous. Since the response variable is binary, the activation function could be chosen as logistic function (default) and the error function as cross-entropy (err.fct="ce"). Additionally, the item linear.output should be stated as FALSE to ensure that the output is mapped by the activation function to the interval [0,1]. The number of hidden neurons should be determined in relation to the needed complexity. A neural network with for example two hidden neurons is trained by the following statements:

```
> library(neuralnet)
Loading required package: grid
Loading required package: MASS
>
> nn <- neuralnet(
+       case~age+parity+induced+spontaneous,
+       data=infert, hidden=2, err.fct="ce",
+       linear.output=FALSE)
> nn
Call:
  neuralnet(
  formula = case~age+parity+induced+spontaneous,
  data = infert, hidden = 2, err.fct = "ce",
  linear.output = FALSE)

1 repetition was calculated.

        Error Reached Threshold Steps
1 125.2126851    0.008779243419  5254
```

Basic information about the training process and the trained neural network is saved in nn. This includes all information that has to be known to reproduce the results as for instance the starting weights. Important values are the following:

- net.result, a list containing the overall result, i.e. the output, of the neural network for each replication.
- weights, a list containing the fitted weights of the neural network for each replication.
- generalized.weights, a list containing the generalized weights of the neural network for each replication.
- result.matrix, a matrix containing the error, reached threshold, needed steps, AIC and BIC (computed if likelihood=TRUE) and estimated weights for each replication. Each column represents one replication.
- startweights, a list containing the starting weights for each replication.

A summary of the main results is provided by nn$result.matrix:

```
> nn$result.matrix
                                          1
error                     125.212685099732
reached.threshold           0.008779243419
steps                    5254.000000000000
Intercept.to.1layhid1       5.593787533788
age.to.1layhid1            -0.117576380283
parity.to.1layhid1          1.765945780047
induced.to.1layhid1        -2.200113693672
spontaneous.to.1layhid1    -3.369491912508
Intercept.to.1layhid2       1.060701883258
age.to.1layhid2             2.925601414213
parity.to.1layhid2          0.259809664488
induced.to.1layhid2        -0.120043540527
spontaneous.to.1layhid2    -0.033475146593
Intercept.to.case           0.722297491596
1layhid.1.to.case          -5.141324077052
1layhid.2.to.case           2.623245311046
```

The training process needed 5254 steps until all absolute partial derivatives of the error function were smaller than 0.01 (the default threshold). The estimated weights range from -5.14 to 5.59. For instance, the intercepts of the first hidden layer are 5.59 and 1.06 and the four weights leading to the first hidden neuron are estimated as -0.12, 1.77, -2.20, and -3.37 for the covariates age, parity, induced and spontaneous, respectively. If the error function is equal to the negative log-likelihood function, the error refers to the likelihood as is used for example to calculate Akaike's Information Criterion (AIC).

The given data is saved in nn$covariate and nn$response as well as in nn$data for the whole data set inclusive non-used variables. The output of the neural network, i.e. the fitted values $o(\mathbf{x})$, is provided by nn$net.result:

```
> out <- cbind(nn$covariate,
+              nn$net.result[[1]])
> dimnames(out) <- list(NULL,
+                  c("age","parity","induced",
+                    "spontaneous","nn-output"))
> head(out)
     age parity induced spontaneous    nn-output
[1,]  26      6       1           2 0.1519579877
[2,]  42      1       1           0 0.6204480608
[3,]  39      6       2           0 0.1428325816
[4,]  34      4       2           0 0.1513351888
[5,]  35      3       1           1 0.3516163154
[6,]  36      4       2           1 0.4904344475
```

In this case, the object nn$net.result is a list consisting of only one element relating to one calculated replication. If more than one replication were calculated, the outputs would be saved each in a separate list element. This approach is the same for all values that change with replication apart from net.result that is saved as matrix with one column for each replication.

To compare the results, neural networks are trained with the same parameter setting as above using **neuralnet** with algorithm="backprop" and the package **nnet**.

```
> nn.bp <- neuralnet(
+          case~age+parity+induced+spontaneous,
+          data=infert, hidden=2, err.fct="ce",
+          linear.output=FALSE,
+          algorithm="backprop",
+          learningrate=0.01)
> nn.bp
Call:
  neuralnet(
  formula = case~age+parity+induced+spontaneous,
  data = infert, hidden = 2, learningrate = 0.01,
  algorithm = "backprop", err.fct = "ce",
  linear.output = FALSE)

1 repetition was calculated.

       Error Reached Threshold Steps
1 158.085556    0.008087314995     4
>
>
> nn.nnet <- nnet(
+            case~age+parity+induced+spontaneous,
+            data=infert, size=2, entropy=T,
+            abstol=0.01)
# weights:  13
initial  value 158.121035
final  value 158.085463
converged
```

nn.bp and nn.nnet show equal results. Both training processes last only a very few iteration steps and the error is approximately 158. Thus in this little comparison, the model fit is less satisfying than that achieved by resilient backpropagation.

neuralnet includes the calculation of generalized weights as introduced by Intrator and Intrator (2001). The generalized weight $\tilde{w}_i$ is defined as the contribution of the ith covariate to the log-odds:

$$\tilde{w}_i = \frac{\partial \log\left(\frac{o(\mathbf{x})}{1-o(\mathbf{x})}\right)}{\partial x_i}.$$

The generalized weight expresses the effect of each covariate x_i and thus has an analogous interpretation as the ith regression parameter in regression models. However, the generalized weight depends on all other covariates. Its distribution indicates whether the effect of the covariate is linear since a small variance suggests a linear effect (Intrator and Intrator, 2001). They are saved in nn$generalized.weights and are given in the following format (rounded values)

```
> head(nn$generalized.weights[[1]])
       [,1]       [,2]      [,3]      [,4]
1 0.0088556 -0.1330079 0.1657087 0.2537842
2 0.1492874 -2.2422321 2.7934978 4.2782645
3 0.0004489 -0.0067430 0.0084008 0.0128660
4 0.0083028 -0.1247051 0.1553646 0.2379421
5 0.1071413 -1.6092161 2.0048511 3.0704457
6 0.1360035 -2.0427123 2.5449249 3.8975730
```

The columns refer to the four covariates age ($j =$ 1), parity ($j = 2$), induced ($j = 3$), and spontaneous ($j = 4$) and a generalized weight is given for each observation even though they are equal for each covariate combination.

Visualizing the results

The results of the training process can be visualized by two different plots. First, the trained neural network can simply be plotted by

```
> plot(nn)
```

The resulting plot is given in Figure 3.

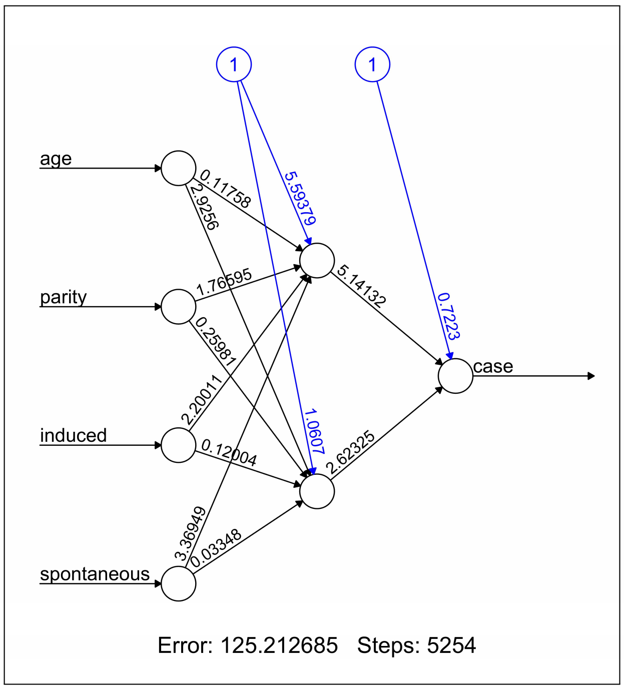


Figure 3: Plot of a trained neural network including trained synaptic weights and basic information about the training process.

It reflects the structure of the trained neural network, i.e. the network topology. The plot includes by default the trained synaptic weights, all intercepts as well as basic information about the training process like the overall error and the number of steps needed to converge. Especially for larger neural networks, the size of the plot and that of each neuron can be determined using the parameters dimension and radius, respectively.

The second possibility to visualize the results is to plot generalized weights. gwplot uses the calculated generalized weights provided by nn$generalized.weights and can be used by the following statements:

```
> par(mfrow=c(2,2))
> gwplot(nn,selected.covariate="age",
+        min=-2.5, max=5)
> gwplot(nn,selected.covariate="parity",
+        min=-2.5, max=5)
> gwplot(nn,selected.covariate="induced",
+       min=-2.5, max=5)
> gwplot(nn,selected.covariate="spontaneous",
+       min=-2.5, max=5)
```

The corresponding plot is shown in Figure 4.

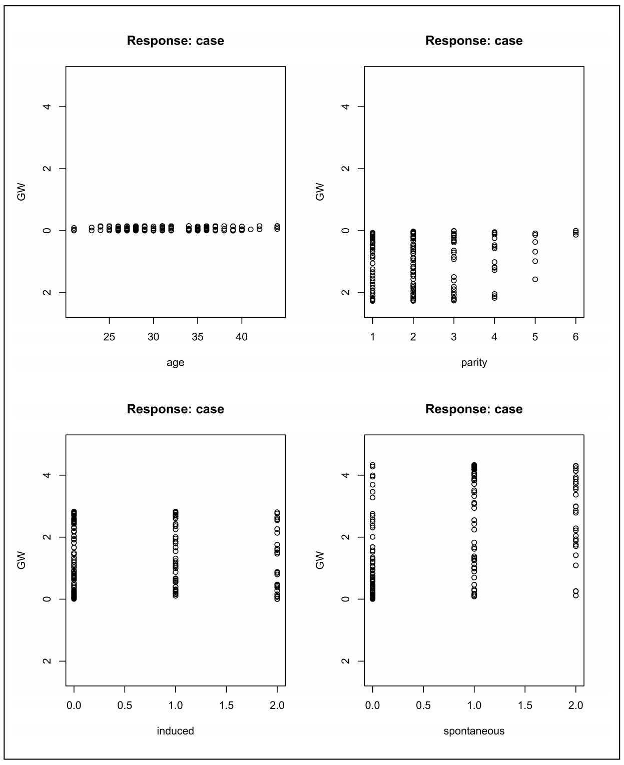


Figure 4: Plots of generalized weights with respect to each covariate.

The generalized weights are given for all covariates within the same range. The distribution of the generalized weights suggests that the covariate `age` has no effect on the case-control status since all generalized weights are nearly zero and that at least the two covariates `induced` and `spontaneous` have a non-linear effect since the variance of their generalized weights is overall greater than one.

Additional features

The `compute` function

`compute` calculates and summarizes the output of each neuron, i.e. all neurons in the input, hidden and output layer. Thus, it can be used to trace all signals passing the neural network for given covariate combinations. This helps to interpret the network topology of a trained neural network. It can also easily be used to calculate predictions for new covariate combinations. A neural network is trained with a training data set consisting of known input-output pairs. It learns an approximation of the relationship between inputs and outputs and can then be used to predict outputs $o(\mathbf{x}_{new})$ relating to new covariate combinations $\mathbf{x}_{new}$. The function `compute` simplifies this calculation. It automatically redefines the structure of the given neural network and calculates the output for arbitrary covariate combinations.

To stay with the example, predicted outputs can be calculated for instance for missing combinations with `age=22`, `parity=1`, `induced` $\leq$ `1`, and `spontaneous` $\leq$ `1`. They are provided by `new.output$net.result`

```
> new.output <- compute(nn,
                covariate=matrix(c(22,1,0,0,
                                   22,1,1,0,
                                   22,1,0,1,
                                   22,1,1,1),
                          byrow=TRUE, ncol=4))
> new.output$net.result
          [,1]
[1,] 0.1477097
[2,] 0.1929026
[3,] 0.3139651
[4,] 0.8516760
```

This means that the predicted probability of being a case given the mentioned covariate combinations, i.e. $o(\mathbf{x})$, is increasing in this example with the number of prior abortions.

The `confidence.interval` function

The weights of a neural network follow a multivariate normal distribution if the network is identified (White, 1989). A neural network is identified if it does not include irrelevant neurons neither in the input layer nor in the hidden layers. An irrelevant neuron in the input layer can be for instance a covariate that has no effect or that is a linear combination of other included covariates. If this restriction is fulfilled and if the error function equals the neagtive log-likelihood, a confidence interval can be calculated for each weight. The **neuralnet** package provides a function to calculate these confidence intervals regardless of whether all restrictions are fulfilled. Therefore, the user has to be careful interpreting the results.

Since the covariate `age` has no effect on the outcome and the related neuron is thus irrelevant, a new neural network (`nn.new`), which has only the three input variables `parity`, `induced`, and `spontaneous`, has to be trained to demonstrate the usage of `confidence.interval`. Let us assume that all restrictions are now fulfilled, i.e. neither the three input variables nor the two hidden neurons are irrelevant. Confidence intervals can then be calculated with the function `confidence.interval`:

```
> ci <- confidence.interval(nn.new, alpha=0.05)
> ci$lower.ci
[[1]]
[[1]][[1]]
             [,1]          [,2]
[1,]   1.830803796  -2.680895286
[2,]   1.673863304  -2.839908343
[3,]  -8.883004913 -37.232020925
```

```
[4,] -48.906348154 -18.748849335

[[1]][[2]]
              [,1]
[1,]  1.283391149
[2,] -3.724315385
[3,] -2.650545922
```

For each weight, `ci$lower.ci` provides the related lower confidence limit and `ci$upper.ci` the related upper confidence limit. The first matrix contains the limits of the weights leading to the hidden neurons. The columns refer to the two hidden neurons. The other three values are the limits of the weights leading to the output neuron.

Summary

This paper gave a brief introduction to multi-layer perceptrons and supervised learning algorithms. It introduced the package **neuralnet** that can be applied when modeling functional relationships between covariates and response variables. **neuralnet** contains a very flexible function that trains multi-layer perceptrons to a given data set in the context of regression analyses. It is a very flexible package since most parameters can be easily adapted. For example, the activation function and the error function can be arbitrarily chosen and can be defined by the usual definition of functions in R.

Acknowledgements

The authors thank Nina Wawro for reading preliminary versions of the paper and for giving helpful comments. Additionally, we would like to thank two anonymous reviewers for their valuable suggestions and remarks.

We gratefully acknowledge the financial support of this research by the grant PI 345/3-1 from the German Research Foundation (DFG).

Bibliography

H. Akaike. Information theory and an extension of the maximum likelihood principle. In Petrov BN and Csaki BF, editors, *Second international symposium on information theory*, pages 267–281. Academiai Kiado, Budapest, 1973.

C. Almeida, C. Baugh, C. Lacey, C. Frenk, G. Granato, L. Silva, and A. Bressan. Modelling the dsty universe i: Introducing the artificial neural network and first applications to luminosity and colour distributions. *Monthly Notices of the Royal Astronomical Society*, 402:544–564, 2010.

A. Anastasiadis, G. Magoulas, and M. Vrahatis. New globally convergent training scheme based on the resilient propagation algorithm. *Neurocomputing*, 64:253–270, 2005.

C. Bishop. *Neural networks for pattern recognition*. Oxford University Press, New York, 1995.

S. Fritsch and F. Günther. *neuralnet: Training of Neural Networks*. R Foundation for Statistical Computing, 2008. R package version 1.2.

F. Günther, N. Wawro, and K. Bammann. Neural networks for modeling gene-gene interactions in association studies. *BMC Genetics*, 10:87, 2009. `http://www.biomedcentral.com/1471-2156/10/87`.

K. Hornik, M. Stichcombe, and H. White. Multi-layer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

O. Intrator and N. Intrator. Interpreting neural-network results: a simulation study. *Computational Statistics & Data Analysis*, 37:373–393, 2001.

A. Kumar and D. Zhang. Personal recognition using hand shape and texture. *IEEE Transactions on Image Processing*, 15:2454–2461, 2006.

M. C. Limas, E. P. V. G. Joaquín B. Ordieres Meré, F. J. M. de Pisón Ascacibar, A. V. P. Espinoza, and F. A. Elías. *AMORE: A MORE Flexible Neural Network Package*, 2007. URL `http://wiki.r-project.org/rwiki/doku.php?id=packages:cran:amore`. R package version 0.2-11.

P. McCullagh and J. Nelder. *Generalized Linear Models*. Chapman and Hall, London, 1983.

M. Riedmiller. Advanced supervised learning in multi-layer perceptrons - from backpropagation to adaptive learning algorithms. *International Journal of Computer Standards and Interfaces*, 16:265–278, 1994.

M. Riedmiller and H. Braun. A direct method for faster backpropagation learning: the rprop algorithm. *Proceedings of the IEEE International Conference on Neural Networks (ICNN)*, 1:586–591, 1993.

M. Rocha, P. Cortez, and J. Neves. Evolutionary neural network learning. *Lecture Notes in Computer Science*, 2902:24–28, 2003.

R. Rojas. *Neural Networks*. Springer-Verlag, Berlin, 1996.

W. Schiffmann, M. Joost, and R. Werner. Optimization of the backpropagation algorithm for training multilayer perceptrons. Technical report, University of Koblenz, Insitute of Physics, 1994.

G. Schwarz. Estimating the dimension of a model. *Ann Stat*, 6:461–464, 1978.

D. Trichopoulos, N. Handanos, J. Danezis, A. Kalandidi, and V. Kalapothaki. Induced abortion and secondary infertility. *British Journal of Obstetrics and Gynaecology*, 83:645–650, 1976.

W. Venables and B. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. URL http://www.stats.ox.ac.uk/pub/MASS4. ISBN 0-387-95457-0.

H. White. Learning in artificial neural networks: a statistical perspective. *Neural Computation*, 1:425–464, 1989.

Frauke Günther
University of Bremen, Bremen Institute for Prevention Research and Social Medicine
guenther@bips.uni-bremen.de

Stefan Fritsch
University of Bremen, Bremen Institute for Prevention Research and Social Medicine

glmperm: A Permutation of Regressor Residuals Test for Inference in Generalized Linear Models

by Wiebke Werft and Axel Benner

Abstract We introduce a new R package called **glmperm** for inference in generalized linear models especially for small and moderate-sized data sets. The inference is based on the permutation of regressor residuals test introduced by Potter (2005). The implementation of **glmperm** outperforms currently available permutation test software as **glmperm** can be applied in situations where more than one covariate is involved.

Introduction

A novel permutation test procedure for inference in logistic regression with small- and moderate-sized datasets was introduced by Potter (2005) and showed good performance in comparison to exact conditional methods. This so-called permutation of regressor residuals (PRR) test is implemented in the R package **logregperm**. However, the application field is limited to logistic regression models. The new **glmperm** package offers an extension of the PRR test to generalized linear models (GLMs) especially for small and moderate-sized data sets. In contrast to existing permutation test software, the **glmperm** package provides a permutation test for situations in which more than one covariate is involved, e.g. if established covariates need to be considered together with the new predictor under test.

Generalized linear models

Let $\boldsymbol{Y}$ be a random response vector whose components are independently distributed with means $\boldsymbol{\mu}$. Furthermore, let the covariates $\boldsymbol{x}_1,...,\boldsymbol{x}_p$ be related to the components of $\boldsymbol{Y}$ via the generalized linear model

$$\begin{aligned} E(Y_i) &= \mu_i, && (1) \\ g(\mu_i) &= \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j, && (2) \\ Var(Y_i) &= \frac{\phi}{w_i} V(\mu_i), && (3) \end{aligned}$$

with $i = 1,...,n$, g a link function and $V(\cdot)$ a known variance function; ϕ is called the dispersion parameter and w_i is a known weight that depends on the underlying observations. The dispersion parameter is constant over observations and might be unknown, e.g. for normal distributions and for binomial and Poisson distributions with over-dispersion (see below).

One is now interested in testing the null hypothesis that the regression coefficient for a covariate of interest, say without loss of generality $\boldsymbol{x}_1$, is zero, i.e. $H_0 : \beta_1 = 0$ vs. $H_1 : \beta_1 \neq 0$. Let $\boldsymbol{y}$ be the observed vector of the outcome variable $\boldsymbol{Y}$. Inference is based on the likelihood-ratio test statistic $LR(\boldsymbol{y}; \boldsymbol{x}_1,...,\boldsymbol{x}_p)$, which is defined as the difference in deviances of the models with and without the covariate of interest divided by the dispersion parameter. For simplicity we assume that each variable is represented by a single parameter that describes the contribution of the variable to the model. Therefore, the test statistic has an asymptotic chi-squared distribution with one degree of freedom. (For a deeper look into generalized linear models McCullagh and Nelder (1989) is recommended.)

The PRR test

The basic concept of the PRR test is that it replaces the covariate of interest by its residual $\boldsymbol{r}$ from a linear regression on the remaining covariates $\boldsymbol{x}_2,...,\boldsymbol{x}_p$. This is a simple orthogonal projection of $\boldsymbol{x}_1$ on the space spanned by $\boldsymbol{x}_2,...,\boldsymbol{x}_p$ and ensures that $\boldsymbol{r}$ by its definition is not correlated with $\boldsymbol{x}_2,...,\boldsymbol{x}_p$ while $\boldsymbol{x}_1$ may be correlated. An interesting feature of this projection is that the maximum value of the likelihood for a generalized linear model of $\boldsymbol{y}$ on $\boldsymbol{r}, \boldsymbol{x}_2,...,\boldsymbol{x}_p$ is the same as that for $\boldsymbol{y}$ on $\boldsymbol{x}_1, \boldsymbol{x}_2,...,\boldsymbol{x}_p$. Hence, the likelihood-ratio test is the same when using the residuals $\boldsymbol{r}$ instead of the covariate $\boldsymbol{x}_1$. This leads to the idea of using permutations of the residuals $\boldsymbol{r}$ to estimate the null distribution and thus the p-value.

The algorithm for the PRR test to obtain a p-value for testing the null hypothesis $H_0 : \beta_1 = 0$ of the model

$$g(\boldsymbol{\mu}) = \beta_0 + \beta_1\boldsymbol{x}_1 + \beta_2\boldsymbol{x}_2 + ... + \beta_p\boldsymbol{x}_p \quad (4)$$

is as follows:

1. Calculate least squares residuals $\boldsymbol{r} = (r_1,...,r_n)$ via

$$\boldsymbol{r} = \boldsymbol{x}_1 - \left(\hat{\gamma}_0 + \hat{\gamma}_1\boldsymbol{x}_2 + ... + \hat{\gamma}_{p-1}\boldsymbol{x}_p\right) \quad (5)$$

   where $\hat{\gamma}_0, \hat{\gamma}_1,...,\hat{\gamma}_{p-1}$ are least squares estimates of the coefficients from the linear model

$$E(\boldsymbol{x}_1) = \gamma_0 + \gamma_1\boldsymbol{x}_2 + ... + \gamma_{p-1}\boldsymbol{x}_p. \quad (6)$$

   Then derive the p-value $\tilde{p}$ of the likelihood-ratio test statistic $LR(\boldsymbol{y}; \boldsymbol{r}, \boldsymbol{x}_2,...,\boldsymbol{x}_p)$ based on $\boldsymbol{r}$ replacing the covariate $\boldsymbol{x}_1$.

2. For resampling iterations $b = 1,...,B$
   - Randomly draw $\boldsymbol{r}^*$ from $\boldsymbol{r} = (r_1,...,r_n)$ without replacement.
   - Calculate p-values p_b^* of the likelihood-ratio test statistic $LR(\boldsymbol{y}; \boldsymbol{r}^*, \boldsymbol{x}_2,...,\boldsymbol{x}_p)$.
3. Calculate a permutation p-value for the PRR test according to

$$p_f = \frac{\#(p_b^* \leq f \cdot \tilde{p})}{B}. \tag{7}$$

Thus, the permutation p-value p_f is the fraction of permutations that have a likelihood-based p-value less than or equal to that for the unpermuted data times a factor f. This factor f is equal or slightly bigger than one, i.e. $f \in \{1; 1.005; 1.01; 1.02; 1.04\}$. It was introduced by Potter (2005) to account for numerical instabilities which might occur when the algorithms to maximize the likelihood do not converge to exactly the same value for different configurations of the data. Note that varying the factor only changes the results by a few per cent or less. In the R package **glmperm** various factors are implemented and are displayed in the summary output. Here, the result presentations will be restricted to factors 1 and 1.02.

The number of resampling iterations B of the algorithm is implemented as `nrep` in the function `prr.test`. By default `nrep=1000`. However, depending on the precision that is desired, the number of iterations could be increased at the cost of longer computation time. We recommend using `nrep=10000`.

In order to provide an estimate of the variance of the result, the permutation p-value p_f could be regarded as a binomial random variable $\mathcal{B}(B,p)$, where B is the number of resampling iterations and p is the unknown value of the true significance level (Good, 2005). As estimate of p the observed p-value $\tilde{p}$ is used multiplied by the factor f. Hence, the standard error for the permutation p-value p_f is provided by $\sqrt{f\tilde{p}(1-f\tilde{p})/B}$.

For binary outcome variables the PRR test is a competitive approach to exact conditional logistic regression described by Hirji et al. (1987) and Mehta and Patel (1995). The commercial LogXact software has implemented this conditional logistic regression approach. At present, statistical tests employing unconditional permutation methods are not commercially available and the package **glmperm** bridges this gap.

The rationale of exact conditional logistic regression is based on the exact permutation distribution of the sufficient statistic for the parameter of interest, conditional on the sufficient statistics for the specified nuisance parameters. However, its algorithmic constraint is that the distribution of interest will be degenerate if a conditioning variable is continuous. The difference between the two procedures lies in the permutation scheme: While the conditional approach permutes the outcome variable, the PRR test permutes the residuals from the linear model. Note that if one considers regressions with only a single regressor the residuals $\boldsymbol{r}$ are basically equal to $\boldsymbol{x}_1$, and the PRR test reduces to a simple permutation test (shuffle-Z method, see below).

An overview of the methodologic differences between these two and other permutation schemes is provided by Kennedy and Cade (1996). In the context of their work the permutation method used for the PRR test can be viewed as an extension of their shuffle-R permutation scheme for linear models. The variable associated with the parameter under the null hypothesis is regressed on the remaining covariables and replaced by the corresponding residuals; these residuals are then permuted while the response and the covariates are held constant. Freedman and Lane (1983) introduced the shuffle-R method in combination with tests of significance and a detailed discussion of this permutation scheme is provided by ter Braak (1992). The conditional method can be implied with the shuffle-Z method which was first mentioned in the context of multiple regression by Draper and Stoneman (1966). Here the variables associated with the parameters being tested under the null hypothesis are randomized while the response and the covariates are held constant. Kennedy and Cade (1996) discuss the potential pitfalls of the shuffle-Z method and point out that this method violates the ancillarity principle by not holding constant the collinearity between the covariables and the variable associated with the parameter under the null hypothesis. Therefore, Kennedy and Cade (1996) do not recommend the shuffle-Z method unless it employs a pivotal statistic or the hypothesized variable and the remaining covariables are known to be independent.

Modifications for the extension to GLMs

Three main modifications have been implemented in the **glmperm** package in comparison to the **logregperm** package. Note that the new package **glmperm** includes all features of the package **logregperm**. At present, the **logregperm** package is still available on CRAN. In general, the **glmperm** package could replace the **logregperm** package.

1. The extension of the `prr.test` function in the **glmperm** package provides several new arguments compared to the version in **logregperm**. The input is now organised as a formula expression equivalent to fitting a generalized linear model with `glm` (R package **stats**). Hence, for easier usage the syntax of `prr.test` is adapted from `glm`. The covariable of inter-

est about which inference is to be made is to be included as argument `var='x1'`. An argument `seed` is provided to allow for reproducibility.

2. The implicit function `glm.perm` is extended to calculate not only the deviances for the different resampling iterations but also the dispersion parameter for each permutation separately. For Poisson and logistic regression models the dispersion parameters are pre-defined as $\phi = 1$ for all iterations; the likelihood-ratio test statistic is then simply the difference of deviances. For all other GLM the dispersion parameters will be estimated for each resampling iteration based on the underlying data. This ensures that the p-value of the likelihood-ratio test statistic for this precise resampling iteration is accordingly specified given the data.

3. A new feature of the package is that it includes a summary function to view the main results of `prr.test`. It summarises the permutation of regressor residual-based p-value p_f for the various factors f, the observed likelihood-ratio test statistic and the observed p-value $\tilde{p}$ based on the chi-squared distribution with one degree of freedom. For Poisson and logistic regression models a warning occurs in case of possible overdispersion ($\phi > 1.5$) or underdispersion ($\phi < 0.5$) and recommends use of `family=quasibinomial()` or `quasipoisson()` instead.

An example session

To illustrate the usage of the **glmperm** package for a GLM, an example session with simulated data is presented. First, we simulated data for three independent variables with $n = 20$ samples and binary and discrete response variables for the logistic and Poisson regression model, respectively.

```
# binary response variable
n <- 20
set.seed(4278)
x1 <- rnorm(n)
x0 <- rnorm(n)+x1
y1 <- ifelse(x0+x1+2*rnorm(n)>0,1,0)
test1 <- prr.test(y1~x0+x1,
          var="x0", family=binomial())
x2 <- rbinom(n,1,0.6)
y2 <- ifelse(x1+x2+rnorm(n)>0,1,0)
test2 <- prr.test(y2~x1+x2, var="x1",
          nrep=10000,family=binomial())

# Poisson response variable
set.seed(4278)
x1 <- rnorm(n)
x0 <- rnorm(n) + x1
nu <- rgamma(n, shape = 2, scale = 1)
y <- rpois(n, lambda = exp(2) * nu)
test3 <- prr.test(y~x0+x1,
          var="x0", family=poisson())
test4 <- prr.test(y~x0, var="x0",
          nrep=1000,family=poisson())
```

A condensed version of the displayed result summary of `test2` (only factors $f = 1$ and $f = 1.02$ are shown) is given by:

```
> summary(test2)
-----------------------------------------
Results based on chi-squared distribution
-----------------------------------------
observed p-value: 0.0332
--------------------
Results based on PRR
--------------------
permutation p-value for simulated p-values <=
observed p-value: 0.0522 (Std.err: 0.0018)
permutation p-value for simulated p-values <=
1.02 observed p-value: 0.0531 (Std.err: 0.0018)
```

For the above example `test2` the exact conditional logistic regression p-value calculated via LogXact-4 is 0.0526, whereas the p-value of the PRR test is 0.0522 for factor $f = 1$, and based on the chi-squared distribution it is 0.0332. The example demonstrates that the p-value obtained via PRR test (or LogXact) leads to a different rejection decision than a p-value calculated via an approximation by the chi-squared distribution. Hence, when small sample sizes are considered the PRR test should be preferred over an approximation via chi-squared distribution.

Special case: Overdispersion

For the computation of GLM the dispersion parameter ϕ for the binomial and Poisson distribution is set equal to one. However, there exist cases of data distribution which violate this assumption. The variance in (3) can then be better described when using a dispersion parameter $\phi \neq 1$. The case of $\phi > 1$ is known as overdispersion as the variance is larger than expected under the assumption of binomial or Poisson distribution. In practice, one can still use the algorithms for generalized linear models if the estimation of the variance takes account of the dispersion parameter $\phi > 1$. As a consequence of overdispersion the residual deviance is then divided by the estimation $\hat{\phi}$ of the dispersion factor instead of $\phi = 1$. Hence, this has direct influence on the likelihood-ratio test statistic which is the difference in deviances and therefore is also scaled by $\hat{\phi}$. The corresponding p-values differ if overdispersion is considered or not, i.e. if $\phi = 1$ or $\phi = \hat{\phi}$ is used. In the PRR test one can account for overdispersion when using `family=quasipoisson()` or quasibinomial() instead of `family=poisson()` or binomial().

We experienced a stable performance of the PRR test for overdispersed data. The following treepipit data (Müller and Hothorn, 2004) provides an example of overdispersed data. We show the results of the chi-squared approximation of the likelihood-ratio test statistic as well as the results of the PRR test for the usage of `family=poisson()` and `family=quasipoisson()`, respectively.

```
# example with family=poisson()
data(treepipit, package="coin")
test5<-prr.test(counts~cbpiles+coverstorey
 +coniferous+coverregen,data=treepipit,
 var="cbpiles",family=poisson())
summary(test5)
-----------------------------------------
Results based on chi-squared distribution
-----------------------------------------
observed p-value: 0.0037
--------------------
Results based on PRR
--------------------
permutation p-value for simulated p-values <=
observed p-value: 0.083 (Std.err: 0.0019)
permutation p-value for simulated p-values <=
1.02 observed p-value: 0.084 (Std.err: 0.0019)

# example with family=quasipoisson()
test6<-prr.test(counts~cbpiles+coverstorey
 +coniferous+coverregen,data=treepipit,
 var="cbpiles",family=quasipoisson())
summary(test6)
-----------------------------------------
Results based on chi-squared distribution
-----------------------------------------
observed p-value: 0.0651
--------------------
Results based on PRR
--------------------
permutation p-value for simulated p-values <=
observed p-value: 0.07 (Std.err: 0.0078)
permutation p-value for simulated p-values <=
1.02 observed p-value: 0.071 (Std.err: 0.0079)
```

The p-values based on the chi-squared distribution of the likelihood-ratio test statistic are $p = 0.0037$ and $p = 0.0651$ when using `family=poisson()` and `family=quasipoisson()`, respectively. Hence, a different test decision is made whereas the test decision for the PRR test is the same for both cases ($p = 0.083$ and $p = 0.07$).

Summary

The **glmperm** package provides a permutation of regressor residuals test for generalized linear models. This version of a permutation test for inference in GLMs is especially suitable for situations in which more than one covariate is involved in the model. The key input feature of the PRR test is to use the orthogonal projection of the variable of interest on the space spanned by all other covariates instead of the variable of interest itself. This feature provides a reasonable amendment to existing permutation test software which do not incorporate situations with more than one covariate. Applications to logistic and Poisson models show good performance when compared to gold standards. For the special case of data with overdispersion the PRR test is more robust compared to methods based on approximations of the test statistic distribution.

Acknowledgements

We would like to acknowledge the many valuable suggestions made by two anonymous referees.

Bibliography

C.J.F. ter Braak. Permutation versus bootstrap significance tests in multiple regression and anova. In K.H. Jöckel, G. Rothe and W. Sendler, editors. *Bootstraping and Related Techniques*, 79-85, Springer, 1992.

N.R. Draper and D.M. Stoneman. Testing for the inclusion of variables in linear regression by a randomization technique. *Technometrics*, **8**: 695-698, 1966.

D. Freedman and D. Lane. A nonstochastic interpretation of reported significance levels. *Journal of Business and Economic Statistics*, **1**:292-298, 1983.

P. Good. *Permutation, Parametric, and Bootstrap Tests of Hypotheses, 3rd edn.* Springer series in statistics, 2005. ISBN 0-387-20279-X.

K.F. Hirji, C.R. Mehta and N.R. Patel. Computing distributions for exact logistic regression. *Journal of the American Statistical Association*, **82**:1110-1117, 1987.

P.E. Kennedy and B.S. Cade. Randomization tests for multiple regression. *Communications in Statistics - Simulation and Computation*, **25(4)**:923-936, 1996.

D.M. Potter. A permutation test for inference in logistic regression with small- and moderate-sized datasets, *Statistics in Medicine*, **24**:693-708, 2005.

P. McCullagh and J.A. Nelder. *Generalized Linear Models, 2nd edn.* Chapman and Hall, London, 1989. ISBN 0-412-31760-5.

C.R. Mehta and N.R. Patel Exact logistic regression: theory and examples. *Statistics in Medicine*, **14**:2143-2160, 1995.

J. Müller and T. Hothorn. Maximally selected two-sample statistics as a new tool for the identification and assessment of habitat factors with an application to breeding bird communities in oak forests, *European Journal of Forest Research*, **123**:219-228, 2004.

Wiebke Werft
German Cancer Research Center
Im Neuenheimer Feld 280, 69120 Heidelberg
Germany
w.werft@dkfz.de

Axel Benner
German Cancer Research Center
Im Neuenheimer Feld 280, 69120 Heidelberg
Germany
benner@dkfz.de

Online Reproducible Research: An Application to Multivariate Analysis of Bacterial DNA Fingerprint Data

by Jean Thioulouse, Claire Valiente-Moro and Lionel Zenner

Abstract This paper presents an example of online reproducible multivariate data analysis. This example is based on a web page providing an online computing facility on a server. HTML forms contain editable R code snippets that can be executed in any web browser thanks to the Rweb software. The example is based on the multivariate analysis of DNA fingerprints of the internal bacterial flora of the poultry red mite *Dermanyssus gallinae*. Several multivariate data analysis methods from the **ade4** package are used to compare the fingerprints of mite pools coming from various poultry farms. All the computations and graphical displays can be redone interactively and further explored online, using only a web browser. Statistical methods are detailed in the duality diagram framework, and a discussion about online reproducibility is initiated.

Introduction

Reproducible research has gained much attention recently (see particularly `http://reproducibleresearch.net/` and the references therein). In the area of Statistics, the availability of Sweave (Leisch (2002), `http://www.stat.uni-muenchen.de/~leisch/Sweave/`) has proved extremely useful and Sweave documents are now widely used. Sweave offers the possibility to have text, R code, and outputs of this code in the same document. This makes reproducibility of a scientific paper written in Sweave very straightforward.

However, using Sweave documents implies a good knowledge of R, of Sweave and of LaTeX. It also requires having R installed on one's computer. The installed version of R must be compatible with the R code in the Sweave document, and all the needed packages must also be installed. This may be a problem for some scientists, for example for many biologists who are not familiar with R and LaTeX, or who do not use them on a regular basis.

In this paper, we demonstrate an online reproducibility system that helps circumvent these problems. It is based on a web page that can be used with any web browser. It does not require R to be installed locally, nor does it need a thorough knowledge of R and LaTeX. Nevertheless it allows the user to present and comment R code snippets, to redo all the computations and to draw all the graphical displays of the original paper.

The example presented here relates to multivariate data analysis. Reproducibility of multivariate data analyses is particularly important because there is a large number of different methods, with poorly defined names, making it often difficult to know what has been done exactly. Many methods have several different names, according to the community where they were invented (or re-invented), and where they are used. Translation of technical terms into languages other than English is also difficult, as many are ambiguous and have "false friends". Moreover, data analysis methods make intensive use of graphical displays, with a large number of graphical parameters which can be changed to produce a wide variety of displays.

This situation is similar to what was described by Buckheit and Donoho (1995) in the area of wavelet research. Their solution was to publish the Matlab code used to produce the figures in their research articles. Rather than do this we decided to set up a simple computer environment using R to offer online reproducibility. In 2002, we installed an updated version of the Rweb system (Banfield, 1999) on our department server (see `http://pbil.univ-lyon1.fr/Rweb/Rweb.general.html`), and we implemented several computational web services in the field of comparative genomics (Perrière et al. (2003), see for example `http://pbil.univ-lyon1.fr/mva/coa.php`).

This server now combines the computational power of R with simple HTML forms (as suggested by de Leeuw (2001) for Xlisp-Stat) and the ability to search online molecular databases with the **seqinr** package (Charif and Lobry, 2007). It is also used by several researchers to provide an online reproducibility service for scientific papers (see for example `http://pbil.univ-lyon1.fr/members/lobry/`).

The present paper provides an example of an application of this server to the analysis of DNA fingerprints by multivariate analysis methods, using the **ade4** package (Chessel et al., 2004; Dray and Dufour, 2007). We have shown recently (Valiente-Moro et al., 2009) that multivariate analysis techniques can be used to analyse bacterial DNA fingerprints and that they make it possible to draw useful conclusions about the composition of the bacterial communities from which they originate. We also demonstrate the effectiveness of principal component analysis, between-group analysis and within-group analysis [PCA, BGA and WGA, Benzécri (1983); Dolédec and Chessel (1987)] to show differences in diversity between bacterial communities of various origins.

In summary, we show here that it is easy to set up a software environment offering full online reproducibility of computations and graphical displays of multivariate data analysis methods, even for users who are not familiar with R, Sweave or LaTeX.

Data and methods

In this section, we first describe the biological material used in the example data set. Then we present the statistical methods (PCA, BGA and WGA), in the framework of the duality diagram (Escoufier, 1987; Holmes, 2006). We also detail the software environment that was used.

Biological material

The poultry red mite, *Dermanyssus gallinae* is an haematophagous mite frequently present in breeding facilities and especially in laying hen facilities (Chauve, 1998). This arthropod can be responsible for anemia, dermatitis, weight loss and a decrease in egg production (Kirkwood, 1967). It has also been involved in the transmission of many pathogenic agents responsible for serious diseases in both animals and humans (Valiente Moro et al., 2005; Valiente-Moro et al., 2007). The poultry red mite is therefore an emerging problem that must be studied to maintain good conditions in commercial egg production facilities. Nothing is known about its associated non-pathogenic bacterial community and how the diversity of the microflora within mites may influence the transmission of pathogens.

Most studies on insect microflora are based on isolation and culture of the constituent microorganisms. However, comparison of culture-based and molecular methods reveals that only 20-50% of gut microbes can be detected by cultivation (Suau et al., 1999). Molecular methods have been developed to analyse the bacterial community in complex environments. Among these methods, Denaturing Gradient and Temporal Temperature Gel Electrophoresis (DGGE and TTGE) (Muyzer, 1998) have already been used successfully.

Full details of the example data set used in this paper (mites sampling, DNA extraction, PCR amplification of 16S rDNA fragments and TTGE banding pattern achieving) are given in Valiente-Moro et al. (2009). Briefly, 13 poultry farms were selected in the Bretagne region in France, and in each farm 15 single mites, five pools of 10 mites and one pool of 50 mites were collected. The results for single mites and mite pools were analysed separately, but only the results of mite pools are presented in this study, as they had the most illustrative analysis. Banding patterns can be analysed as quantitative variables (intensity of bands), or as binary indicators (presence/absence of bands), but for the same reason, only the presence/absence data were used in this paper.

TTGE banding patterns were collected in a data table, with bands in columns (55 columns) and mite pools in rows (73 rows). This table was first subjected to a principal component analysis (PCA) to get an overall idea of the structure of the banding patterns. Between-group analysis (BGA) was then applied to study the differences between poultry farms, and finally within-group analysis (WGA) was used to eliminate the farm effect and obtain the main characteristics of the common bacterial flora associated with *D. gallinae* in standard breeding conditions. A good knowledge of these characteristics would allow comparisons of the standard bacterial flora among various situations, such as geographic regions, type and location of breeding facilities (particularly organic farms), or developmental stages of *D. gallinae*.

Duality diagram of principal component analysis

Let $\mathbf{X} = [x_{ij}]_{(n,p)}$ be the TTGE data table with n rows (individuals = mite pools) and p columns (variables = TTGE bands). Variables have mean $\bar{x}_j = \frac{1}{n}\sum_i x_{ij}$ and variance $\sigma_j^2 = \frac{1}{n}\sum_i (x_{ij} - \bar{x}_j)^2$. Individuals belong to g groups (or classes), namely $G_1, \dots, G_g$, with group counts $n_1, \dots, n_g$, and $\sum n_k = n$.

Using duality diagram theory and triplet notation, the PCA of $\mathbf{X}$ is the analysis of a triplet $(\mathbf{X}_0, \mathbf{D}_p, \mathbf{D}_n)$. $\mathbf{X}_0$ is the table of standardized values:

$$\mathbf{X}_0 = [\tilde{x}_{ij}]_{(n,p)}$$

with $\tilde{x}_{ij} = \frac{x_{ij} - \bar{x}_j}{\sigma_j}$, and $\mathbf{D}_n$ and $\mathbf{D}_p$ are the diagonal matrices of row and column weights: $\mathbf{D}_p = \mathbf{I}_p$ and $\mathbf{D}_n = \frac{1}{n}\mathbf{I}_n$.

This information is summarized in the following mnemonic diagram, called the "duality diagram" because R^{p^*} is the dual of R^p and R^{n^*} is the dual of R^n:

$$\begin{tikzcd}
\boxed{\mathrm{R}^p} \arrow[r, "\mathbf{D}_p"] & \boxed{\mathrm{R}^{p^*}} \arrow[d, "\mathbf{X}_0"] \\
\boxed{\mathrm{R}^{n^*}} \arrow[u, "\mathbf{X}_0^T"] & \boxed{\mathrm{R}^n} \arrow[l, "\mathbf{D}_n"]
\end{tikzcd}$$

$\mathbf{X}_0^T$ is the transpose of $\mathbf{X}_0$. The analysis of this triplet leads to the diagonalization of matrix:

$$\mathbf{X}_0^T \mathbf{D}_n \mathbf{X}_0 \mathbf{D}_p$$

i.e., the matrix obtained by proceeding counterclockwise around the diagram, starting from R^p.

Principal components, (variable loadings and row scores), are computed using the eigenvalues and eigenvectors of this matrix.

Between-group analysis

The between-group analysis (BGA) is the analysis of triplet $(\mathbf{X}_B, \mathbf{D}_p, \mathbf{D}_{n_k})$, where $\mathbf{X}_B$ is the (g, p) matrix of group means:

$$\mathbf{X}_B = [\bar{x}_j^k]_{(g,p)}.$$

The term $\bar{x}_j^k = \frac{1}{n_k}\sum_{i \in G_k} \tilde{x}_{ij}$ is the mean of variable j in group k. In matrix notation, if $\mathbf{B}$ is the matrix of class indicators: $\mathbf{B} = [b_i^k]_{(n,g)}$, with $b_i^k = 1$ if $i \in G_k$ and $b_i^k = 0$ if $i \notin G_k$, then we have:

$$\mathbf{X}_B = \mathbf{D}_{n_k} \mathbf{B}^T \mathbf{X}_0.$$

Matrix $\mathbf{D}_{n_k} = Diag(\frac{1}{n_k})$ is the diagonal matrix of (possibly non uniform) group weights, and $\mathbf{B}^T$ is the transpose of $\mathbf{B}$.

BGA is therefore the analysis of the table of group means, leading to the diagonalization of matrix $\mathbf{X}_B^T \mathbf{D}_{n_k} \mathbf{X}_B \mathbf{D}_p$. Its aim is to reveal any evidence of differences between groups. The statistical significance of these differences can be tested with a permutation test. Row scores of the initial data table can be computed by projecting the rows of the standardized table $\mathbf{X}_0$ onto the principal components subspaces.

Within-group analysis

The within-group analysis (WGA) is the analysis of triplet $(\mathbf{X}_W, \mathbf{D}_p, \mathbf{D}_n)$, where $\mathbf{X}_W$ is the (n, p) matrix of the differences between the standardized values and the group means:

$$\mathbf{X}_W = [\tilde{x}_{ij} - \bar{x}_{ij}^k]_{(n,p)}$$

with $\bar{x}_{ij}^k = \bar{x}_j^k, \forall i \in G_k$. In matrix notation

$$\mathbf{X}_W = \mathbf{X}_0 - \mathbf{X}_{\tilde{B}}$$

where $\mathbf{X}_{\tilde{B}}$ is the matrix of groups means, repeated inside each group.

WGA is therefore the analysis of the matrix of the residuals obtained by eliminating the between-group effect. It leads to the diagonalization of matrix $\mathbf{X}_W^T \mathbf{D}_n \mathbf{X}_W \mathbf{D}_p$. It is useful when looking for the main features of a data table after removing an unwanted characteristic.

Software

We used R version 2.11.0 (R Development Core Team, 2009) for all computations, and the **ade4** package (version 1.4-14) for multivariate analyses (Chessel et al., 2004; Dray and Dufour, 2007; Thioulouse and Dray, 2007). PCA, BGA and WGA were computed with the `dudi.pca`, `between` and `within` functions of **ade4**. BGA permutation tests were done with the `randtest` function.

Online reproducibility

The statistical analyses presented in Valiente-Moro et al. (2009) are reproducible (Gentleman and Lang, 2004) via the web page `http://pbil.univ-lyon1.fr/TTGE/`.

This page presents the problem, gives access to the data set, and allows the user to execute R code snippets that reproduce all the computations and graphical displays of the original article (Valiente-Moro et al., 2009). The R code is explained, with a description of important variables and function calls, and the R outputs are documented. Links to the information pages of the main **ade4** functions are also provided.

This page is therefore an example of *online* literate programming (Knuth, 1992), and also of an online *dynamic document* (Gentleman and Lang, 2004). This is made possible using HTML forms containing R code stored in editable text fields. The code itself can be modified by the user and executed on demand. Code execution is achieved by sending it to the Rweb software (Banfield, 1999) running on our department server: `http://pbil.univ-lyon1.fr/Rweb/Rweb.general.html`.

Links to the complete R code and data sets are provided on the web page. They can be used to download these files and use them locally if desired. The full R code (the collection of all code snippets) is available directly from `http://pbil.univ-lyon1.fr/TTGE/allCode.R`.

PCA, BGA and permutation test

The first code snippet in the reproducibility page reads the TTGE data table, builds the factor describing the poultry farms, and computes the three analyses (PCA, BGA, and WGA). The PCA is first done using the `dudi.pca` function of the **ade4** package, and the resulting object is passed to the `between` and `within` functions to compute BGA and WGA respectively. The BGA permutation test is then computed and the test output is plotted.

On the reproducibility web page, these successive steps are explained: links to the data files are presented, the R code is displayed and commented, and the "Do it again!" button can be used to redo the computations. The code can be freely modified by the user and executed again. For example, it is possible to change the scaling of the PCA, or the number of random permutations in the Monte Carlo test to check the influence of these parameters on analysis outputs.

Figure 1 shows the result of the permutation test of the BGA. The null hypothesis is that there is no difference between farms. The test checks that the observed value of the ratio of between-group to total inertia (0.67) is much higher than expected under the null hypothesis. Under the null hypothesis, mite pools can be permuted randomly among farms without changing significantly the ratio of between to total inertia. To compute the test, the rows of the dataframe are randomly permuted, and the ratio is computed again. This is done many times, to get an idea of the distribution of the between to total inertia ratio. Figure 1 shows that the observed value (black diamond, far on the right) is much higher than all the values obtained after permuting the pools. The p-value is 0.001, implying that the hypothesis of no differences between the farms can confidently be rejected.

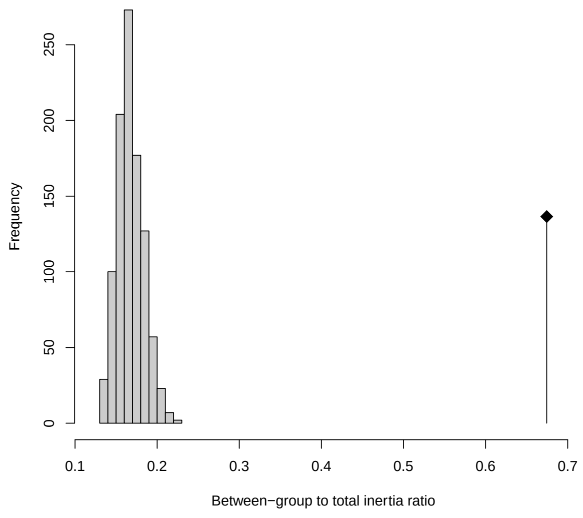


Figure 1: Permutation test of the BGA. The observed value of the between-group to total inertia ratio is equal to 0.67 (black diamond on the right). The histogram on the left shows the distribution of 1000 values of this ratio obtained after permuting the rows of the data table.

BGA plots

The second code snippet draws Figure 2, showing the factor maps of the BGA.

The loadings of TTGE bands are in the `bga1$co` dataframe, and they are plotted using the `s.label` function (first panel). To get an idea of the dispersion of the six mite pools in each farm, we can plot the projection of each pool on the factor map (second panel). The row scores of the pools are in the `bga1$ls` dataframe, and two graphs are superimposed: the graph of pool stars (with `s.class`), and the graph of convex hulls surrounding the pools belonging to the same farm (with `s.chull`). We can see that, as the permutation test had just evidenced, the farms are indeed very different.

Here also, the user can very easily change the R code to draw alternative graphs. It is possible, for example, to explore the effect of the arguments to the `s.label` and `s.class` functions.

Interpreting the differences between farms evidenced in Figure 2 was not easy. All 13 farms are standard poultry farms, using exactly the same breeding conditions, and the differences between TTGE banding patterns could not be attributed to any other factors.

Since the aim of the study was to find the common bacterial flora associated with *D. gallinae* in standard breeding conditions, we decided to remove this unwanted between-farm effect by doing a within-group analysis.

WGA and cluster analysis plots

The third code snippet draws Figure 3, showing the factor maps of WGA and the dendrogram of the cluster analysis on TTGE band loadings.

The loadings of the 55 TTGE bands are in the `wga1$co` dataframe, and they are plotted using the `s.label` function (top-left panel). The scores of the 73 mite pools are in the `wga1$li` dataframe. They are grouped by convex hulls, according to the poultry farm from which they originate using the `s.class` and `s.chull` functions (top-right panel). The row scores of the WGA are centered by group, so the 13 farms are centered on the origin (this corresponds to the fact that the "farm effect" has been removed in this analysis).

The TTGE bands corresponding to the common dominant bacterial flora associated with *D. gallinae* in standard breeding conditions were selected on Figure 3 (top-left panel), using cluster analysis (lower panel). This was done using the complete linkage algorithm, with euclidean distances computed on WGA variable loadings on the first three axes (wga1$co) and not on raw data.

The reproducibility page can be used to compare these results and the ones obtained with other clustering algorithms or other distance measures. Another possibility is to check the effect of the number of axes on which variable loadings are computed (three axes in the R code proposed by default).

We included the two leftmost outgroups (bands 0.15, 0.08, 0.11, 0.51, and 0.88) and the right group (0.38, 0.75, 0.03, 0.26, 0.41, 0.59, 0.13, 0.32, 0.23, 0.21, 0.31, 0.28, and 0.3). Band 0.39 was excluded because it was present in only two mite pools belonging to one single farm.

These results show that there is a strong between-farm effect in the TTGE banding patterns corresponding to the bacterial flora associated with *D. gallinae* in standard breeding conditions. However, it was not possible to give a satisfying biological interpretation of this effect: the farms evidenced by the BGA did not show any particular characteristic. The

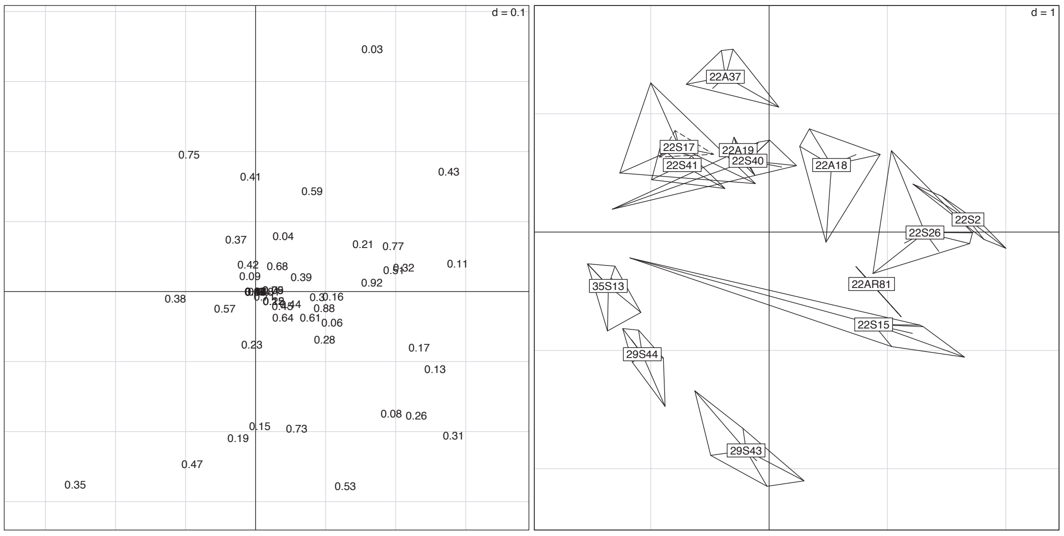


Figure 2: Factor maps of BGA (x-axis=first principal component, y-axis=second principal component, inertia percentages: 20% and 17%). The scale is given by the value d (top-right corner) that represents the size of the background grid. The first panel shows the map of the 55 TTGE bands (labels correspond to the position of the band on the electrophoresis gel). The second panel shows the 73 mite pools, grouped by convex hulls according to the poultry farm from which they originate (labels correspond to the farms).

WGA could remove this farm effect, and revealed the common dominant bacterial flora (Valiente-Moro et al., 2009). The knowledge of this flora will allow us to compare the variations observed in different conditions, such as different geographic regions, different types of breeding farms (for example organic farms), or different developmental stages of *D. gallinae*. Data from farms in different geographic regions and from organic farms are already available, and a paper presenting the results of their analysis has been submitted for publication.

Discussion

Statistical methodology

From the point of view of statistical methodology, BGA appears as a robust method that can be used to reveal any evidence for, and test, a simple effect, namely the effect of a single factor, in a multivariate data table. Many recent examples of its use can be found in the area of Genomics (Culhane et al., 2002, 2005; Baty et al., 2005; Jeffery et al., 2007).

Furthermore WGA can eliminate a simple effect from a multivariate data table. Like BGA, it can be used even when the number of cases is less than the number of variables. A recent example of WGA in the area of Genomics can be found in Suzuky et al. (2008).

BGA can be seen as a particular case of redundancy analysis [RDA, Stewart and Love (1968)] and WGA as a particular case of partial RDA [CANOCO (ter Braak and Šmilauer, 2002)]. Both cases correspond to covariates reduced to a single dummy variable.

This can be checked with **vegan**, another classical R package for multivariate ecological data analysis, using the `rda` function, as explained on the online reproducibility page in code snippet 4. Outputs are not presented here, but they are available on that site, where they may be interactively explored.

Computer setup

The reproducibility web page presented here is just an example of use of a simple setup that can be implemented easily on any web server. The current configuration of our server is an old Sun Fire 800, with 8 UltraSparc III processors at 900 MHz, 28 GB of memory and 6 x 36 GB disks. From the point of view of software, the server is running Solaris 10, the HTTP server is Apache 2.2.13, and we use Rweb 1.03 and R 2.11.0. Rweb is mainly a set of CGI scripts written in Perl, and the installed Perl version is v5.8.4. The page itself is written in plain HTML code, using standard HTML forms to communicate with Rweb CGI scripts.

Rweb source code is available from the Montana State University (`http://bayes.math.montana.edu/Rweb/Rweb1.03.tar.gz`).

There has been many other attempts at mixing R and HTML on the web. Alternative solutions could for example make use of **CGIwithR** (Firth, 2003) and **R2HTML** (Lecoutre, 2003), or Rscript.

Running a server offering public computing services necessarily raises many security issues. The server that is used for Rweb at the PBIL (`http://`

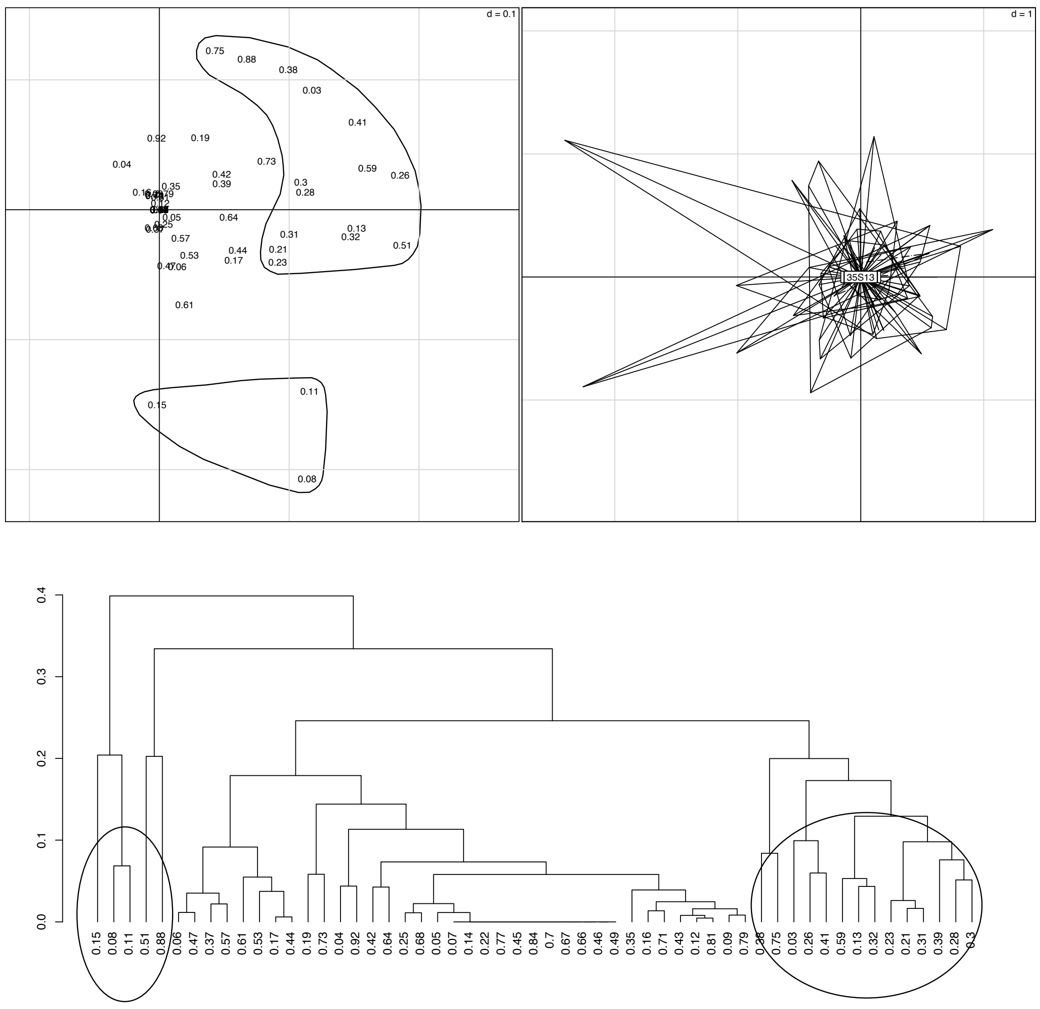


Figure 3: Factor maps of WGA (x-axis=first principal component, y-axis=second principal component, inertia percentages: 14 and 8). The scale is given by the value d (top-right corner) that represents the size of the background grid. Top-left panel: map of the 55 TTGE bands (see legend to Figure 2). Top-right panel: map of the 73 mite pools, grouped by farms. Bottom panel: dendrogram of the cluster analysis on WGA band loadings.

`pbil.univ-lyon1.fr`) also offers many other computing services in the area of Genomic research, including complex database exploration and analysis of huge molecular databases (GenBank, EMBL, etc.) This server has been attacked and successfully compromised several times with common rootkits, but in eight years [we started the Rweb service in 2002, Perrière et al. (2003)], the Rweb server has never been used in these attacks. Rweb precludes the use of the most sensitive functions (particularly "system", "eval", "call", "sink", etc.) and an additional security measure taken on our server is that all computations are performed in a temporary directory that is cleaned up automatically after use.

Online reproducibility

Sweave is a very good solution for the reproducibility of scientific papers' statistical computation and graphical display. It is more and more frequently used, and not only for reproducible research in the strictest sense. For example, it is used for quality control and for keeping up-to-date the documents used in the statistical teaching modules of the Biometry and Evolutionary Biology department at the University of Lyon, France: `http://pbil.univ-lyon1.fr/R/enseignement.html` (in French).

Using Sweave documents, however, is very demanding for users. LaTeX and R must be installed on the local computer, and users must have at least some notions of how to compile these documents. The version of R and the list of packages must be compatible with the R code included in the Sweave document. This can be a serious obstacle for many researchers, for example in Biology and Ecology.

The online reproducibility solution presented in this paper is much easier to use for people who do not know LaTeX and who have only vague notions of

R. It can be used with any web browser and it can make the analyses used in an applied statistics paper accessible to anyone. It can even be a good tool to help people get a better knowledge of R and encourage them to install R and use it locally.

Rweb servers

Many examples of the use of the PBIL Rweb server are available on the home page of J. R. Lobry, `http://pbil.univ-lyon1.fr/members/lobry/`. The articles under the tag "[ONLINE REPRODUCIBILITY]" are reproducible through a reproducibility web page, using HTML forms and Rweb as explained in this paper.

Writing such web pages is very easy, and does not require R, Rweb or any other particular software to be installed on the HTTP server. The HTML code below is a minimal example of such a page. It can be used with any web server using, for example, the apache2 HTTP server, or even locally in any web browser:

```
<html>
<head><title>Rweb</title></head>
<body>
<p style="font-size:30px">
Rweb server at PBIL</p>
<form onSubmit = "return checkData(this)"
action="http://pbil.univ-lyon1.fr/cgi-bin/
Rweb/Rweb.cgi"
enctype="multipart/form-data"
method="post">
<textarea name="Rcode" rows=5 cols=80>
plot(runif(100))
</textarea><br />
<input type="submit" value="Run it!">
</form>
</body>
</html>
```

The main feature of this code is the `form` tag that declares the CGI script of the Rweb server: `http://pbil.univ-lyon1.fr/cgi-bin/Rweb/Rweb.cgi`. The `textarea` tag in this form contains the R code: here by default it is just one plot function call that draws 100 random points. This default code can be changed in the HTML document, or modified interactively by the user when the page is loaded in a web browser. The `input` tag defines the "Run it!" button that can be used to execute the R code. When the user clicks on this button, the code is sent to the Rweb CGI script and executed by R. The browser display is updated, and the execution listing and the graph appear.

Another solution is to install Rweb and run a local (public or private) Rweb server.

Maintenance and durability

An important problem for reproducible research is durability. How long will a reproducible work stay reproducible?

The answer depends on the availability of the software needed to redo the computations and the graphics. R and contributed packages evolve, and R code can become deprecated if it is not updated on a regular basis. Sweave has been used mainly for just this reason in the documents used in the statistical teaching modules at the University of Lyon. All the documents are recompiled automatically to check their compatibility with the current version of R.

For an online reproducibility system to endure, the same conditions apply. The Biometry and Evolutionary Biology department has been running an Rweb server continuously since 2002, regularly updating the various pieces of software on which it depends (R, contributed packages, Rweb, Perl, Apache, etc.).

Reproducible research needs an ongoing effort; reproducibility ceases when this effort is not sustained.

Acknowledgment

We thank the Editor and two anonymous reviewers for many useful comments and suggestions that helped improve the first version of this paper.

Bibliography

J. Banfield. Rweb: web-based statistical analysis. *Journal of Statistical Software*, 4(1):1–15, 1999.

F. Baty, M. P. Bihl, G. Perrière, A. C. Culhane, and M. H. Brutsche. Optimized between-group classification: a new jackknife-based gene selection procedure for genome-wide expression data. *Bioinformatics*, 6:1–12, 2005.

J. P. Benzécri. Analyse de l'inertie intra-classe par l'analyse d'un tableau de correspondances. *Les Cahiers de l'Analyse des données*, 8:351–358, 1983.

J. B. Buckheit and D. L. Donoho. Wavelab and reproducible research. Tech. rep. 474, Dept. of Statistics, Stanford University, 1995. URL `http://www-stat.stanford.edu/~wavelab/Wavelab_850/wavelab.pdf`.

D. Charif and J. Lobry. SeqinR 1.0-2: a contributed package to the R project for statistical computing devoted to biological sequences retrieval and analysis. In H. R. U. Bastolla, M. Porto and M. Vendruscolo, editors, *Structural approaches to sequence evolution: Molecules, networks, populations*, Biological and Medical Physics, Biomedical Engineering, pages 207–232. Springer Verlag, New York, 2007. URL `http://seqinr.r-forge.r-project.org/`. ISBN : 978-3-540-35305-8.

C. Chauve. The poultry red mite *Dermanyssus gallinae* (DeGeer, 1778): current situation and future prospects for control. *Veterinary Parasitology*, 79: 239–245, 1998.

D. Chessel, A.-B. Dufour, and J. Thioulouse. The ade4 package-I- One-table methods. *R News*, 4:5–10, 2004.

A. C. Culhane, G. Perrière, E. C. Considine, T. G. Cotter and D. G. Higgins. Between-group analysis of microarray data. *Bioinformatics*, 18:1600–1608, 2002.

A. C. Culhane, J. Thioulouse, G. Perrière, and D. G. Higgins. MADE4: an R package for multivariate analysis of gene expression data. *Bioinformatics*, 21: 2789–2790, 2005.

J. de Leeuw. Reproducible research: the bottom line. Technical report, UCLA Statistics Program, 2001. URL http://preprints.stat.ucla.edu/301/301.pdf.

S. Dolédec and D. Chessel. Rythmes saisonniers et composantes stationnelles en milieu aquatique I- description d'un plan d'observations complet par projection de variables. *Acta Oecologica, Oecologia Generalis*, 8(3):403–426., 1987.

S. Dray and A.-B. Dufour. The ade4 package: Implementing the duality diagram for ecologists. *Journal of Statistical Software*, 22(4):1–20, 2007. URL http://www.jstatsoft.org/v22/i04.

Y. Escoufier. The duality diagramm : a means of better practical applications. In P. Legendre and L. Legendre, editors, *Development in numerical ecology*, NATO advanced Institute, Serie G, pages 139–156. Springer Verlag, Berlin,1987.

D. Firth. CGIwithR: Facilities for processing web forms using R. *Journal of Statistical Software*, 8(10): 1–8, 2003. URL http://www.jstatsoft.org/v08/i10.

R. Gentleman and D. T. Lang. Statistical analyses and reproducible research. *Bioconductor Project Working Papers.*, May 2004. URL http://www.bepress.com/bioconductor/paper2.

S. Holmes. Multivariate analysis: The French way. In D. Nolan and T. Speed, editors, *Festschrift for David Freedman*, pages 1–14. IMS, Beachwood, OH, 2006.

I. B. Jeffery, S. F. Madden, P. A. McGettigan, G. Perrière, A. C. Culhane and D. G. Higgins. Integrating transcription factor binding site information with gene expression datasets. *Bioinformatics*, 23:298–305, 2007.

A. Kirkwood. Anaemia in poultry infested with the red mite *Dermanyssus gallinae*. *The Veterinary Record*, 80(514-516), 1967.

D. Knuth. Literate Programming. Center for the Study of Language and Information, Stanford, California, 1992.

E. Lecoutre. The R2HTML Package. *R News*, 3(33-36), 2003.

F. Leisch. Sweave, Part I: Mixing R and LaTeX *R News*, 2(28-31), 2002.

K. S. Muyzer, G. and. Application of denaturing gradient gel electrophoresis (DGGE) and temperature gradient gel electrophoresis (TGGE) in microbial ecology. *Antonie van Leeuwenhoek*, 73:127–141, 1998.

G. Perrière, C. Combet, S. Penel, C. Blanchet, J. Thioulouse, C. Geourjon, J. Grassot, C. Charavay, M. Gouy, L. Duret, and G. Deléage Integrated databanks access and sequence/structure analysis services at the PBIL. *Nucleic Acids Research*, 31:3393–3399, 2003.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009. URL http://www.R-project.org. ISBN 3-900051-07-0.

D.K. Stewart, and W.A. Love. A general canonical correlation index. *Psychological Bulletin*, 70:160–163, 1968.

A. Suau, R. Bonnet, M. Sutren, J. Godon, G. R. Gibson, M. D. Collins, and J. Doré. Direct analysis of genes encoding 16s rRNA from complex communities reveals many novel molecular species within the human gut. *Appl. Env. Microbiol.*, 65:4799–4807, 1999.

H. Suzuky, C. J. Brown, L. J. Forney, and E. M. Top. Comparison of correspondence analysis methods for synonymous codon usage in bacteria. *DNA Research*, (in press), 2008. doi: 10.1093/dnares/dsn028.

C. J. F. ter Braak and P. Šmilauer. *CANOCO Reference Manual and CanoDraw for Windows User's Guide: Software for Canonical Community Ordination (version 4.5).* Microcomputer Power, Ithaca NY, USA, 2002.

J. Thioulouse, D. Chessel, S. Doledec, and J.M. Olivier. ADE-4: a multivariate analysis and graphical display software. *Statistics and Computing*, 7:75–82, 1997.

J. Thioulouse and S. Dray. Interactive multivariate data analysis in R with the **ade4** and **ade4TkGUI** packages. *Journal of Statistical Software*, 22(5):1–14, 10 2007. URL http://www.jstatsoft.org/v22/i05.

C. Valiente-Moro, C. Chauve, and L. Zenner. Vectorial role of some dermanyssoid mites (Acari, Mesostigmata, Dermanyssoidea). *Parasite*, 12:99–109, 2005.

C. Valiente-Moro, C. Chauve, and L. Zenner. Experimental infection of *Salmonella Enteritidis* by the poultry red mite, *Dermanyssus gallinae*. *Veterinary Parasitology*, 31:329–336, 2007.

C. Valiente-Moro, J. Thioulouse, C. Chauve, P. Normand, and L. Zenner. Bacterial taxa associated with the hematophagous mite, *Dermanyssus gallinae* detected by 16s rDNA PCR amplification and TTGE fingerprint. *Research in Microbiology*, 160:63–70, 2009.

Jean Thioulouse
Université de Lyon, F-69000, Lyon ; Université Lyon 1;
CNRS, UMR5558, Biométrie et Biologie Evolutive,
F-69622 Villeurbanne Cedex, France.
jean.thioulouse@univ-lyon1.fr
http://pbil.univ-lyon1.fr/JTHome/

Claire Valiente-Moro
Ecole Nationale Vétérinaire de Lyon,
1 avenue Bourgelat, 69280 Marcy l'Etoile, France.
Université de Lyon, F-69000, Lyon ; Université Lyon 1;
CNRS, UMR5557, Écologie Microbienne,
F-69622 Villeurbanne Cedex, France.
claire.valiente-moro@univ-lyon1.fr.

Lionel Zenner
Ecole Nationale Vétérinaire de Lyon,
1 avenue Bourgelat, 69280 Marcy l'Etoile, France.
Université de Lyon, F-69000, Lyon ; Université Lyon 1;
CNRS, UMR5558, Biométrie et Biologie Evolutive,
F-69622 Villeurbanne Cedex, France.
l.zenner@vet-lyon.fr

Two-sided Exact Tests and Matching Confidence Intervals for Discrete Data

by Michael P. Fay

Abstract There is an inherent relationship between two-sided hypothesis tests and confidence intervals. A series of two-sided hypothesis tests may be inverted to obtain the *matching* 100(1-α)% confidence interval defined as the smallest interval that contains all point null parameter values that would not be rejected at the α level. Unfortunately, for discrete data there are several different ways of defining two-sided exact tests and the most commonly used two-sided exact tests are defined one way, while the most commonly used exact confidence intervals are inversions of tests defined another way. This can lead to inconsistencies where the exact test rejects but the exact confidence interval contains the null parameter value. The packages **exactci** and **exact2x2** provide several exact tests with the matching confidence intervals avoiding these inconsistencies as much as possible. Examples are given for binomial and Poisson parameters and both paired and unpaired 2×2 tables.

Applied statisticians are increasingly being encouraged to report confidence intervals (CI) and parameter estimates along with p-values from hypothesis tests. The `htest` class of the **stats** package is ideally suited to these kinds of analyses, because all the related statistics may be presented when the results are printed. For exact two-sided tests applied to discrete data, a test-CI inconsistency may occur: the p-value may indicate a significant result at level α while the associated 100(1-α)% confidence interval may cover the null value of the parameter. Ideally, we would like to present a unified report (Hirji, 2006), whereby the p-value and the confidence interval match as much as possible.

A motivating example

I was asked to help design a study to determine if adding a new drug (*albendazole*) to an existing treatment regimen (*ivermectin*) for the treatment of a parasitic disease (*lymphatic filariasis*) would increase the incidence of a rare serious adverse event when given in an area endemic for another parasitic disease (*loa loa*). There are many statistical issues related to that design (Fay et al., 2007), but here consider a simple scenario to highlight the point of this paper. A previous mass treatment using the existing treatment had 2 out of 17877 experiencing the serious adverse event (SAE) giving an observed rate of 11.2 per 100,000. Suppose the new treatment was given to 20,000 new subjects and suppose that 10 subjects experienced the SAE giving an observed rate of 50 per 100,000. Assuming Poisson rates, an exact test using `poisson.test(c(2,10),c(17877,20000))` from the **stats** package (throughout we assume version 2.11.0 for the stats package) gives a p-value of $p = 0.0421$ implying significant differences between the rates at the 0.05 level, but `poisson.test` also gives a 95% confidence interval of $(0.024, 1.050)$ which contains a rate ratio of 1, implying no significant difference. We return to the motivating example in the 'Poisson two-sample' section later.

Overview of two-sided exact tests

We briefly review inferences using the p-value function for discrete data. For details see Hirji (2006) or Blaker (2000). Suppose you have a discrete statistic t with random variable T such that larger values of T imply larger values of a parameter of interest, θ. Let $F_\theta(t) = Pr[T \le t; \theta]$ and $\bar{F}_\theta(t) = Pr[T \ge t; \theta]$. Suppose we are testing

$$H_0 : \theta \ge \theta_0$$
$$H_1 : \theta < \theta_0$$

where θ_0 is known. Then smaller values of t are more likely to reject and if we observe t, then the probability of observing equal or smaller values is $F_{\theta_0}(t)$ which is the one-sided p-value. Conversely, the one-sided p-value for testing $H_0 : \theta \le \theta_0$ is $\bar{F}_{\theta_0}(t)$. We reject when the p-value is less than or equal to the significance level, α. The one-sided confidence interval would be all values of θ_0 for which the p-value is greater than α.

We list 3 ways to define the two-sided p-value for testing $H_0 : \theta = \theta_0$, which we denote p_c, p_m and p_b for the `central`, `minlike`, and `blaker` methods, respectively:

central: p_c is 2 times the minimum of the one-sided p-values bounded above by 1, or mathematically,

$$p_c = \min\left\{1, 2 \times \min\left(F_{\theta_0}(t), \bar{F}_{\theta_0}(t)\right)\right\}.$$

The name `central` is motivated by the associated inversion confidence intervals which are central intervals, i.e., they guarantee that the lower (upper) limit of the 100(1-α)% confidence interval has less than $\alpha/2$ probability of being greater (less) than the true parameter. This is called the TST (twice the smaller tail method) by Hirji (2006).

minlike: p_m is the sum of probabilities of outcomes with likelihoods less than or equal to the observed likelihood, or

$$p_m = \sum_{T:f(T) \leq f(t)} f(T)$$

where $f(t) = Pr[T = t; \theta_0]$. This is called the PB (probability based) method by Hirji (2006).

blaker: p_b combines the probability of the smaller observed tail with the smallest probability of the opposite tail that does not exceed that observed tail probability. Blaker (2000) showed that this p-value may be expressed as

$$p_b = Pr[\gamma(T) \leq \gamma(t)]$$

where $\gamma(T) = \min\{F_{\theta_0}(T), \bar{F}_{\theta_0}(T)\}$. The name `blaker` is motivated by Blaker (2000) which comprehensively studies the associated method for confidence intervals, although the method had been mentioned in the literature earlier, see e.g., Cox and Hinkley (1974), p. 79. This is called the CT (combined tail) method by Hirji (2006).

There are other ways to define two-sided p-values, such as defining extreme values according to the score statistic (see e.g., Hirji (2006, Chapter 3), or Agresti and Min (2001)). Note that $p_c \geq p_b$ for all cases, so that p_b gives more powerful tests than p_c. On the other hand, although generally $p_c > p_m$ it is possible for $p_c < p_m$.

If $p(\theta_0)$ is a two-sided p-value testing $H_0 : \theta = \theta_0$, then its $100(1-\alpha)$% matching confidence interval is the smallest interval that contains all θ_0 such that $p(\theta_0) > \alpha$. To calculate the matching confidence intervals, we consider only regular cases where $F_\theta(t)$ and $\bar{F}_\theta(t)$ are monotonic functions of θ (except perhaps the degenerate cases where $F_\theta(t) = 1$ or $\bar{F}_\theta(t) = 0$ for all θ when t is the maximum or minimum). In this case the matching confidence limits to the `central` test are (θ_L, θ_U) which are solutions to:

$$\alpha/2 = \bar{F}_{\theta_L}(t)$$

and

$$\alpha/2 = F_{\theta_U}(t)$$

except when t is the minimum or maximum, in which case the limit is set at the appropriate extreme of the parameter space. The matching confidence intervals for p_m and p_b require a more complicated algorithm to ensure precision of the confidence limits (Fay, 2010).

If matching confidence intervals are used then test-CI inconsistencies will not happen for the `central` method, and will happen very rarely for the `minlike` and `blaker` methods. We discuss those rare test-CI inconsistencies in the 'Unavoidable inconsistencies' section later, but the main point of this article is that it is not rare for p_m to be inconsistent with the `central` confidence interval (Fay, 2010) and that particular test-CI combination is the default for many exact tests in the **stats** package. We show some examples of such inconsistencies in the following sections.

Binomial: one-sample

If X is binomial with parameters n and θ, then the `central` exact interval is the Clopper-Pearson confidence interval. These are the intervals given by `binom.test`. The p-value given by `binom.test` is p_m. The matching interval to the p_m was proposed by Stern (1954) (see Blaker (2000)).

When $\theta_0 = 0.5$ we have $p_c = p_m = p_b$, and there is no chance of a test-CI inconsistency even when the confidence intervals are not inversions of the test as is the case in `binom.test`. When $\theta_0 \neq 0.5$ there may be problems. We explore these cases in the 'Poisson: two-sample' section later, since the associated tests reduce through conditioning to one-sample binomial tests.

Note that there is a theoretically proven set of shortest confidence intervals for this problem. These are called the Blyth-Still-Casella intervals in StatXact (StatXact Procs Version 8). The problem with these shortest intervals is that they are not nested, so that one could have a parameter value that is included in the 90% confidence interval but not in the 95% confidence interval (see Theorem 2 of Blaker (2000)). In contrast, the matching intervals of the `binom.exact` function of the **exactci** will always give nested intervals.

Poisson: one-sample

If X is Poisson with mean θ, then `poisson.test` from **stats** gives the exact `central` confidence intervals (Garwood, 1936), while the p-value is p_m. Thus, we can easily find a test-CI inconsistency: `poisson.test(5, r=1.8)` gives a p-value of $p_m = 0.036$ but the 95% central confidence interval of $(1.6, 11.7)$ contains the null rate of 1.8. As θ gets large the Poisson distribution may be approximated by the normal distribution and these test-CI inconsistencies become more rare.

The **exactci** package contains the `poisson.exact` function, which has options for each of the three methods of defining p-values and gives matching confidence intervals. The code `poisson.exact(5, r=1.8, tsmethod="central")` gives the same confidence interval as above, but a p-value of $p_c = 0.073$; while `poisson.exact(5, r=1.8, tsmethod="minlike")` returns a p-value equal to p_m, but a 95% confidence interval of $(2.0, 11.8)$. Finally, using `tsmethod="blaker"` we get $p_b = 0.036$ (it is not uncommon for p_b to equal p_m) and a 95% confidence interval of $(2.0, 11.5)$. We see that there is no test-CI inconsistency when using the matching confidence intervals.

Poisson: two-sample

For the control group, let the random variable of the counts be Y_0, the rate be λ_0 and the population at risk be m_0. Let the corresponding values for the test group be Y_1, λ_1 and m_1. If we condition on $Y_0 + Y_1 = N$ then the distribution of Y_1 is binomial with parameters N and

$$\theta = \frac{m_1\lambda_1}{m_0\lambda_0 + m_1\lambda_1}$$

This parameter may be written in terms of the ratio of rates, $\rho = \lambda_1/\lambda_2$ as

$$\theta = \frac{m_1\rho}{m_0 + m_1\rho}$$

or equivalently,

$$\rho = \frac{m_0\theta}{m_1(1-\theta)}. \qquad (1)$$

Thus, the null hypothesis that $\lambda_1 = \lambda_0$ is equivalent to $\rho = 1$ or $\theta = m_1/(m_0 + m_1)$, and confidence intervals for θ may be transformed into confidence intervals for ρ by Equation 1. So the inner workings of the `poisson.exact` function when dealing with two-sample tests simply use the `binom.exact` function and transform the results using Equation 1.

Let us return to our motivating example (i.e., testing the difference between observed rates 2/17877 and 10/20000). As in the other sections, the results from `poisson.test` output p_m but the 95% central confidence intervals, as we have seen, give a test-CI inconsistency. The `poisson.exact` function avoids such test-CI inconsistency in this case by giving the matching confidence interval; here are the results of the three `tsmethod` options:

tsmethod	p-value	95% confidence interval
central	0.061	(0.024, 1.050)
minlike	0.042	(0.035, 0.942)
blaker	0.042	(0.035, 0.936)

Analysis of 2×2 tables, unpaired

The 2×2 table may be created from many different designs, consider first designs where there are two groups of observations with binary outcomes. If all the observations are independent, even if the number in each group is not fixed in advance, proper inferences may still be obtained by conditioning on those totals (Lehmann and Romano, 2005). Fay (2010) considers the 2×2 table with independent observations, so we only briefly present his motivating example here. The usual two-sided application of Fisher's exact test: `fisher.test(matrix(c(4,11,50,569), 2, 2))` gives $p_m = 0.032$ using the `minlike` method, but 95% confidence interval on the odds ratio of $(0.92, 14.58)$ using the `central` method. As with the other examples, the test-CI inconsistency disappears when we use either the `exact2x2` or `fisher.exact` function from the **exact2x2** package.

Analysis of 2×2 tables, paired

The case not studied in Fay (2010) is when the data are paired, the case which motivates McNemar's test. For example, suppose you have twins randomized to two treatment groups (Test and Control) then test on a binary outcome (pass or fail). There are 4 possible outcomes for each pair: (a) both twins fail, (b) the twin in the control group fails and the one in the test group passes, (c) the twin on the test group fails and the one in the control group passes, or (d) both twins pass. Here is a table where the numbers of sets of twins falling in each of the four categories are denoted a, b, c and d:

	Test	
Control	Fail	Pass
Fail	a	b
Pass	c	d

In order to test if the treatment is helpful, we use only the numbers of discordant pairs of twins, b and c, since the other pairs of twins tell us nothing about whether the treatment is helpful or not. McNemar's test statistic is

$$Q \equiv Q(b,c) = \frac{(b-c)^2}{b+c}$$

which for large samples is distributed like a chi-squared distribution with 1 degree of freedom. A closer approximation to the chi-squared distribution uses a continuity correction:

$$Q_C \equiv Q_C(b,c) = \frac{(|b-c|-1)^2}{b+c}$$

In R this test is given by the function `mcnemar.test`.

Case-control data may be analyzed this way as well. Suppose you have a set of people with some rare disease (e.g., a certain type of cancer); these are called the cases. For this design you match each case with a control who is as similar as feasible on all important covariates except the exposure of interest. Here is a table:

	Exposed	
Not Exposed	Control	Case
Control	a	b
Case	c	d

For this case as well we can use Q or Q_C to test for no association between case/control status and exposure status.

For either design, we can estimate the odds ratio by b/c, which is the maximum likelihood estimate (see Breslow and Day (1980), p. 165). Consider some hypothetical data (chosen to highlight some points):

	Test	
Control	Fail	Pass
Fail	21	9
Pass	2	12

When we perform McNemar's test with the continuity correction we get $p = 0.070$ while without the correction we get $p = 0.035$. Since the inferences are on either side of the traditional 0.05 cutoff of significance, it would be nice to have an exact version of the test to be clearer about significance at the 0.05 level. From the **exact2x2** package using `mcnemar.exact` we get the exact McNemar's test p-value of $p = .065$. We now give the motivation for the exact version of the test.

After conditioning on the total number of discordant pairs, $b + c$, we can treat the problem as $B \sim Binomial(b + c, \theta)$, where B is the random variable associated with b. Under the null hypothesis $\theta = 0.5$. We can transform the parameter θ into an odds ratio by

$$\text{Odds Ratio} \equiv \phi = \frac{\theta}{1-\theta} \qquad (2)$$

(Breslow and Day (1980), p. 166). Since it is easy to perform exact tests on a binomial parameter, we can perform exact versions of McNemar's test internally using the `binom.exact` function of the package **exactci** then transform the results into odds ratios via Equation 2. This is how the calculations are done in the `exact2x2` function when `paired=TRUE`. The `alternative` and the `tsmethod` options work in the way one would expect. So although McNemar's test was developed as a two-sided test testing the null that $\theta = 0.5$ (or equivalently $\phi = 1$), we can easily extend this to get one-sided exact McNemar-type Tests. For two-sided tests we can get three different versions of the two-sided exact McNemar's p-value function using the three `tsmethod` options, but all three are equivalent to the exact version of McNemar's test when testing the usual null that $\theta = 0.5$ (see the Appendix in `vignette("exactMcNemar")` in **exact2x2**). If we narrowly define McNemar's test as only testing the null that $\theta = 0.5$ as was done in the original formulation, there is only one exact McNemar's test; it is only when we generalize the test to test null hypotheses of $\theta = \theta_0 \neq 0.5$ that there are differences between the three methods. Those differences between the `tsmethod` options become apparent in the calculation of the confidence intervals. The default is to use `central` confidence intervals so that they guarantee that the lower (upper) limit of the 100(1-α)% confidence interval has less than $\alpha/2$ probability of being greater (less) than the true parameter. These guarantees on each tail are not true for the `minlike` and `blaker` two-sided confidence intervals; however, the latter give generally tighter confidence intervals.

Graphing P-values

In order to gain insight as to why test-CI inconsistencies occur, we can plot the p-value function. This type of plot explores one data realization and its many associated p-values on the vertical axis representing a series of tests modified by changing the point null hypothesis parameter (θ_0) on the horizontal axis. There is a default plot command for `binom.exact`, `poisson.exact`, and `exact2x2` that plots the p-value as a function of the point null hypotheses, draws vertical lines at the confidence limits, draws a line at 1 minus the confidence level, and adds a point at the null hypothesis of interest. Other plot functions (`exactbinomPlot`, `exactpoissonPlot`, and `exact2x2Plot`) can be used to add to that plot for comparing different methods. In Figure 1 we create such a plot for the motivating example. Here is the code to create that figure:

```
x <- c(2, 10)
n <- c(17877, 20000)
poisson.exact(x, n, plot = TRUE)
exactpoissonPlot(x, n, tsmethod = "minlike",
    doci = TRUE, col = "black", cex = 0.25,
    newplot = FALSE)
```

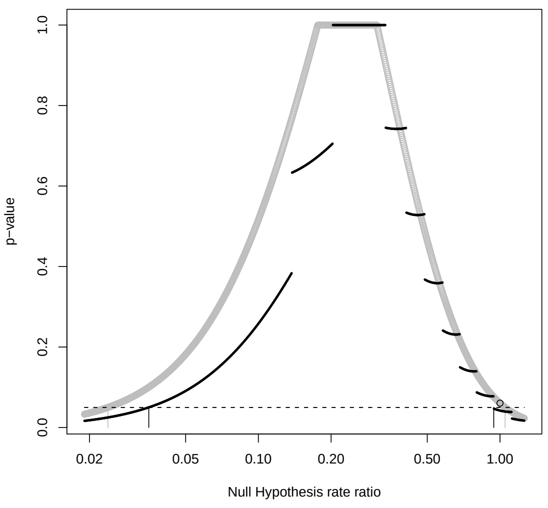


Figure 1: Graph of p-value functions for motivating two-sample Poisson example. Gray is `central` method, black is `minlike` method. Vertical lines are 95% confidence limits, black circle is `central` p-value at null rate ratio of 1.

We see from Figure 1 that the `central` method has smoothly changing p-values, while the `minlike` method has discontinuous ones. The usual confidence interval is the inversion of the `central` method (the limits are the vertical gray lines, where the dotted line at the significance level intersects with the gray p-values), while the usual p-value at the null that the rate ratio is 1 is where the black line is. To see this more clearly we plot the lower right hand corner of Figure 1 in Figure 2.

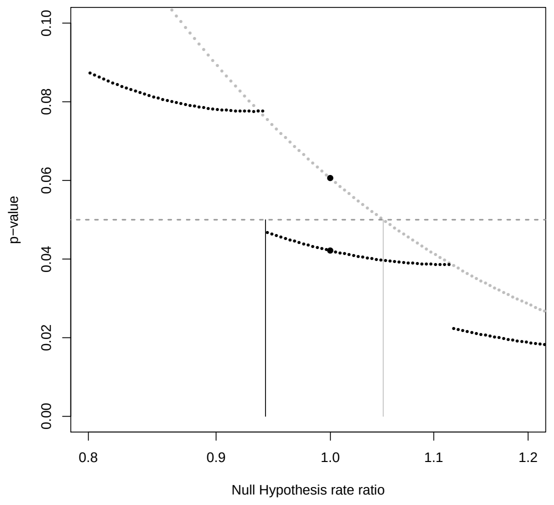


Figure 2: Graph of p-value functions for motivating two-sample Poisson example. Gray is `central` method, black is `minlike` method. Vertical lines are upper 95% confidence limits, solid black circles are the respective p-values at null rate ratio of 1.

From Figure 2 we see why the test-CI inconsistencies occur, the `minlike` method is generally more powerful than the `central` method, so that is why the p-values from the `minlike` method can reject a specific null when the confidence intervals from the `central` method imply failing to reject that same null. We see that in general if you use the matching confidence interval to the p-value, there will not be test-CI inconsistencies.

Unavoidable Inconsistencies

Although the **exactci** and **exact2x2** packages do provide a unified report in the sense described in Hirji (2006), it is still possible in rare instances to obtain test-CI inconsistencies when using the `minlike` or `blaker` two-sided methods (Fay, 2010). These rare inconsistencies are unavoidable due to the nature of the problem rather than any deficit in the packages.

To show the rare inconsistency problem using the motivating example, we consider the unrealistic situation where we are testing the null hypothesis that the rate ratio is 0.93 at the 0.0776 level. The corresponding confidence interval would be a 92.24% $= 100 \times (1 - 0.0776)$ interval. Using `poisson.exact(x, n, r=.93, tsmethod="minlike", conf.level=1-0.0776)` we reject the null (since $p_m = 0.07758 < 0.0776$) but the 92.24% matching confidence interval contains the null rate ratio of 0.93. In this situation, the confidence set that is the inversion of the series of tests is two disjoint intervals ($[0.0454, 0.9257]$ and $[0.9375, 0.9419]$), and the matching confidence interval fills in the hole in the confidence set. This is an unavoidable test-CI inconsistency. Figure 3 plots the situation.

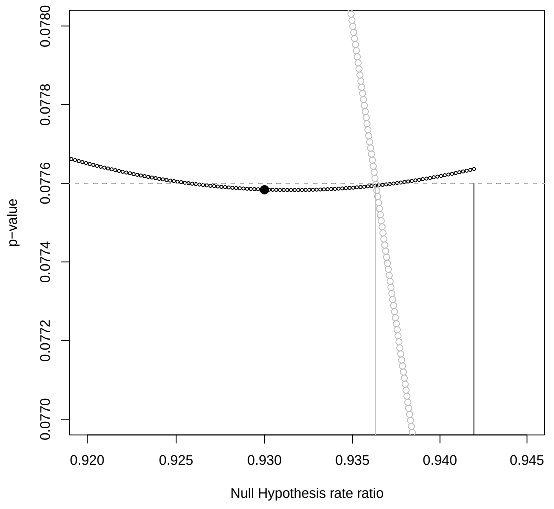


Figure 3: Graph of p-value functions for motivating two-sample Poisson example. Gray is `central` method, black is `minlike` method. Circles are p-values, vertical lines are the upper 92.24% confidence limits, solid black circle is `minlike` p-value at null rate ratio of 0.93. The unrealistic significance level of 0.0776 and null rate ratio of 0.93 are chosen to highlight the rare unavoidable inconsistency problem.

Additionally, the options `tsmethod="minlike"` or `"blaker"` can have other anomalies (see Vos and Hudson (2008) for the single sample binomial case, and Fay (2010) for the two-sample binomial case). For example, the data reject, but fail to reject if an additional observation is added *regardless* of the value of the additional observation. Thus, although the power of the `blaker` (or `minlike`) two-sided method is always (almost always) greater than the `central` two-sided method, the `central` method does avoid all test-CI inconsistencies and the previously mentioned anomalies.

Discussion

We have argued for using a unified report whereby the p-value and the confidence interval are calculated from the same p-value function (also called the evidence function or confidence curve). We have provided several practical examples. Although the theory of these methods have been extensively studied (Hirji, 2006), software has not been readily available. The **exactci** and **exact2x2** packages fill this need. We know of no other software that provides the `minlike` and `blaker` confidence intervals, except the **PropCIs** package which provides the Blaker confidence interval for the single binomial case only.

Finally, we briefly consider closely related software. The **rateratio.test** package does the two-sample Poisson exact test with confidence intervals using the `central` method. The **PropCIs** package does several different asymptotic confidence intervals, as well as the Clopper-Pearson (i.e. `central`) and Blaker exact intervals for a single binomial proportion. The **PropCIs** package also performs the mid-p adjustment to the Clopper-Pearson confidence interval which is not currently available in **exactci**. Other exact confidence intervals are not covered in the current version of **PropCIs** (Version 0.1-6). The **coin** and **perm** packages give very general methods for performing exact permutation tests, although neither perform the exact matching confidence intervals for the cases studied in this paper.

I did not perform a comprehensive search of commercial statistical software; however, SAS (Version 9.2) (perhaps the most comprehensive commercial statistical software) and StatXact (Version 8) (the most comprehensive software for exact tests) both do not implement the `blaker` and `minlike` confidence intervals for binomial, Poisson and 2x2 table cases.

Bibliography

A. Agresti and Y. Min. On small-sample confidence intervals for parameters in discrete distributions. *Biometrics*, 57:963–971, 2001.

H. Blaker. Confidence curves and improved exact confidence intervals for discrete distributions. *Canadian Journal of Statistics*, 28:783–798, 2000.

N. Breslow and N. Day. *Statistical Methods in Cancer Research: Volume 1: Analysis of Case Control Studies*. International Agency for Research in Cancer, Lyon, France, 1980.

D. Cox and D. Hinkley. *Theoretical Statistics*. Chapman and Hall, London, 1974.

M. Fay. Confidence intervals that match Fisher's exact or Blaker's exact tests. *Biostatistics*, 11:373–374, 2010.

M. Fay, C. Huang, and N. Twum-Danso. Monitoring rare serious adverse events from a new treatment and testing for a difference from historical controls. *Controlled Clinical Trials*, 4:598–610, 2007.

F. Garwood. Fiducial limits for the Poisson distribution. *Biometrika*, pages 437–442, 1936.

K. Hirji. *Exact Analysis of Discrete Data*. Chapman and Hall/CRC, New York, 2006.

E. Lehmann and J. Romano. *Testing Statistical Hypotheses, third edition*. Springer, New York, 2005.

T. Stern. Some remarks on confidence and fiducial limits. *Biometrika*, pages 275–278, 1954.

P. Vos and S. Hudson. Problems with binomial two-sided tests and the associated confidence intervals. *Australian and New Zealand Journal of Statistics*, 50: 81–89, 2008.

Michael P. Fay
National Institute of Allergy and Infectious Diseases
6700-A Rockledge Dr. Room 5133, Bethesda, MD 20817
USA
`mfay@niaid.nih.gov`

Book Reviews

A Beginner's Guide to R

Alain F. Zuur, Elena N. Ieno, and Erik H.W.G. Meesters. New York, NY: Springer, 2009. ISBN 978-0-387-93836-3. xv + 218 pp. $59.95 (P).

A Beginner's Guide to R is just what it's title implies, a quick-start guide for the newest R users. A unique feature of this welcome addition to Springer's *Use R!* series is that it is devoted solely to getting the user up and running on R. Unlike other texts geared towards R beginners, such as Verzani (2005), this text does not make the mistake of trying to simultaneously teach statistics. The experienced R user will not have much use for this book, except perhaps for adoption as a textbook. To this end, there are straightforward homework exercises provided throughout (no answers, though), and the data sets can be downloaded from the authors' website `http://www.highstat.com`. It should be noted, however, that the examples and exercises are limited to the authors' area of expertise – ecology.

The book starts at the very beginning by instructing the user how to install R and load packages from CRAN. One small weakness is that the book is directed almost exclusively toward PC users. In particular, I was disappointed by the paucity of information concerning R text editors that are compatible with the Mac. (After a fair amount of trial and error, I finally determined that `gedit` would do the job for me.) A nice feature of Chapter 1 is an annotated bibliography of "must-have" R books. Early on, the authors sagely direct the reader toward the universe of R assistance available online (and console the panicked reader that even experienced R users can be intimidated by the sheer amount of information contained in R help files).

The remainder of the book is devoted to demonstrating how to do the most basic tasks with R. Chapter 2 describes several methods for getting data into R (useful information for anybody facing the daunting prospect of importing a large data set into R for the first time). To appease the novice hungering for some "fancy" R output, the authors provide easy-to-follow instructions for constructing both simple (Chapters 5 and 7) and not-so-simple (Chapter 8) graphical displays. Standard plots from the introductory statistics curriculum are included (e.g., the histogram, boxplot, scatterplot, and dotplot), and the `lattice` package is introduced for the benefit of readers with more advanced graphical needs. Other topics include data management (Chapter 3) and simple functions and loops (Chapters 4 and 6). Chapter 9 concludes the book by suggesting solutions to some problems commonly encountered by R users, beginners and old pros alike.

In sum, *A Beginner's Guide to R* is an essential resource for the R novice, whether an undergraduate learning statistics for the first time or a seasoned statistician biting the bullet and making the switch to R. To get the most bang for the buck (the cost is a bit steep for such a short paperback), I advise the user to set aside a weekend (or a week) to launch R and work through the book from start to finish. It will be time well spent — just keep in mind that this book is all about learning to play a scale; you'll be disappointed if you expect to emerge with all the skills required to perform a concerto.

Laura M. Schultz
Rowan University

Bibliography

J. Verzani. *Using R for Introductory Statistics*. Boca Raton, FL: Chapman & Hall, 2005.

Conference Review: The 2nd Chinese R Conference

by Jing Jiao, Jingjing Guan and Yihui Xie

The 2nd Chinese R Conference was successfully held in December 2009 in Beijing and Shanghai. This conference was organized in two cities so that more people would be able to enjoy similar talks without additional traveling cost. The Beijing session, held in the Renmin University of China (RUC) on December 5–6, 2009, was sponsored and organized by the Center for Applied Statistics and the School of Statistics, RUC. Yanping Chen served as the chair of the conference while Janning Fan was the secretary. Yixuan Qiu and Jingjing Guan were in charge of local arrangements. The Shanghai session took place in the East China Normal University (ECNU) on December 12–13, 2009; it was organized by the School of Resources and Environment Science (SRES) and the School of Finance and Statistics, ECNU, and sponsored by Mango Solutions. Jing Jiao and Xiang Zhang served as the chairs of the conference; the Molecular Ecology Group in SRES was in charge of local arrangements. Both sessions were co-organized by the web community "Capital of Statistics" (`http://cos.name`).

Beside promoting the communication among Chinese R users as the 1st Chinese R Conference did, the 2nd Chinese R Conference concentrated on the applications of R in statistical computing and graphics in various disciplines. The slogan of this conference was "useR eveRywheRe", indicating R is widespread in a variety of fields as well as there are a large amount of useRs in China now.

In total, more than 300 participants from over 90 institutions took part in this conference. There were 19 talks in the Beijing session:

Opening A review on the history of the Chinese R Conference and a brief introduction to R by Yanping Chen;

Statistical Graphics Introduction to R base graphics by Tao Gao and Cheng Li; Matrix visualization using the package **corrplot** by Taiyun Wei;

Statistical Models Nonparametric methods and robust estimation of quantile regression by Chen Zuo; R and WinBUGS by Peng Ding; Grey System Theory in R by Tan Xi;

Software Development Integrating R into C/C++ applications using Visual C++ by Yu Gong; Using R in online data analysis by Zhiyi Huang;

Industrial Applications R in establishing standards for the food industry by Qiding Zhong; Investigation and monitoring on geological environment with R by Yongsheng Liu; R in marketing research by Yingchun Zhu; Zhiyi Huang also gave an example on handling atmosphere data with R; R in near-infrared spectrum analysis by Die Sun;

Data Mining Jingjing Guan introduced **RExcel** with applications in data mining and the trend of data mining focused on ensemble learning; Dealing with large data and reporting automation by Sizhe Liu;

Kaleidoscope Discussing security issues of R by Nan Xiao; Using R in economics and econometrics by Liyun Chen; R in psychology by Xiaoyan Sun and Ting Wang; R in spatial analysis by Huaru Wang; Optimization of molecular structure parameter matrix in QSAR with the package **omd** by Bin Ma;

More than 150 people from nearly 75 institutions all over China participated in the Shanghai session. Among these participants, we were honored to meet some pioneer Chinese useRs such as Prof Yincai Tang, the first teacher introducing R in the School of Finance and Statistics of ECNU. There were 13 talks given in the Shanghai session which partially overlapped with the Beijing session:

Opening Introduction of the 2nd Chinese R conference (Shanghai session) by Jing Jiao, and opening address by Prof Yincai Tang and Prof Xiaoyong Chen (Vice President of SRES);

Theories Bayesian Statistics with R and WinBUGS by Prof Yincai Tang; Grey System Theory in R by Tan Xi;

Graphics Introduction to R base graphics by Tao Gao and Cheng Li; Matrix visualization using the package **corrplot** by Taiyun Wei;

Applications Using R in economics and econometrics by Liyun Chen; Marketing analytical framework by Zhenshun Lin; Decision tree with the **rpart** package by Weijie Wang; Optimization of molecular structure parameter matrix in QSAR with the package **omd** by Bin Ma; Survival analysis in R by Yi Yu; The application of R and statistics in the semiconductor industry by Guangqi Lin;

Software Development Dealing with large data and reporting automation by Sizhe Liu; JAVA development and optimization in R by Jian Li; Discussing security issues of R by Nan Xiao;

On the second day there was a training session for R basics (Learning R in 153 minutes by Sizhe Liu) and extensions (Writing R packages by Sizhe Liu and Yihui Xie; Extending R by Jian Li); a discussion session was arranged after the training. The conference was closed with a speech by Xiang Zhang.

Overall, participants were very interested in the topics contributed on the conference and the discussion session has reflected there was the strong need of learning and using R in China. After this conference, we have also decided to make future efforts on:

- Increasing the number of sessions of this conference to meet the demand of useRs in other areas;
- Promoting the communication between statistics and other disciplines via using R;
- Interacting with different industries through successful applications of R;

All slides are available online at `http://cos.name/2009/12/2nd-chinese-r-conference-summary/`. We thank Prof Xizhi Wu for his consistent encouragement on this conference, and we are especially grateful to Prof Yanyun Zhao, Dean of the School of Statistics of RUC, for his great support on the Beijing session. We look forward to the next R conference in China and warmly welcome more people to attend it. The main session is tentatively arranged in the Summer of 2010. Inquiries and suggestions can be sent to `ChinaR-2010@cos.name` or `http://cos.name/ChinaR/ChinaR-2010`.

Jing Jiao
School of Resources and Environment Science
East China Normal University
China P. R.
`jing.jiao@cos.name`

Jingjing Guan
School of Statistics, Renmin University of China
China P. R.
`jingjing.guan@cos.name`

Yihui Xie
Department of Statistics & Statistical Laboratory
Iowa State University
USA
`yihui.xie@cos.name`

Introducing NppToR: R Interaction for Notepad++

by Andrew Redd

Introduction

A good practice when programming or doing analysis in R is to keep the commands in a script that can be saved and reused. An appropriate editor makes programming in R much more efficient and less error prone.

On Linux/Unix systems, users typically use either Vim or EMACS with ESS. Both of these have features to improve programming in R. Features like syntax highlighting, which means certain keywords are colored or marked to indicate they have special meaning, and code folding, where lines of code can be automatically grouped and temporarily hidden from view. Perhaps the most important feature is the ability to directly evaluate lines of code in the R interpreter. This facilitates quick and easy programming while maintaining a solid record in a reusable script.

The version of R for Microsoft Windows comes with a built in editor. This editor does have the interaction features like VIM and EMACS, but lacks the other features like syntax highlighting and code folding. There are several alternative code editors for Windows, such as Notepad++, WinEDT, Tinn-R, and Eclipse, that do possess advanced features but many novice users are simply unaware of them.

NppToR is a native windows solution that brings the advantages that users of other platforms have to Windows users. NppToR works with the powerful and popular code editor Notepad++ (`http://notepad-plus.sourceforge.net/`). Notepad++ has a natural windows feel to it and so the learning curve for getting started is almost nonexistent, but as users become more accustomed to the advanced features of Notepad++ the real power comes out.

NppToR tries to follow the easy use and low learning curve of Notepad++. It not only simply works, but also becomes more powerful as users become more familiar with it's features. NppToR has been under development for over a year and has gained a serious user base without ever being promoted outside of the R mailing lists. NppToR can be downloaded from SourceForge (`http://sourceforge.net/projects/npptor/`).

Interaction

The primary purpose of NppToR is to add R interaction capability to Notepad++. It does this through the use of hotkeys. When the hotkey is pressed it triggers an action such as evaluating a line of code. Figure 1 lists the hotkeys and what action they perform. The hotkeys are totally configurable through the settings.

NppToR Default Hotkeys and Actions

`F8`	Evaluate a line of code or selection.
`Ctrl+F8`	Evaluate the entire current file.
`Shift+F8`	Evaluate the file to the point of the cursor.
`Ctrl+Alt+F8`	Evaluate as a batch file and examine the results.

Figure 1: The default hotkeys for evaluating code with NppToR.

NppToR sits in in the system tray, as shown in Figure 2. The menu, where the settings and extra features are found, is accessed by right clicking on the NppToR icon in the system tray.

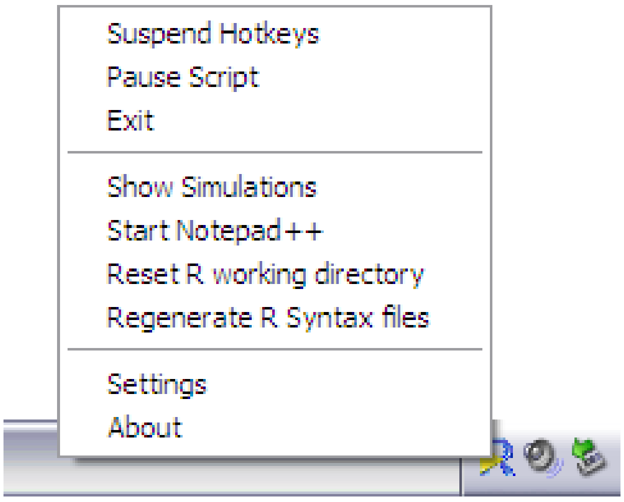


Figure 2: NppToR sits as a utility in the system tray.

One of the ways that NppToR just works is that when evaluating code from Notepad++, NppToR looks for a current open R GUI session in which to evaluate the code. If R is not running, a new R session will be started and the code evaluated there. The R home directory is automatically found through the windows registry. One advantage of having NppToR starting R is that the working directory for R is set to the directory of the open script in Notepad++. This means that file names for data files and other scripts can be made relative rather than absolute. For example, if the script reads in a data file 'data.txt' and the data file is in the same directory as the script the file reference can simply be made as

```
data <- read.table("data.txt")
```

rather than using the absolute path,

```
data <- read.table("C:/path/to/data.txt")
```

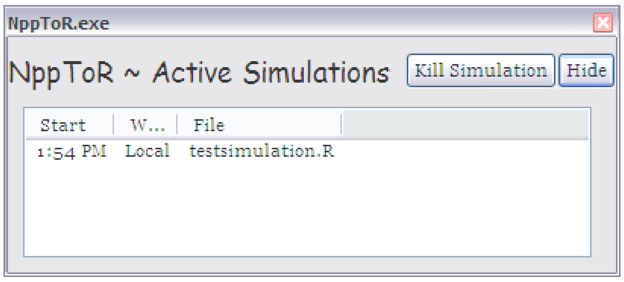
which is not only longer than necessary to type but not portable either. Simply moving a file or renaming a folder could break all the file references. For already open R sessions there is a menu item that will set the working directory to the current script directory.

NppToR is designed to be very small and non-intrusive. It is designed to sit in the background and is only active when Notepad++ is in use. Occasional users of R may not like having the utility running on startup; for those users when NppToR is run manually it will also launch Notepad++.

Other features

Although Notepad++ now has native support for the R language, it is a recent addition. NppToR also generates syntax files for use with Notepad++'s user defined language system, which is an artifact from when Notepad++ did not have support for R. This feature has been retained because it still offers advantages over the built in syntax rules. For example, NppToR generates the syntax files dynamically, and can take advantage of a library to include the keywords from installed packages. This method ensures that all major keywords are highlighted, and can easily be updated to possible changes in the R distribution, as well as the addition of new packages. The generated language differs also in enforcing case sensitivity and proper word boundaries.

R is an ideal environment for performing simulation studies. These simulations can run for a long time, and are typically run in the batch mode. NppToR not only provides an easy hotkey to run these simulations but also provides a simulation monitor to keep track of active simulations, monitor how long they have been running and provide an easy way to terminate those that need to be stopped early. Figure 3 shows the simulation monitor.



Start	W...	File
1:54 PM	Local	testsimulation.R

Figure 3: NppToR monitors active simulations allowing for an easy way to kill off simulations that have been running to long.

Summary

Notepad++ benefits from a large user base, and is constantly updated and improved, including several extensions that also help with developing for R. Notepad++ with NppToR mirrors the advantages of editors on other platforms but with a native windows feel. The power and strength of NppToR comes in its simplicity. It enhances an already powerful editor to become a superior editor for R.

Both Notepad++ and NppToR are easy to use and easy to learn, but as users become proficient with the vast array of keyboard commands and the macro system the true power of Notepad++ is revealed. NppToR should be considered by any who work with R on Windows.

Acknowledgments

The author's research was supported by a grant from the National Cancer Institute (CA57030).

Andrew Redd
Department of Statistics
Texas A&M University
3143 TAMU
College Station, TX 77843-3143, USA
aredd@stat.tamu.edu

Changes in R 2.10.1–2.11.1

by the R Core Team

R 2.11.1 changes

New features

- `R CMD INSTALL` checks if dependent packages are available early on in the installation of source packages, thereby giving clearer error messages.
- `R CMD INSTALL -build` now names the file in the format used for Mac OS X binary files on that platform.
- `BIC()` in package `stats4` now also works with multiple fitted models, analogously to `AIC()`.

Deprecated & defunct

- Use of file extension `.C` for C++ code in packages is now deprecated: it has caused problems for some `make`s on case-insensitive file systems (although it currently works with the recommended toolkits).

Installation changes

- Command `gnutar` is preferred to `tar` when configure sets `TAR`. This is needed on Mac OS 10.6, where the default `tar`, `bsdtar` 2.6.2, has been reported to produce archives with illegal extensions to tar (according to the POSIX standard).

Bug fixes

- The C function `mkCharLenCE` now no longer reads past `len` bytes (unlikely to be a problem except in user code). (PR#14246)
- On systems without any default `LD_LIBRARY_PATH` (not even `/usr/local/lib`), `[DY]LIB_LIBRARY_PATH` is now set without a trailing colon. (PR#13637)
- More efficient `utf8ToInt()` on long multi-byte strings with many multi-byte characters. (PR#14262)
- `aggregate.ts()` gave platform-depedent results due to rounding error for `ndeltat != 1`.
- `package.skeleton()` sometimes failed to fix filenames for .R or .Rd files to start with an alphanumeric. (PR#14253) It also failed when only an S4 class without any methods was defined. (PR#14280)
- `splinefun(*, method = "monoH.FC")` was not quite monotone in rare cases. (PR#14215)
- `Rhttpd` no longer crashes due to SIGPIPE when the client closes the connection prematurely. (PR#14266)
- `format.POSIXlt()` could cause a stack overflow and crash when used on very long vectors. (PR#14267)
- `Rd2latex()` incorrectly escaped special characters in `\usage` sections.
- `mcnemar.test()` could alter the levels (dropping unused levels) if passed `x` and `y` as factors (reported by Greg Snow).
- `Rd2pdf` sometimes needed a further `pdflatex` pass to get hyperlinked pages correct.
- `interaction()` produced malformed results when levels were duplicated, causing segfaults in `split()`.
- `cut(d, breaks = <n>)` now also works for `"Date"` or `"POSIXt"` argument `d`. (PR#14288)
- `memDecompress()` could decompress incompletely rare `xz`-compressed input due to incorrect documentation of `xz` utils. (Report and patch from Olaf Mersmann.)
- The S4 `initialize()` methods for `"matrix"`, `"array"`, and `"ts"` have been fixed to call `validObject()`. (PR#14284)
- `R CMD INSTALL` now behaves the same way with or without `-no-multiarch` on platforms with only one installed architecture. (It used to clean the `src` directory without `-no-multiarch`.)
- `[<-.data.frame` was not quite careful enough in assigning (and potentially deleting) columns right-to-left. (PR#14263)
- `rbeta(n, a,b)` no longer occasionally returns `NaN` for `a >> 1 > b`. (PR#14291)
- `pnorm(x, log.p = TRUE)` could return `NaN` not `-Inf` for x near (minus for `lower.tail=TRUE`) the largest representable number.
- Compressed data files `*.(txt|tab|csv).(gz|bz2|xz)` were not recognized for the list of data topics and hence for packages using LazyData. (PR#14273)

- `textConnection()` did an unnecessary translation on strings in a foreign encoding (e.g. UTF-8 strings on Windows) and so was slower than it could have been on very long input strings. (PR#14286)
- `tools::Rd2txt()` did not render poorly written Rd files consistently with other renderers.
- `na.action()` did not extract the `na.action` component as documented.

R 2.11.0 changes

Significant user-visible changes

- Packages must have been installed under R $\geq$ 2.10.0, as the current help system is the only one now supported.
- A port to 64-bit Windows is now available as well as binary package repositiories: see the 'R Administration and Installation Manual'.
- Argument matching for primitive functions is now done in the same way as for interpreted functions except for the deliberate exceptions

  ```
  call switch .C .Fortran .Call .External
  ```

  all of which use positional matching for their first argument, and also some internal-use-only primitives.
- The default device for command-line R at the console on Mac OS X is now `quartz()` and not `X11()`.

New features

- The `open` modes for connections are now interpreted more consistently. open = `"r"` is now equivalent to open = `"rt"` for all connections. The default open = "" now means `"rt"` for all connections except the compressed file connections `gzfile()`, `bzfile()` and `xzfile()` for which it means `"rb"`.
- `R CMD INSTALL` now uses the internal `untar()` in package utils: this ensures that all platforms can install `bzip2`- and `xz`-compressed tarballs. In case this causes problems (as it has on some Windows file systems when run from Cygwin tools) it can be overridden by the environment variable `R_INSTALL_TAR`: setting this to a modern external tar program will speed up unpacking of large (tens of Mb or more) tarballs.
- `help(try.all.packages = TRUE)` is much faster (although the time taken by the OS to find all the packages the first time it is used can dominate the time).
- `R CMD check` has a new option `-timings` to record per-example timings in file `<pkg>.Rcheck/<pkg>-Ex.timings`.
- The TRE library has been updated to version 0.8.0 (minor bugfixes).
- `grep[l]()`, `[g]sub()` and `[g]regexpr()` now work in bytes in an 8-bit locales if there is no marked UTF-8 input string: this will be somewhat faster, and for `[g]sub()` give the result in the native encoding rather than in UTF-8 (which returns to the behaviour prior to R 2.10.0).
- A new argument `skipCalls` has been added to `browser()` so that it can report the original context when called by other debugging functions.
- More validity checking of UTF-8 and MBCS strings is done by `agrep()` and the regular-expression matching functions.
- The undocumented restriction on `gregexpr()` to `length(text) > 0` has been removed.
- Package `tcltk` now sends strings to Tcl in UTF-8: this means that strings with a marked UTF-8 encoding are supported in non-UTF-8 locales.
- The graphics engine now supports rendering of raster (bitmap) images, though not all graphics devices can provide (full) support. Packages providing graphics devices (e.g., `Cairo`, `RSvgDevice`, `cairoDevice`) will need to be reinstalled.

  There is also support in the graphics engine for capturing raster images from graphics devices (again not supported on all graphics devices).
- `R CMD check` now also checks if the package and namespace can be unloaded: this provides a check of the `.Last.lib()` and `.onUnload()` hook functions (unless `-install=fake`).
- `prop.table(x)` now accepts a one-dimensional table for `x`.
- A new function `vapply()` has been added, based on a suggestion from Bill Dunlap. It requires that a template for the function value be specified, and uses it to determine the output type and to check for consistency in the function values.
- The main HTML help page now links to a reformatted copy of this `NEWS` file. (Suggested by Henrik Bengtsson.) Package index files link to the package `DESCRIPTION` and `NEWS` files and a list of demos when using dynamic help.
- The `[` method for class `"AsIs"` allows the next method to change the underlying class. (Wish of Jens Oehlschlägel.)

- `write.csv[2]()` no longer allow `append` to be changed: as ever, direct calls to `write.table()` give more flexibility as well as more room for error.
- The index page for HTML help for a package now collapses multiple signatures for S4 methods into a single entry.
- The use of `.required` by `require()` and `detach()` has been replaced by `.Depends` which is set from the `Depends` field of a package (even in packages with name spaces). By default `detach()` prevents such dependencies from being detached: this can be overridden by the argument `force`.
- `bquote()` has been extended to work on function definitions (wish of PR#14031).
- `detach()` when applied to an object other than a package returns the environment that has been detached, to parallel `attach()`.
- `readline()` in non-interactive use returns `""` and does not attempt to read from the "terminal".
- New function `file_ext()` in package `tools`.
- `xtfrm()` is now primitive and internally generic, as this allows S4 methods to be set on it without name-space scoping issues.

  There are now `"AsIs"` and `"difftime"` methods, and the default method uses `unclass(x)` if `is.numeric(x)` is true (which will be faster but relies on `is.numeric()` having been set correctly for the class).
- `is.numeric(x)` is now false for a `"difftime"` object (multiplication and division make no sense for such objects).
- The default method of `weighted.mean(x, w)` coerces `w` to be numeric (aka double); previously only integer weights were coerced. Zero weights are handled specially so an infinite value with zero weight does not force an `NaN` result.

  There is now a `"difftime"` method.
- `bug.report()` now has `package` and `lib.loc` arguments to generate bug reports about packages. When this is used, it looks for a `BugReports` field in the package `DESCRIPTION` file, which will be assumed to be a URL at which to submit the report, and otherwise generates an email to the package maintainer. (Suggested by Barry Rowlingson.)
- `quantile()` now has a method for the date-time class `"POSIXt"`, and types 1 and 3 (which never interpolate) work for Dates and ordered factors.
- `length(<POSIXlt>)` now returns the length of the corresponding abstract timedate-vector rather than always 9 (the length of the underlying list structure). (Wish of PR#14073 and PR#10507.)
- The readline completion backend no longer sorts possible completions alphabetically (e.g., function argument names) if R was built with readline ≥ 6.
- `select.list()` gains a `graphics` argument to allow Windows/Mac users to choose the text interface. This changes the behaviour of `new.packages(ask=TRUE)` to be like `update.packages(ask=TRUE)` on those platforms in using a text menu: use `ask="graphics"` for a graphical menu.
- New function `chooseBioCmirror()` to set the `"BioC_mirror"` option.
- The R grammar prevents using the argument `name` in signatures of S4 methods for `$` and `$<-`, since they will always be called with a character string value for `name`. The implicit S4 generic functions have been changed to reflect this: packages which included `name` in the signature of their methods need to be updated and re-installed.
- The handling of the `method` argument of `glm()` has been refined following suggestions by Ioannis Kosmidis and Heather Turner.
- `str()` gains a new argument `list.len` with default 99, limiting the number of `list()` items (per level), thanks to suggestions from David Winsenius.
- Having formal arguments of an S4 method in a different order from the generic is now an error (the warning having been ignored by some package maintainers for a long time).
- New functions `enc2native()` and `enc2utf8()` convert character vectors with possibly marked encodings to the current locale and UTF-8 respectively.
- Unrecognized escapes and embedded nuls in character strings are now an error, not just a warning. Thus option `"warnEscapes"` is no longer needed. `rawToChar()` now removes trailing nuls silently, but other embedded nuls become errors.
- Informational messages about masked objects displayed when a package is attached are now more compact, using `strwrap()` instead of one object per line.
- `print.rle()` gains argument `prefix`.

- `download.file()` gains a `"curl"` method, mainly for use on platforms which have `curl` but not `wget`, but also for some hard-to-access URLs.

- In Rd, `\eqn` and `\deqn` will render in HTML (and convert to text) upper- and lower-case Greek letters (entered as `\alpha` ...), `\ldots`, `\dots`, `\ge` and `\le`.

- `utf8ToInt()` and `intToUtf8()` now map `NA` inputs to `NA` outputs.

- `file()` has a new argument `raw` which may help if it is used with something other than a regular file, e.g. a character device.

- New function `strtoi()`, a wrapper for the C function `strtol`.

- `as.octmode()` and `as.hexmode()` now allow inputs of length other than one.

  The `format()` and `print()` methods for `"octmode"` now preserve names and dimensions (as those for `"hexmode"` did).

  The `format()` methods for classes `"octmode"` and `"hexmode"` gain a `width` argument.

- `seq.int()` returns an integer result in some further cases where `seq()` does, e.g. `seq.int(1L, 9L, by = 2L)`.

- Added `\subsection{}{}` macro to Rd syntax, for subsections within sections.

- n-dimensional arrays with dimension names can now be indexed by an n-column character matrix. The indices are matched against the dimension names. `NA` indices are propagated to the result. Unmatched values and `""` are not allowed and result in an error.

- `interaction(drop=TRUE)` uses less memory (related to PR#14121).

- `summary()` methods have been added to the `"srcref"` and `"srcfile"` classes, and various encoding issues have been cleaned up.

- If option `"checkPackageLicense"` is set to `TRUE` (not currently the default), users will be asked to agree to non-known-to-be-FOSS package licences at first use.

- Checking `setAs(a,b)` methods only gives a message instead of a warning, when one of `a` or `b` is unknown.

- New function `norm()` to compute a matrix norm. `norm()` and also `backsolve()` and `sample()` have implicit S4 generics.

- `Renviron.site` and `Rprofile.site` can have architecture-specific versions on systems with sub-architectures.

- `R CMD check` now (by default) also checks Rd files for auto-generated content in need of editing, and missing argument descriptions.

- `aggregate()` gains a formula method thanks to a contribution by Arni Magnusson. The data frame method now allows summary functions to return arbitrarily many values.

- `path.expand()` now propagates `NA` values rather than converting them to `"NA"`.

- `file.show()` now disallows `NA` values for file names, headers, and pager.

- The 'fuzz' used by `seq()` and `seq.int()` has been reduced from 1e-7 to 1e-10, which should be ample for the double-precision calculations used in R. It ensures that the fuzz never comes into play with sequences of integers (wish of PR#14169).

- The default value of `RSiteSearch(restrict=)` has been changed to include vignettes but to exclude R-help. The R-help archives available have been split, with a new option of `"Rhelp10"` for those from 2010.

- New function `rasterImage()` in the `graphics` package for drawing raster images.

- `stats:::extractAIC.coxph()` now omits aliased terms when computing the degrees of freedom (suggestion of Terry Therneau).

- `cor()` and `cov()` now test for misuse with non-numeric arguments, such as the non-bug report PR#14207.

- `pchisq(ncp =, log.p = TRUE)` is more accurate for probabilities near one. E.g. `pchisq(80, 4, ncp=1, log.p=TRUE)`. (Maybe what was meant in PR#14216.)

- `maintainer()` has been added, to give convenient access to the name of the maintainer of a package (contributed by David Scott).

- `sample()` and `sample.int()` allow zero items to be sampled from a zero-length input. `sample.int()` gains a default value `size=n` to be more similar to `sample()`.

- `switch()` returned `NULL` on error (not previously documented on the help page): it now does so invisibly, analogously to if-without-else.

  It is now primitive: this means that EXPR is always matched to the first argument and there is no danger of partial matching to later named arguments.

- Primitive functions `UseMethod()`, `attr()`, `attr<-()`, `on.exit()`, `retracemem()` and `substitute()` now use standard argument matching (rather than positional matching). This means that all multi-argument primitives which are not internal now use standard argument matching except where positional matching is desirable (as for `switch()`, `call()`, `.C()`, `...`).

- All the one-argument primitives now check that any name supplied for their first argument is a partial match to the argument name as documented on the help page: this also applies to replacement functions of two arguments.

- `base::which()` uses a new .Internal function when arr.ind is `FALSE` resulting in a 10x speedup. Thanks to Patrick Aboyoun for implementation suggestions.

- Help conversion to text now uses the first part of `\enc{}{}` markup if it is representable in the current output encoding. On the other hand, conversion to LaTeX with the default `outputEncoding = "ASCII"` uses the second part.

- A new class `"listOfMethods"` has been introduced to represent the methods in a methods table, to replace the deprecated class `"MethodsList"`.

- `any()` and `all()` return early if possible. This may speed up operations on long vectors.

- `strptime()` now accepts `"%z"` (for the offset from UTC in the RFC822 format of +/-hhmm).

- The PCRE library has been updated to version 8.02, a bug-fix release which also updates tables to Unicode 5.02.

- Functions which may use a graphical `select.list()` (including `menu()` and `install.packages()`) now check on a Unix-alike that Tk can be started (and not just `capabilities("tcltk") && capabilities("X11")`).

- The parser no longer marks strings containing octal or hex escapes as being in UTF-8 when entered in a UTF-8 locale.

- On platforms with cairo but not Pango (notably Mac OS X) the initial default `X11()` type is set to `"Xlib"`: this avoids several problems with font selection when done by cairo rather than Pango (at least on Mac OS X).

- New `arrayInd()` such that `which(x, arr.ind = TRUE)` for an array `x` is now equivalent to `arrayInd(which(x), dim(x), dimnames(x))`.

Deprecated & defunct

- Bundles of packages are defunct.

- `stats::clearNames()` is defunct: use `unname()`.

- Basic regular expressions are defunct, and `strsplit()`, `grep()`, `grepl()`, `sub()`, `gsub()`, `regexpr()` and `gregexpr()` no longer have an `extended` argument.

- `methods::trySilent()` is defunct.

- `index.search()` (which was deprecated in 2.10.0) is no longer exported and has a different argument list.

- Use of multiple arguments to `return()` is now defunct.

- The use of `UseMethod()` with more than two arguments is now defunct.

- In the `methods` package, the MethodsList metadata objects which had been superseded by hash tables (environments) since R 2.8.0 are being phased out. Objects of this class are no longer assigned or used as metadata by the package.

  `getMethods()` is now deprecated, with its internal use replaced by `findMethods()` and other changes. Creating objects from the MethodsList class is also deprecated.

- Parsing strings containing both octal/hex and Unicode escapes now gives a warning and will become an error in R 2.12.0.

Installation changes

- UTF-8 is now used for the reference manual and package manuals. This requires LaTeX `2005/12/01` or later.

- configure looks for a POSIX compliant `tr`, Solaris's `/usr/ucb/tr` having been found to cause `Rdiff` to malfunction.

- configure is now generated with autoconf-2.65, which works better on recent systems and on Mac OS X.

Package installation changes

- Characters in R source which are not translatable to the current locale are now handled more tolerantly: these will be converted to hex codes with a warning. Such characters are only really portable if they appear in comments.

- `R CMD INSTALL` now tests that the installed package can be loaded (and backs out the installation if it cannot): this can be suppressed by `-no-test-load`. This avoids installing/updating a package that cannot be used: common causes of failures to load are missing/incompatible external software and missing/broken dependent packages.

- Package installation on Windows for a package with a src directory now checks if a DLL is created unless there is a `src/Makefile.win` file: this helps catch broken installations where the toolchain has not reported problems in building the DLL. (Note: this can be any DLL, not just one named `<pkg-name>.dll`.)

Bug fixes

- Using `with()`, `eval()` etc with a list with some unnamed elements now works. (PR#14035)

- The "quick" dispatch of S4 methods for primitive functions was not happening, forcing a search each time. (Dispatch for closures was not affected.) A side effect is that default values for arguments in a method that do not have defaults in the generic will now be ignored.

- Trying to dispatch S4 methods for primitives during the search for inherited methods slows that search down and potentially could cause an infinite recursion. An internal switch was added to turn off all such methods from `findInheritedMethods()`.

- R framework installation (on Mac OS X) would not work properly if a rogue Resources directory was present at the top level. Such a non-symlink will now be renamed to Resources.old (and anything previously named Resources.old removed) as part of the framework installation process.

- The checks for conforming S4 method arguments could fail when the signature of the generic function omitted some of the formal arguments (in addition to `...`). Arguments omitted from the method definition but conforming (per the documentation) should now be ignored (treated as `"ANY"`) in dispatching.

- The computations for S4 method evaluation when `...` was in the signature could fail, treating `...` as an ordinary symbol. This has been fixed, for the known cases.

- Various `ar()` fitting methods have more protection for singular fits.

- `callNextMethod()` now works again with the `drop=` argument in `‘`

- `parse()` and `parse_Rd()` miscounted columns when multibyte UTF-8 characters were present.

- Formatting of help pages has had minor improvements: extra blank lines have been removed from the text format, and empty package labels removed from HTML.

- `cor(A, B)` where A has $n \times 1$ and B a 1-dimensional array segfaulted or gave an internal error. (The case `cor(B, A)` was PR#7116.)

- `cut.POSIXt()` applied to a start value after the DST transition on a DST-change day could give the wrong time for `breaks` in units of days or longer. (PR#14208)

- `do_par()` UNPROTECTed too early (PR#14214)

- subassignment `x[[....]]  <- y` didn't check for a zero-length right hand side, and inserted rubbish value. (PR#14217)

- `fisher.test()` no longer gives a P-value *very* slightly > 1, in some borderline cases.

- Internal function `matchArgs()` no longer modifies the general purpose bits of the SEXPs that make up the formals list of R functions. This fixes an invalid error message that would occur when a garbage collection triggered a second call to `matchArgs` for the same function via a finalizer.

- `gsub()` in 2.10.x could fail from stack overflow for extremely long strings due to temporary data being allocated on the stack. Also, `gsub()` with `fixed=TRUE` is in some circumstances considerably faster.

- Several primitives, including `attributes()`, `attr<-()` `interactive()`, `nargs()` and `proc.time()`, did not check that they were called with the correct number of arguments.

- A potential race condition in `list.files()` when other processes are operating on the directory has been fixed; the code now dynamically allocates memory for file listings in a single pass instead of making an initial count pass.

- `mean(x, trim=, na.rm = FALSE)` failed to return `NA` if `x` contained missing values. (Reported by Bill Dunlap.)

- Extreme tail behavior of, `pbeta()` and hence `pf()`, e.g., `pbeta(x, 3, 2200, lower.tail=FALSE, log.p=TRUE)` now returns finite values instead of jumping to `-Inf` too early (PR#14230).

- `parse(text=x)` misbehaved for objects `x` that were not coerced internally to character, notably symbols. (Reported to R-devel by Bill Dunlap.)

- The internal C function `coerceSymbol` now handles coercion to character, and warns if coercion fails (rather than silently returning `NULL`). This allows a name to be given where a character vector is required in functions which coerce internally.

- The interpretation by `strptime()` of that it is ever advisable to use locale- and system-specific input formats).

- `capabilities("X11")` now works the same way on Mac OS X as on other platforms (and as documented: it was always true for R built with `-with-aqua`, as the CRAN builds are).

- The `X11()` device with cairo but not Pango (notably Mac OS X) now checks validity of text strings in UTF-8 locales (since Pango does but cairo it seems does not).

- `read.fwf()` misread multi-line records when n was specified. (PR#14241)

- `all.equal(*, tolerance = e)` passes the numeric tolerance also to the comparison of the attributes.

- `pgamma(0,0)`, a boundary case, now returns 0, its limit from the left, rather than the limit from the right.

- Issuing POST requests to the internal web server could stall the request under certain circumstances.

- `gzcon( <textConnection> )`, an error, no longer damages the connection (in a way to have it seg.fault). (PR#14237)

- All the results from `hist()` now use the nominal `breaks` not those adjusted by the numeric `fuzz`: in recent versions the nominal `breaks` were reported but the `density` referred to the intervals used in the calculation — which mattered very slightly for one of the extreme bins. (Based on a report by Martin Becker.)

- If `xy[z].coords()` (used internally by many graphics functions) are given a list as `x`, they now check that the list has suitable names and give a more informative error message. (PR#13936)

R 2.10.1 patched changes

New features

- The handling of line textures in the `postscript()` and `pdf()` devices was set up for round end caps (the only type which existed at the time): it has now been adjusted for butt endcaps.

- `lchoose(a,k)` is now defined as `log(abs(choose(a,k)))`, analogously to `lfactorial()`.

- Although `\eqn{}` in Rd files is defined as a `verbatim` macro, many packages expected `\dots` and `\ldots` to be interpreted there (as was the case in R < 2.10.0), so this is now done (using an ellipsis in HTML rendering).

- Escaping of braces in quoted strings in R-code sections of Rd files is allowed again. This had been changed for the new Rd format in R 2.10.0 but was only documented on the developer site and was handled inconsistently by the converters: text and example conversion removed the escapes but HTML conversion did not.

- The PCRE library has been updated to version 8.01, a bug-fix release.

- `tools::readNEWS()` now accepts a digit as the first character of a news section.

Bug fixes

- Using `read.table(header=TRUE)` on a header with an embedded new line would copy part of the header into the data. (PR#14103)

- `qpois(p = 1, lambda = 0)` now gives 0 as for all other `p`. (PR#14135)

- Functions related to string comparison (e.g. `unique()`, `match()`) could cause crashes when used with strings not in the native encoding, e.g. UTF-8 strings on Windows. (PR#14114 and PR#14125)

- `x[ , drop=TRUE]` dropped an `NA` level even if it was in use.

- The dynamic HTML help system reported the wrong MIME type for the style sheet.

- `tools::codoc()` (used by `R CMD check`) was missing cases where the function had no arguments but was documented to have some.

- Help links containing special characters (e.g. `"?"`) were not generated correctly when rendered in HTML. (PR#14155)

- `lchoose(a,k)` no longer wrongly gives `NaN` for negative `a`.
- `ks.test()` could give a p-value that was off by one observation due to rounding error. (PR#14145)
- `readBin()`/`readChar()` when reading millions of character strings in a single call used excessive amounts of memory (which also slowed them down).
- `R CMD SHLIB` could fail if used with paths that were not alphanumeric, e.g. contained `+`. (PR#14168)
- `sprintf()` was not re-entrant, which potentially caused problems if an `as.character()` method called it.
- The `quartz()` device did not restore the clipping region when filling the background for a new page. This could be observed in multi-page bitmap output as stale outer regions of the plot.
- `p.adjust(*, method, n)` now works correctly for the rare case `n > length(p)`, also when method differs from `"bonferroni"` or `"none"`, thanks to a patch from Gordon Smyth.
- `tools::showNonASCII()` failed to detect non-ASCII characters if `iconv()` (incorrectly) converted them to different ASCII characters. (Seen on Windows only.)
- `tcrossprod()` wrongly failed in some cases when one of the arguments was a vector and the other a matrix.
- `[cr]bind(..., deparse.level=2)` was not always giving names when documented to do so. (Discovered whilst investigating PR#14189.)
- `match(incomparables=<non-NULL>)` could in rare cases infinite- loop.
- `poisson.test()` needed to pass `conf.level` to `binom.test()` (PR#14195)
- The `"nls"` method for `df.residual()` gave incorrect results for models fitted with `na.action = na.exclude`. (PR#14194)
- A change to `options(scipen=)` was only implemented when printing next occurred, even though it should have affected intervening calls to `axis()`, `contour()` and `filledcontour()`.
- `prettyNum(drop0trailing=TRUE)` did not handle signs of imaginary parts of complex numbers correctly (and this was used by `str()`: PR#14201).
- `system.time()` had sys.child component wrong (copied user.child instead) on systems with `HAVE_GETRUSAGE` (PR#14210)
- Changing both line texture and line cap (end) resulted in the latter to be ommitted form the PDF code. In addition, line cap (end) and join are now set explicitly in PDF output to ensure correct defaults.
- The suppression of auto-rotation in `bitmap()` and `dev2bitmap()` with the `pdfwrite` device was not working correctly.
- `plot(ecdf(), log="x")` no longer gives an incorrect warning.
- `read.fwf()` works again when `file` is a connection.
- Startup files will now be found if their paths exceed 255 bytes. (PR#14228)
- `contrasts<-` (in the `stats` package) no longer has an undeclared dependence on `methods` (introduced in 2.10.0).

R 2.10.1 changes since previous Journal Issue

(Most of the changes to what was then “2.10.0 Patched” were described in Vol 1/2.)

Bug fixes

- `switch(EXPR = "A")` now returns `NULL`, as `switch(1)` which used to signal an error.

Changes on CRAN

2009-12-29 to 2010-06-03 for CRAN task views
2009-10-23 to 2010-06-03 for CRAN packages

by Kurt Hornik and Achim Zeileis

New CRAN task views

Phylogenetics Phylogenetics, especially comparative methods.
Packages: **MCMCglmm**, **PHYLOGR**, **PhySim**, **TreeSim**, **ade4**, **adephylo**, **apTreeshape**, **ape***, **geiger**, **laser**, **maticce**, **ouch**, **paleoTS**, **phangorn**, **phybase**, **phyclust**, **phylobase**, **phyloclim**, **picante**, **scaleboot**, **vegan**.
Maintainer: Brian O'Meara.

(* = core package)

New packages in CRAN task views

Bayesian **GLMMarp**, **zic**.

Cluster **apcluster**.

Econometrics **BayesPanel**, **VGAM**, **effects**, **fxregime**, **gam**, **gamlss**, **mgcv**, **micEconCES**, **micEconSNQP**, **truncreg**.

Environmetrics **FD**.

ExperimentalDesign **DiceDesign**, **DiceKriging**, **FrF2.catlg128**, **dae**.

Finance **SV**.

HighPerformanceComputing **Rdsm**, **batch**, **biglars**, **doMPI**, **doRedis**.

MedicalImaging **PTAk**, **cudaBayesreg**, **dcemriS4***, **oro.dicom***, **oro.nifti***.

Psychometrics **BradleyTerry2**, **CMC**, **caGUI**, **lavaan***, **lordif**, **pscl**.

Spatial **Guerry**, **cshapes**, **rworldmap**.

Survival **AIM**, **Cprob**, **DTDA**, **TraMineR**, **flexCrossHaz**, **ipw**, **mspath**.

TimeSeries **rtv**.

gR **parcor**.

(* = core package)

New contributed packages

AIM AIM: adaptive index model. Authors: L. Tian and R. Tibshirani. In view: *Survival*.

B2Z Bayesian Two-Zone Model. Authors: João Vitor Dias Monteiro, Sudipto Barnejee and Gurumurthy Ramachandran.

BBMM This package fits a Brownian bridge movement model to observed locations in space and time. Authors: Ryan Nielson, Hall Sawyer and Trent McDonald.

BLR Bayesian Linear Regression. Authors: Gustavo de los Campos and Paulino Pérez Rodríguez.

BTSPAS Bayesian Time-Stratified Population Analysis. Authors: Carl J Schwarz and Simon J Bonner.

BayesPanel Bayesian methods for panel data modeling and inference. Authors: Chunhua Wu and Siddhartha Chib. In view: *Econometrics*.

BayesQTLBIC Bayesian multi-locus QTL analysis based on the BIC criterion. Author: Rod Ball.

Bergm Bayesian inference for exponential random graph models. Authors: Alberto Caimo and Nial Friel.

BioPhysConnectoR Connecting sequence information and biophysical models Authors: Franziska Hoffgaard, with contributions from Philipp Weil and Kay Hamacher.

BioStatR Companion to the book "Exercices et problèmes de statistique avec le logiciel R". Authors: Frederic Bertrand and Myriam Maumy-Bertrand.

Bmix Bayesian sampling for stick-breaking mixtures. Author: Matt Taddy. In view: *Cluster*.

BoSSA A bunch of structure and sequence analysis. Author: Pierre Lefeuvre.

Bolstad2 Bolstad functions. Author: James Curran.

BoolNet Generation, reconstruction, simulation and analysis of synchronous, asynchronous, and probabilistic Boolean networks. Authors: Christoph Müssel, Martin Hopfensitz, Dao Zhou and Hans Kestler.

Boruta A tool for finding significant attributes in information systems. Authors: Algorithm by Witold R. Rudnicki, R implementation by Miron B. Kursa.

BradleyTerry2 Bradley-Terry models. Authors: Heather Turner and David Firth. In view: *Psychometrics*.

CAVIAR Cambial Activity and wood formation data processing, VIsualisation and Analysis using R. Author: Cyrille Rathgeber.

CCMtools Clustering through "Correlation Clustering Model" (CCM) and cluster analysis tools. Author: Mathieu Vrac.

CCP Significance tests for Canonical Correlation Analysis (CCA). Author: Uwe Menzel.

CMC Cronbach-Mesbah Curve. Authors: Michela Cameletti and Valeria Caviezel. In view: *Psychometrics*.

COMPoissonReg Conway-Maxwell Poisson (COM-Poisson) Regression. Author: Kimberly Sellers.

CausalGAM Estimation of causal effects with Generalized Additive Models. Authors: Adam Glynn and Kevin Quinn.

CoCoCg Graphical modelling by CG regressions. Author: Jens Henrik Badsberg.

CoCoGraph Interactive and dynamic graphs for the CoCo objects. Author: Jens Henrik Badsberg.

CollocInfer Collocation inference for dynamic systems. Authors: Giles Hooker, Luo Xiao and James Ramsay.

CorrBin Nonparametrics with clustered binary data. Author: Aniko Szabo.

Cprob Conditional probability function of a competing event. Author: Arthur Allignol. In view: *Survival*.

DAMisc Dave Armstrong's miscellaneous functions. Author: Dave Armstrong.

DEMEtics Calculation of Gst and D values characterizing the genetic differentiation between populations. Authors: Alexander Jueterbock, Philipp Kraemer, Gabriele Gerlach and Jana Deppermann.

DOSim Computation of functional similarities between DO terms and gene products; DO enrichment analysis; Disease Ontology annotation. Author: Jiang Li.

DatABEL File-based access to large matrices stored on HDD in binary format. Authors: Yurii Aulchenko and Stepan Yakovenko.

DeducerPlugInExample Deducer plug-in example. Author: Ian Fellows.

DesignPatterns Design patterns in R to build reusable object-oriented software. Authors: Jitao David Zhang, supported by Stefan Wiemann and Wolfgang Huber.

DiceDesign Designs of computer experiments. Authors: Jessica Franco, Delphine Dupuy and Olivier Roustant. In view: *ExperimentalDesign*.

DiceEval Construction and evaluation of metamodels. Authors: D. Dupuy and C. Helbert.

DiceKriging Kriging methods for computer experiments. Authors: O. Roustant, D. Ginsbourger and Y. Deville. In view: *ExperimentalDesign*.

DiceOptim Kriging-based optimization for computer experiments. Authors: D. Ginsbourger and O. Roustant.

DistributionUtils Distribution utilities. Author: David Scott.

DoseFinding Planning and analyzing dose finding experiments. Authors: Björn Bornkamp, José Pinheiro and Frank Bretz.

EMT Exact Multinomial Test: Goodness-of-fit test for discrete multivariate data. Author: Uwe Menzel.

FAMT Factor Analysis for Multiple Testing (FAMT): Simultaneous tests under dependence in high-dimensional data. Authors: David Causeur, Chloé Friguet, Magalie Houée-Bigot and Maela Kloareg.

FNN Fast Nearest Neighbor search algorithms and applications. Author: Shengqiao Li.

FrF2.catlg128 Complete catalogues of resolution IV 128 run 2-level fractional factorials up to 24 factors. Author: Ulrike Grömping. In view: *ExperimentalDesign*.

FunctSNP SNP annotation data methods and species specific database builder. Authors: S. J. Goodswen with contributions from N. S. Watson-Haigh, H. N. Kadarmideen and C. Gondro.

GAMens Applies GAMbag, GAMrsm and GAMens ensemble classifiers for binary classification. Authors: Koen W. De Bock, Kristof Coussement and Dirk Van den Poel.

GEVcdn GEV conditonal density estimation network. Author: Alex J. Cannon.

GGally Extension to **ggplot2**. Authors: Barret Schloerke, Di Cook, Heike Hofmann and Hadley Wickham.

GWRM A package for fitting Generalized Waring Regression Models. Authors: A.J. Saez-Castillo, J. Rodríguez-Avi, A. Conde-Sánchez, M.J. Olmo-Jiménez and A.M. Martinez-Rodriguez.

GeneralizedHyperbolic The generalized hyperbolic distribution. Author: David Scott.

GrassmannOptim Grassmann Manifold Optimization. Authors: Kofi Placid Adragni and Seongho Wu.

Guerry Guerry: maps, data and methods related to Guerry (1833) "Moral Statistics of France". Authors: Michael Friendly and Stephane Dray. In view: *Spatial*.

HDMD Statistical analysis tools for High Dimension Molecular Data (HDMD). Author: Lisa McFerrin.

HGLMMM Hierarchical Generalized Linear Models. Author: Marek Molas.

HMM Hidden Markov Models. Authors: Scientific Software Development — Dr. Lin Himmelmann.

HistData Data sets from the history of statistics and data visualization. Authors: Michael Friendly, with contributions by Stephane Dray and Hadley Wickham.

IFP Identifying functional polymorphisms in genetic association studies. Author: Leeyoung Park.

Imap Interactive Mapping. Author: John R. Wallace.

JJcorr Calculates polychorical correlations for several copula families. Authors: Jeroen Ooms and Joakim Ekström.

JOP Joint Optimization Plot. Authors: Sonja Kuhnt and Nikolaus Rudak.

LDdiag Link function and distribution diagnostic test for social science researchers. Author: Yongmei Ni.

LS2W Locally stationary two-dimensional wavelet process estimation scheme. Authors: Idris Eckley and Guy Nason.

LiblineaR Linear predictive models based on the Liblinear C/C++ Library. Author: Thibault Helleputte.

LogitNet Infer network based on binary arrays using regularized logistic regression. Authors: Pei Wang, Dennis Chao and Li Hsu.

MADAM This package provides some basic methods for meta-analysis. Author: Karl Kugler.

MAMA Meta-Analysis of MicroArray. Author: Ivana Ihnatova.

MAc Meta-analysis with correlations. Author: AC Del Re and William T. Hoyt.

MAd Meta-analysis with mean differences. Author: AC Del Re and William T. Hoyt.

MFDF Modeling Functional Data in Finance. Author: Wei Dou.

MImix Mixture summary method for multiple imputation. Authors: Russell Steele, Naisyin Wang and Adrian Raftery.

MNM Multivariate Nonparametric Methods. An approach based on spatial signs and ranks. Authors: Klaus Nordhausen, Jyrki Mottonen and Hannu Oja.

MTSKNN Multivariate two-sample tests based on K-Nearest-Neighbors. Authors: Lisha Chen, Peng Dai and Wei Dou.

MVpower Give power for a given effect size using multivariate classification methods. Authors: Raji Balasubramanian and Yu Guo.

MaXact Exact max-type Cochran-Armitage trend test (CATT). Authors: Jianan Tian and Chenliang Xu.

ModelGood Validation of prediction models. Author: Thomas A. Gerds.

MortalitySmooth Smoothing Poisson counts with P-splines. Author: Carlo G. Camarda.

MplusAutomation Automating Mplus model estimation and interpretation. Author: Michael Hallquist.

MuMIn Multi-model inference. Author: Kamil Bartoń.

NMF Algorithms and framework for Nonnegative Matrix Factorization (NMF). Author: Renaud Gaujoux.

NMFN Non-Negative Matrix Factorization. Authors: Suhai (Timothy) Liu based on multiplicative updates (Lee and Sung 2001) and alternating least squares algorithm; Lars Kai Hansen's `nnmf_als` Matlab implementation; Torsten Hothorn's Moore-Penrose inverse function.

OligoSpecificitySystem Oligo Specificity System. Authors: Rory Michelland and Laurent Cauquil.

PBSadmb PBS ADMB. Authors: Jon T. Schnute and Rowan Haigh.

PKgraph Model diagnostics for pharmacokinetic models. Author: Xiaoyong Sun.

PKmodelFinder Software for Pharmacokinetic model. Authors: Eun-Kyung Lee, Gyujeong Noh and Hyeong-Seok Lim.

PermuteNGS Significance testing of transcriptome profiling for RNA-sequencing data. Authors: Taeheon Lee, Bo-Young Lee, Hee-Seok Oh, Heebal Kim.

PowerTOST Power and sample size based on two-one-sided-t-tests (TOST) for BE studies. Author: Detlew Labes.

ProDenICA Product Density Estimation for ICA using tilted Gaussian density estimates. Authors: Trevor Hastie and Rob Tibshirani.

PropCIs Computes confidence intervals for proportions, differences in proportions, an odds-ratio and the relative risk in a 2×2 table. Author: Ralph Scherer.

QT QT Knowledge Management System. Author: Christoffer W. Tornoe.

QTLNetworkR Interactive software package for QTL visualization. Author: Zheng Wenjun.

R2Cuba Multidimensional numerical integration. Authors: The Cuba library has been written by Thomas Hahn; Interface to R was written by Annie Bouvier and Kiên Kiêu.

R2PPT Simple R Interface to Microsoft PowerPoint using **rcom**. Author: Wayne Jones.

RANN Fast Nearest Neighbour Search. Authors: Samuel E. Kemp and Gregory Jefferis.

RFLPtools Tools to analyse RFLP data. Authors: Fabienne Flessa, Alexandra Kehl, Matthias Kohl.

RFinanceYJ Japanese stock market from Yahoo!-finance-Japan Authors: Yohei Sato and Nobuaki Oshiro.

RGtk2DfEdit Improved data frame editor for **RGtk2**. Authors: Tom Taverner, with contributions from John Verzani.

RH2 DBI/RJDBC interface to H2 Database. Author: G. Grothendieck. Author of H2 is Thomas Mueller.

RODM R interface to Oracle Data Mining. Authors: Pablo Tamayo and Ari Mozes.

ROracleUI Convenient Tools for Working with Oracle Databases. Author: Arni Magnusson.

RPPanalyzer Reads, annotates, and normalizes reverse phase protein array data. Author: Heiko Mannsperger with contributions from Stephan Gade.

RProtoBuf R interface to the Protocol Buffers API. Authors: Romain François and Dirk Eddelbuettel.

RSQLite.extfuns Math and String Extension Functions for **RSQLite**. Author: Seth Falcon.

Rcgmin Conjugate gradient minimization of nonlinear functions with box constraints. Author: John C. Nash.

RcmdrPlugin.MAc Meta-analysis with correlations (MAc) **Rcmdr** Plug-in. Author: AC Del Re.

RcmdrPlugin.MAd Meta-analysis with mean differences (MAd) **Rcmdr** Plug-in. Author: AC Del Re.

RcmdrPlugin.PT Some discrete exponential dispersion models: Poisson-Tweedie. Authors: David Pechel Cactcha, Laure Pauline Fotso and Célestin C Kokonendji.

RcmdrPlugin.SLC SLC **Rcmdr** plug-in. Authors: Antonio Solanas and Rumen Manolov.

RcmdrPlugin.doex **Rcmdr** plugin for Stat 4309 course. Author: Erin Hodgess.

RcmdrPlugin.sos Efficiently search R Help pages. Author: Liviu Andronic.

RcmdrPlugin.steepness Steepness **Rcmdr** plug-in. Author: David Leiva and Han de Vries.

RcppArmadillo Rcpp/Armadillo bridge. Authors: Romain François, Dirk Eddelbuettel and Doug Bates.

RcppExamples Examples using Rcpp to interface R and C++. Authors: Dirk Eddelbuettel and Romain François, based on code written during 2005 and 2006 by Dominick Samperi.

Rdsm Threads-Like Environment for R. Author: Norm Matloff. In view: *HighPerformanceComputing*.

RecordLinkage Record linkage in R. Authors: Andreas Borg and Murat Sariyar.

Rhh Calculating multilocus heterozygosity and heterozygosity-heterozygosity correlation. Authors: Jussi Alho and Kaisa Välimäki.

RobLoxBioC Infinitesimally robust estimators for preprocessing omics data. Author: Matthias Kohl.

RpgSQL DBI/RJDBC interface to PostgreSQL database. Author: G. Grothendieck.

Rsolnp General non-linear optimization. Authors: Alexios Ghalanos and Stefan Theussl.

Rvmmin Variable metric nonlinear function minimization with bounds constraints. Author: John C. Nash.

SDMTools Species Distribution Modelling Tools: Tools for processing data associated with species distribution modelling exercises. Authors: Jeremy VanDerWal, Luke Shoo and Stephanie Januchowski. This package was made possible in part by support from Lorena Falconi and Collin Storlie, and financial support from the Australian Research Council & ARC Research Network for Earth System Science.

SDisc Integrated methodology for the identification of homogeneous profiles in data distribution. Author: Fabrice Colas.

SHIP SHrinkage covariance Incorporating Prior knowledge. Authors: Monika Jelizarow and Vincent Guillemot.

SLC Slope and level change. Authors: Antonio Solanas, Rumen Manolov and Patrick Onghena.

SOAR Memory management in R by delayed assignments. Authors: Bill Venables, based on original code by David Brahm.

SPACECAP A program to estimate animal aabundance and density using Spatially-Explicit Capture-Recapture. Authors: Pallavi Singh, Arjun M. Gopalaswamy, Andrew J. Royle, N. Samba Kumar and K. Ullas Karanth with contributions from Sumanta Mukherjee, Vatsaraj and Dipti Bharadwaj.

SQN Subset quantile normalization. Author: Zhijin(Jean) Wu.

SV Indirect inference in non-Gaussian stochastic volatility models. Author: Øivind Skare. In view: *Finance*.

SampleSizeMeans Sample size calculations for normal means. Authors: Lawrence Joseph and Patrick Belisle.

SampleSizeProportions Calculating sample size requirements when estimating the difference between two binomial proportions. Authors: Lawrence Joseph, Roxane du Berger and Patrick Belisle.

SkewHyperbolic The Skew Hyperbolic Student t-distribution. Authors: David Scott and Fiona Grimson.

Sleuth2 Data sets from Ramsey and Schafer's "Statistical Sleuth (2nd ed)". Authors: Original by F.L. Ramsey and D.W. Schafer, modifications by Daniel W Schafer, Jeannie Sifneos and Berwin A Turlach.

SpatialEpi Performs various spatial epidemiological analyses. Author: Albert Y. Kim.

StMoSim Simulate QQ-Norm and TA-Plot. Author: Matthias Salvisberg.

TANOVA Time Course Analysis of Variance for Microarray. Authors: Baiyu Zhou and Weihong Xu.

TGUICore Teaching GUI — Core functionality. Authors: Matthias Templ, Gerlinde Dinges, Alexander Kowarik and Bernhard Meindl.

TGUITeaching Teaching GUI — prototype. Authors: Matthias Templ, Gerlinde Dinges, Alexander Kowarik and Bernhard Meindl.

ToxLim Incorporating ecological data and associated uncertainty in bioaccumulation modeling. Authors: Frederik De Laender and Karline Soetaert.

TreeRank Implementation of the TreeRank methodology for building tree-based ranking rules for bipartite ranking through ROC curve optimization. Author: Nicolas Baskiotis.

TreeSim Simulating trees under the birth-death model. Author: Tanja Stadler. In view: *Phylogenetics*.

UScensus2000 US Census 2000 suite of R Packages. Author: Zack W. Almquist. In view: *Spatial*.

UScensus2000add US Census 2000 suite of R Packages: Demographic add. Author: Zack W. Almquist.

UScensus2000blkgrp US Census 2000 block group shapefiles and additional Demographic Data. Author: Zack W. Almquist.

UScensus2000cdp US Census 2000 designated places shapefiles and additional demographic data. Author: Zack W. Almquist.

UScensus2000tract US Census 2000 tract level shapefiles and additional demographic Data. Author: Zack W. Almquist.

Unicode Unicode data and utilities. Author: Kurt Hornik.

VHDClassification Discrimination/Classification in very high dimension with linear and quadratic rules. Author: Robin Girard.

VPdtw Variable Penalty Dynamic Time Warping. Authors: David Clifford and Glenn Stone.

VhayuR Vhayu R interface. Author: The Brookhaven Group. In view: *TimeSeries*.

VizCompX Visualisation of Computer Models. Author: Neil Diamond.

WMBrukerParser William and Mary Parser for Bruker-Ultraflex mass spectrometry data. Authors: William Cooke, Maureen Tracy and Dariya Malyarenko.

WaveCD Wavelet change point detection for array CGH data. Author: M. Shahidul Islam.

adephylo Exploratory analyses for the phylogenetic comparative method. Authors: Thibaut Jombart and Stephane Dray. In view: *Phylogenetics*.

agilp Extracting and preprocessing Agilent express arrays. Author: Benny Chain.

amba Additive Models for Business Applications. Author: Charlotte Maia.

anesrake ANES Raking Implementation. Author: Josh Pasek.

apcluster Affinity Propagation Clustering. Authors: Ulrich Bodenhofer and Andreas Kothmeier. In view: *Cluster*.

aqp Algorithms for Quantitative Pedology. Author: Dylan Beaudette.

aroma.cn Copy-number analysis of large microarray data sets. Authors: Henrik Bengtsson and Pierre Neuvial.

aroma.light Light-weight methods for normalization and visualization of microarray data using only basic R data types. Author: Henrik Bengtsson.

asbio A collection of statistical tools for biologists. Authors: Ken Aho; many thanks to V. Winston and D. Roberts.

asd Simulations for adaptive seamless designs. Author: Nick parsons.

base64 Base 64 encoder/decoder. Authors: Romain François, based on code by Bob Trower available at `http://base64.sourceforge.net/`.

batch Batching routines in parallel and passing command-line arguments to R. Author: Thomas Hoffmann. In view: *HighPerformanceComputing*.

bfast BFAST: Breaks For Additive Seasonal and Trend. Authors: Jan Verbesselt, with contributions from Rob Hyndman.

bibtex BibTeX parser. Author: Romain François.

biganalytics A library of utilities for `"big.matrix"` objects of package **bigmemory**. Authors: John W. Emerson and Michael J. Kane.

biglars Scalable Least-Angle Regression and Lasso. Authors: Mark Seligman, Chris Fraley and Tim Hesterberg. In view: *HighPerformanceComputing*.

bigtabulate `table`-, `tapply`-, and `split`-like functionality for `"matrix"` and `"big.matrix"` objects. Authors: Michael J. Kane and John W. Emerson.

binhf Haar-Fisz functions for binomial data. Author: Matt Nunes.

boolfun Cryptographic Boolean functions. Author: F. Lafitte.

bootruin A bootstrap test for the probability of ruin in the classical risk process. Authors: Benjamin Baumgartner, Riccardo Gatto.

catnet Categorical Bayesian network inference. Authors: Nikolay Balov and Peter Salzman.

clusterCons Calculate the consensus clustering result from re-sampled clustering experiments with the option of using multiple algorithms and parameter. Authors: Dr. T. Ian Simpson.

cmaes Covariance Matrix Adapting Evolutionary Strategy. Authors: Heike Trautmann, Olaf Mersmann and David Arnu.

coarseDataTools A collection of functions to help with analysis of coarse infectious disease data. Author: Nicholas G. Reich.

codep Multiscale codependence analysis. Authors: Guillaume Guenard, with contributions from Bertrand Pages.

coenoflex Gradient-Based Coenospace Vegetation Simulator. Author: David W. Roberts.

compute.es Compute Effect Sizes. Author: AC Del Re.

corrplot Visualization of a correlation matrix. Author: Taiyun Wei.

crmn CCMN and other noRMalizatioN methods for metabolomics data. Author: Henning Redestig.

cthresh Complex wavelet denoising software. Author: Stuart Barber.

cubature Adaptive multivariate integration over hypercubes. Authors: C code by Steven G. Johnson, R by Balasubramanian Narasimhan.

cusp Cusp Catastrophe Model Fitting Using Maximum Likelihood. Author: Raoul P. P. P. Grasman.

dae Functions useful in the design and ANOVA of experiments. Author: Chris Brien. In view: *ExperimentalDesign*.

dagR R functions for directed acyclic graphs. Author: Lutz P Breitling.

datamap A system for mapping foreign objects to R variables and environments. Author: Jeffrey Horner.

dcemriS4 A Package for Medical Image Analysis (S4 implementation). Authors: Brandon Whitcher and Volker Schmid, with contributions from Andrew Thornton. In view: *MedicalImaging*.

dclone Data Cloning and MCMC Tools for Maximum Likelihood Methods. Author: Peter Solymos.

decon Deconvolution Estimation in Measurement Error Models. Authors: Xiao-Feng Wang and Bin Wang.

degenes Detection of differentially expressed genes. Author: Klaus Jung.

digitize A plot digitizer in R. Author: Timothee Poisot.

distrEllipse S4 classes for elliptically contoured distributions. Author: Peter Ruckdeschel.

doMPI Foreach parallel adaptor for the **Rmpi** package. Author: Steve Weston. In view: *HighPerformanceComputing*.

doRedis Foreach parallel adapter for the **rredis** package. Author: B. W. Lewis. In view: *HighPerformanceComputing*.

dpmixsim Dirichlet Process Mixture model simulation for clustering and image segmentation. Author: Adelino Ferreira da Silva.

dvfBm Discrete variations of a fractional Brownian motion. Author: Jean-Francois Coeurjolly.

dynaTree Dynamic trees for learning and design. Authors: Robert B. Gramacy and Matt A. Taddy.

ebdbNet Empirical Bayes Estimation of Dynamic Bayesian Networks. Author: Andrea Rau.

edrGraphicalTools Provide tools for dimension reduction methods. Authors: Benoit Liquet and Jerome Saracco.

edtdbg Integrating R's `debug()` with Your Text Editor. Author: Norm Matloff.

egonet Tool for ego-centric measures in Social Network Analysis. Authors: A. Sciandra, F. Gioachin and L. Finos.

emoa Evolutionary Multiobjective Optimization Algorithms. Author: Olaf Mersmann.

envelope Envelope normality test. Author: Felipe Acosta.

epinet A collection of epidemic/network-related tools. Authors: Chris Groendyke, David Welch and David R. Hunter.

equate Statistical methods for test score equating. Author: Anthony Albano.

esd4all Functions for post-processing and gridding empirical-statistical downscaled climate scenarios. Author: Rasmus E. Benestad.

exactci Exact p-values and matching confidence intervals for simple discrete parametric cases. Author: M. P. Fay.

expectreg Expectile regression. Authors: Fabian Sobotka, Thomas Kneib, Sabine Schnabel and Paul Eilers.

extracat Graphic axtensions for categorical data. Author: Alexander Pilhöfer.

extremevalues Univariate outlier detection. Author: Mark van der Loo.

fCertificates Basics of certificates and structured products valuation. Author: Stefan Wilhelm.

favir Formatted actuarial vignettes in R. Author: Benedict Escoto.

fdth Frequency Distribution Tables, Histograms and Poligons. Author: Jose Claudio Faria and Enio Jelihovschi.

fftw Fast FFT and DCT based on FFTW. Author: Sebastian Krey, Uwe Ligges and Olaf Mersmann.

fisheyeR Fisheye and hyperbolic-space-alike interactive visualization tools in R. Author: Eduardo San Miguel Martin.

flexCrossHaz Flexible crossing hazards in the Cox model. Authors: Vito M.R. Muggeo and Miriam Tagliavia. In view: *Survival*.

forensim Statistical tools for the interpretation of forensic DNA mixtures. Author: Hinda Haned.

formatR Format R code automatically. Author: Yihui Xie.

formula.tools Tools for working with formulas, expressions, calls and other R objects. Author: Christopher Brown.

fptdApprox Approximation of first-passage-time densities for diffusion processes. Authors: Patricia Román-Román, Juan J. Serrano-Pérez and Francisco Torres-Ruiz.

futile.any Futile library to provide some polymorphism. Author: Brian Lee Yung Rowe.

futile.logger Futile logger subsystem. Author: Brian Lee Yung Rowe.

futile.matrix Futile matrix utilities. Author: Brian Lee Yung Rowe.

futile.options Futile options management. Author: Brian Lee Yung Rowe.

gamlss.add GAMLSS additive. Authors: Mikis Stasinopoulos and Bob Rigby.

gamlss.demo Demos for GAMLSS. Authors: Mikis Stasinopoulos, Bob Rigby, Paul Eilers and Brian Marx with contributions from Larisa Kosidou.

gamlss.util GAMLSS utilities. Authors: Mikis Stasinopoulos, Bob Rigby, Paul Eilers.

gcolor Provides an inequation algorithm function and a solution to import DIMACS files. Authors: Jeffrey L. Duffany, Ph.D. and Euripides Rivera Negron.

genefu Relevant functions for gene expression analysis, especially in breast cancer. Authors: Benjamin Haibe-Kains, Gianluca Bontempi and Christos Sotiriou.

genoPlotR Plot publication-grade gene and genome maps. Author: Lionel Guy.

genomatic Manages microsatellite projects. Creates 96-well maps, genotyping submission forms, rerun management, and import into statistical software. Author: Brian J. Knaus.

geophys Geophysics, continuum mechanics, Mogi model. Author: Jonathan M. Lees.

geosphere Spherical trigonometry. Authors: Robert J. Hijmans, Ed Williams and Chris Vennes.

grofit The package was developed to fit many growth curves obtained under different conditions. Authors: Matthias Kahm and Maik Kschischo.

haarfisz Software to perform Haar Fisz transforms. Author: Piotr Fryzlewicz.

halp Show certain type of comments from function bodies as "live-help". Author: Daniel Haase.

hda Heteroscedastic Discriminant Analysis. Author: Gero Szepannek.

heavy Estimation in the linear mixed model using heavy-tailed distributions. Author: Felipe Osorio.

hergm Hierarchical Exponential-family Random Graph Models. Author: Michael Schweinberger.

hgam High-dimensional Additive Modelling. Authors: The students of the "Advanced R Programming Course" Hannah Frick, Ivan Kondofersky, Oliver S. Kuehnle, Christian Lindenlaub, Georg Pfundstein, Matthias Speidel, Martin Spindler, Ariane Straub, Florian Wickler and Katharina Zink, under the supervision of Manuel Eugster and Torsten Hothorn.

hglm Hierarchical Generalized Linear Models. Authors: Moudud Alam, Lars Ronnegard, Xia Shen.

highlight Syntax highlighter. Author: Romain François.

histogram Construction of regular and irregular histograms with different options for automatic choice of bins. Authors: Thoralf Mildenberger, Yves Rozenholc and David Zasada.

hts Hierarchical time series. Authors: Rob J Hyndman, Roman A Ahmed, and Han Lin Shang.

hyperSpec Interface for hyperspectral data sets, i.e., spectra + meta information (spatial, time, concentration, ...). Author: Claudia Beleites.

iBUGS An interface to **R2WinBUGS** by **gWidgets**. Author: Yihui Xie.

iCluster Integrative clustering of multiple genomic data types. Author: Ronglai Shen.

ibr Iterative Bias Reduction. Authors: Pierre-Andre Cornillon, Nicolas Hengartner and Eric Matzner-Lober.

ipw Estimate inverse probability weights. Author: Willem M. van der Wal. In view: *Survival*.

isopam Isopam (hierarchical clustering). Author: Sebastian Schmidtlein.

itertools Iterator tools. Authors: Steve Weston and Hadley Wickham.

kinfit Routines for fitting kinetic models to chemical degradation data. Author: Johannes Ranke.

kml3d K-means for joint longitudinal data. Author: Christophe Genolini.

lavaan Latent Variable Analysis. Author: Yves Rosseel. In view: *Psychometrics*.

lcmm Estimation of latent class mixed models. Authors: Cecile Proust-Lima and Benoit Liquet.

leiv Bivariate linear errors-in-variables estimation. Author: David Leonard.

limitplot Jitter/Box Plot with Ordered Points Below the Limit of Detection. Author: Omar E. Olmedo.

lmPerm Permutation tests for linear models. Author: Bob Wheeler.

log10 Decimal log plotting in two and three dimensions. Author: Timothée Poisot.

logging A tentative logging package. Author: Mario Frasca.

lossDev Robust loss development using MCMC. Authors: Christopher W. Laws and Frank A. Schmid.

lqa Penalized likelihood inference for GLMs. Author: Jan Ulbricht.

magnets Simulate micro-magnets dynamics. Author: Hai Qian.

mbmdr Model Based Multifactor Dimensionality Reduction. Authors: Victor Urrea, Malu Calle, Kristel Van Steen and Nuria Malats.

mclogit Mixed Conditional Logit. Author: Martin Elff.

mecdf Multivariate ECDF-Based Models. Author: Charlotte Maia. In view: *Distributions*.

micEconAids Demand analysis with the Almost Ideal Demand System (AIDS). Author: Arne Henningsen. In view: *Econometrics*.

micEconCES Analysis with the Constant Elasticity of Scale (CES) function. Authors: Arne Henningsen and Geraldine Henningsen. In view: *Econometrics*.

micEconSNQP Symmetric Normalized Quadratic Profit Function. Author: Arne Henningsen. In view: *Econometrics*.

migui Graphical User Interface of the **mi** Package. Authors: Daniel Lee and Yu-Sung Su.

miniGUI tktcl quick and simple function GUI. Author: Jorge Luis Ojeda Cabrera.

minqa Derivative-free optimization algorithms by quadratic approximation. Authors: Douglas Bates, Katharine M. Mullen, John C. Nash and Ravi Varadhan.

miscTools Miscellanneous Tools and Utilities. Author: Arne Henningsen.

missMDA Handling missing values with/in multivariate data analysis (principal component methods). Authors: François Husson and Julie Josse.

mixOmics Integrate omics data project. Authors: Sébastien Dejean, Ignacio González and Kim-Anh Lê Cao.

mkin Routines for fitting kinetic models with one or more state variables to chemical degradation data. Author: Johannes Ranke.

mlogitBMA Bayesian Model Averaging for multinomial logit models. Authors: Hana Ševčíková and Adrian Raftery.

mmap Map pages of memory. Author: Jeffrey A. Ryan.

monmlp Monotone multi-layer perceptron neural network. Author: Alex J. Cannon.

mrdrc Model-robust concentration-response analysis. Authors: Christian Ritz and Mads Jeppe Tarp-Johansen.

mseq Modeling non-uniformity in short-read rates in RNA-Seq data. Author: Jun Li.

mspath Multi-state Path-dependent models in discrete time. Authors: Ross Boylan and Peter Bacchetti, building on the work of Christopher H. Jackson. Thorsten Ottosen for the `ptr_container` library. Gennadiy Rozental for the Boost Test Library. And the authors of other Boost libraries used by the previous two. In view: *Survival*.

mtsc Multivariate timeseries cluster-based models. Author: Charlotte Maia.

mugnet Mixture of Gaussian Bayesian network model. Author: Nikolay Balov.

multisensi Multivariate Sensitivity Analysis. Authors: Matieyendou Lamboni and Hervé Monod.

munfold Metric Unfolding. Author: Martin Elff.

mutatr Mutable objects for R. Author: Hadley Wickham.

mvabund Statistical methods for analysing multivariate abundance data. Authors: Ulrike Naumann, Yi Wang, Stephen Wright and David Warton.

mvngGrAd Software for moving grid adjustment in plant breeding field trials. Author: Frank Technow.

nanop Tools for nanoparticle simulation and PDF calculation. Author: Katharine Mullen.

ncdf4 Interface to Unidata netCDF (version 4 or earlier) format data files. Author: David Pierce.

ncvreg Regularization paths for SCAD- and MCP-penalized regression models. Author: Patrick Breheny.

neldermead R port of the Scilab neldermead module. Authors: Sebastien Bihorel and Michael Baudin (author of the original module).

nga NGA earthquake ground motion prediction equations. Authors: James Kaklamanos and Eric M. Thompson.

nlADG Regression in the Normal Linear ADG Model. Authors: Lutz F. Gruber

nnDiag k-Nearest Neighbor diagnostic tools. Authors: Brian Walters, Andrew O. Finley and Zhen Zhang.

nonparaeff Nonparametric methods for measuring efficiency and productivity. Author: Donghyun Oh.

nonrandom Stratification and matching by the propensity score. Author: Susanne Stampf.

npRmpi Parallel nonparametric kernel smoothing methods for mixed data types. Authors: Tristen Hayfield and Jeffrey S. Racine.

nppbib Nonparametric Partially-Balanced Incomplete Block Design Analysis. Authors: David Allingham and D.J. Best.

nutshell Data for "R in a Nutshell". Author: Joseph Adler.

nytR Provides access to the NY Times API. Author: Shane Conway.

ofp Object-Functional Programming. Author: Charlotte Maia.

omd Filter the molecular descriptors for QSAR. Author: Bin Ma.

openintro Open Intro data sets and supplement functions. Authors: David M Diez and Christopher D Barr.

optimbase R port of the Scilab optimbase module. Authors: Sebastien Bihorel and Michael Baudin (author of the original module).

optimsimplex R port of the Scilab optimsimplex module. Authors: Sebastien Bihorel and Michael Baudin (author of the original module).

optparse Command line option parser. Authors: Trevor Davis. `getopt` function, some documentation and unit tests ported from Allen Day's **getopt** package. Based on the optparse Python library by the Python Software Foundation.

optpart Optimal partitioning of similarity relations. Author: David W. Roberts.

ordinal Regression models for ordinal data. Author: Rune Haubo B Christensen.

oro.dicom Rigorous — DICOM Input / Output. Author: Brandon Whitcher. In view: *MedicalImaging*.

oro.nifti Rigorous — NIfTI Input / Output. Authors: Brandon Whitcher, Volker Schmid and Andrew Thornton. In view: *MedicalImaging*.

pGLS Generalized Least Square in comparative Phylogenetics. Authors: Xianyun Mao and Timothy Ryan.

pROC Display and analyze ROC curves. Authors: Xavier Robin, Natacha Turck, Alexandre Hainard, Natalia Tiberti, Frédérique Lisacek, Jean-Charles Sanchez and Markus Müller.

paleoMAS Paleoecological Analysis. Authors: Alexander Correa-Metrio, Dunia Urrego, Kenneth Cabrera, Mark Bush.

pamctdp Principal Axes Methods for Contingency Tables with Partition Structures on Rows and Columns. Author: Campo Elías Pardo.

partitionMetric Compute a distance metric between two partitions of a set. Authors: David Weisman and Dan Simovici.

parviol Parviol combines parallel coordinates and violin plot. Author: Jaroslav Myslivec.

pedantics Functions to facilitate power and sensitivity analyses for genetic studies of natrual popualtions. Author: Michael Morrissey.

pgfSweave Quality speedy graphics compilation with Sweave. Authors: Cameron Bracken and Charlie Sharpsteen.

phyclust Phylogenetic Clustering (Phyloclustering). Author: Wei-Chen Chen. In view: *Phylogenetics*.

phylobase Base package for phylogenetic structures and comparative data. Authors: R Hackathon et al. (alphabetically: Ben Bolker, Marguerite Butler, Peter Cowan, Damien de Vienne, Thibaut Jombart, Steve Kembel, François Michonneau, David Orme, Brian O'Meara, Emmanuel Paradis, Jim Regetz, Derrick Zwickl). In view: *Phylogenetics*.

phyloclim Integrating phylogenetics and climatic niche modelling. Author: Christoph Heibl. In view: *Phylogenetics*.

plgp Particle Learning of Gaussian Processes. Author: Robert B. Gramacy.

plsdof Degrees of freedom and confidence intervals for Partial Least Squares regression. Authors: Nicole Krämer and Mikio L. Braun.

pmlr Penalized Multinomial Logistic Regression. Authors: Sarah Colby, Sophia Lee, Juan Pablo Lewinger and Shelley Bull.

png Read and write PNG images. Author: Simon Urbanek.

poistweedie Poisson-Tweedie exponential family models. Authors: David Pechel Cactcha, Laure Pauline Fotso and Célestin C Kokonendji.

polySegratio Simulate and test marker dosage for dominant markers in autopolyploids. Author: Peter Baker.

polySegratioMM Bayesian mixture models for marker dosage in autopolyploids. Author: Peter Baker.

popPK Summary of population pharmacokinetic analysis. Author: Christoffer W. Tornoe and Fang Li.

powerMediation Power/sample size calculation for mediation analysis. Author: Weiliang Qiu.

ppMeasures Point pattern distances and prototypes. Authors: David M Diez, Katherine E Tranbarger Freier, and Frederic P Schoenberg.

profdpm Profile Dirichlet Process Mixtures. Author: Matt Shotwell.

psychotree Recursive partitioning based on psychometric models. Authors: Achim Zeileis, Carolin Strobl and Florian Wickelmaier. In view: *Psychometrics*.

ptw Parametric Time Warping. Authors: Jan Gerretzen, Paul Eilers, Hans Wouters, Tom Bloemberg and Ron Wehrens. In view: *TimeSeries*.

pyramid Functions to draw population pyramid. Author: Minato Nakazawa.

qrnn Quantile regression neural network. Author: Alex J. Cannon.

quaternions Arithmetics and linear algebra with Quaternions. Author: K. Gerald van den Boogaart.

r2dRue 2dRue: 2d Rain Use Efficience model. Authors: Gabriel del Barrio, Juan Puigdefabregas, Maria E. Sanjuan and Alberto Ruiz.

r2lh R to LaTeX and HTML. Authors: Christophe Genolini, Bernard Desgraupes, Lionel Riou Franca.

r4ss R code for Stock Synthesis. Authors: Ian Taylor, Ian Stewart, Tommy Garrison, Andre Punt, John Wallace, and other contributors.

rEMM Extensible Markov Model (EMM) for data stream clustering in R. Authors: Michael Hahsler and Margaret H. Dunham.

random.polychor.pa A parallel analysis with polychoric correlation matrices. Author: Fabio Presaghi and Marta Desimoni.

rangeMapper A package for easy generation of biodiversity (species richness) or life-history traits maps using species geographical ranges. Authors: Mihai Valcu and James Dale.

raster Geographic analysis and modeling with raster data. Author: Robert J. Hijmans and Jacob van Etten.

recommenderlab Lab for testing and developing recommender algorithms for binary data. Author: Michael Hahsler.

remix Remix your data. Author: David Hajage.

rgp R genetic programming framework. Authors: Oliver Flasch, Olaf Mersmann, Thomas Bartz-Beielstein and Joerg Stork.

ris RIS package for bibliographic analysis. Author: Stephanie Kovalchik.

rngWELL toolbox for WELL random number generators. Authors: C code by F. Panneton, P. L'Ecuyer and M. Matsumoto; R port by Christophe Dutang and Petr Savicky.

rocc ROC based classification. Author: Martin Lauss.

rpsychi Statistics for psychiatric research. Author: Yasuyuki Okumura.

rredis Redis client for R. Author: B. W. Lewis.

rrules Generic rule engine for R. Authors: Oliver Flasch, Olaf Mersmann, Thomas Bartz-Beielstein.

rworldmap For mapping global data: rworldmap. Authors: Andy South, with contributions from Barry Rowlingson, Roger Bivand, Matthew Staines and Pru Foster. In view: *Spatial*.

samplesize A collection of sample size functions. Author: Ralph Scherer.

samplingbook Survey sampling procedures. Authors: Juliane Manitz, contributions by Mark Hempelmann, Göran Kauermann, Helmut Küchenhoff, Cornelia Oberhauser, Nina Westerheide, Manuel Wiesenfarth.

saws Small-Sample Adjustments for Wald tests Using Sandwich Estimators. Author: Michael Fay.

scaRabee Optimization toolkit for pharmacokinetic-pharmacodynamic models. Author: Sebastien Bihorel.

scrapeR Tools for scraping data from HTML and XML documents. Author: Ryan M. Acton.

secr Spatially explicit capture-recapture. Author: Murray Efford.

semPLS Structural Equation Modeling using Partial Least Squares. Author: Armin Monecke.

sensitivityPStrat Principal stratification sensitivity analysis functions. Author: Charles Dupont.

sifds Swedish inflation forecast data set. Author: Michael Lundholm.

simPopulation Simulation of synthetic populations for surveys based on sample data. Authors: Stefan Kraft and Andreas Alfons.

sisus SISUS: Stable Isotope Sourcing using Sampling. Author: Erik Barry Erhardt.

smd.and.more t-test with Graph of Standardized Mean Difference, and More. Authors: David W. Gerbing, School of Business Administration, Portland State University.

solaR Solar photovoltaic aystems. Author: Oscar Perpiñán Lamigueiro.

someKfwer Controlling the Generalized Familywise Error Rate. Authors: L. Finos and A. Farcomeni.

spaa Species Association Analysis. Author: Jinlong Zhang Qiong Ding Jihong Huang.

space Sparse PArtial Correlation Estimation. Authors: Jie Peng , Pei Wang, Nengfeng Zhou and Ji Zhu.

sparcl Perform sparse hierarchical clustering and sparse k-means clustering. Authors: Daniela M. Witten and Robert Tibshirani.

sparkTable Sparklines and graphical tables for TEXand HTML. Authors: Alexander Kowarik, Bernhard Meindl and Stefan Zechner.

sparr The sparr package: SPAtial Relative Risk. Authors: T.M. Davies, M.L. Hazelton and J.C. Marshall.

spef Semiparametric estimating functions. Authors: Xiaojing Wang and Jun Yan.

sphet Spatial models with heteroskedastic innovations. Author: Gianfranco Piras.

spikeslab Prediction and variable selection using spike and slab regression. Author: Hemant Ishwaran.

stam Spatio-Temporal Analysis and Modelling. Author: Zhijie Zhang.

steepness Testing steepness of dominance hierarchies. Author: David Leiva and Han de Vries.

stockPortfolio Build stock models and analyze stock portfolios. Authors: David Diez and Nicolas Christou. In view: *Finance*.

stoichcalc R functions for solving stoichiometric equations. Author: Peter Reichert.

stratification Univariate stratification of survey populations. Authors: Sophie Baillargeon and Louis-Paul Rivest.

stratigraph Toolkit for the plotting and analysis of stratigraphic and palaeontological data. Author: Walton A. Green.

stringr Make it easier to work with strings. Author: Hadley Wickham.

survPresmooth Presmoothed estimation in survival analysis. Authors: Ignacio Lopez-de-Ullibarri and Maria Amalia Jacome.

symbols Symbol plots. Author: Jaroslav Myslivec.

symmoments Symbolic central moments of the multivariate normal distribution. Author: Kem Phillips.

tclust Robust trimmed clustering. Authors: Agustín Mayo Iscar, Luis Angel García Escudero and Heinrich Fritz.

testthat Testthat code. Tools to make testing fun :). Author: Hadley Wickham.

tgram Functions to compute and plot tracheidograms. Authors: Marcelino de la Cruz and Lucia DeSoto.

topicmodels Topic models. Authors: Bettina Grün and Kurt Hornik.

tourr Implement tour methods in pure R code. Authors: Dianne Cook and Hadley Wickham.

tourrGui A Tour GUI using **gWidgets**. Authors: Bei, Dianne Cook, and Hadley Wickham.

traitr An interface for creating GUIs modeled in part after traits UI module for Python. Author: John Verzani.

trex Truncated exact test for two-stage case-control design for studying rare genetic variants. Authors: Schaid DJ and Sinnwell JP.

trio Detection of disease-associated SNP interactions in case-parent trio data. Authors: Qing Li and Holger Schwender.

tsne T-distributed Stochastic Neighbor Embedding for R (t-SNE). Author: Justin Donaldson.

ttime Translate neurodevelopmental event timing across species. Author: Radhakrishnan Nagarajan.

twiddler Interactive manipulation of R expressions. Authors: Oliver Flasch and Olaf Mersmann.

twopartqtl QTL Mapping for Point-mass Mixtures. Author: Sandra L. Taylor.

umlr Open Source Development with UML and R. Author: Charlotte Maia.

unmarked Models for Data from Unmarked Animals. Authors: Ian Fiske and Richard Chandler.

vcdExtra vcd additions. Authors: Michael Friendly with Heather Turner, David Firth, Achim Zeileis and Duncan Murdoch.

vegdata Functions to use vegetation databases (Turboveg) for vegetation analyses in R. Author: Florian Jansen.

venneuler Venn and Euler Diagrams. Author: Lee Wilkinson.

vmv Visualization of Missing Values. Author: Waqas Ahmed Malik.

waterfall Waterfall Charts in R. Authors: James P. Howard, II.

waveband Computes credible intervals for Bayesian wavelet shrinkage. Author: Stuart Barber.

webvis Create graphics for the web from R. Author: Shane Conway.

wq Exploring water quality monitoring data. Authors: Alan D. Jassby and James E. Cloern.

xlsx Read, write, format Excel 2007 ('xlsx') files. Author: Adrian A. Dragulescu.

xlsxjars Package required jars for the **xlsx** package. Author: Adrian A. Dragulescu.

zic Bayesian Inference for Zero-Inflated Count Models. Author: Markus Jochmann. In view: *Bayesian*.

Other changes

- The following packages were moved to the Archive: **CTFS**, **ComPairWise**, **DICOM**, **GFMaps**, **HFWutils**, **IQCC**, **JointGLM**, **MIfuns**, **RGrace**, **RII**, **SimHap**, **SoPhy**, **adapt**, **agce**, **agsemisc**, **assist**, **cir**, **connectedness**, **dirichlet**, **empiricalBayes**, **gcl**, **gtm**, **latentnetHRT**, **markerSelectionPower**, **mcgibbsit**, **networksis**, **nlts**, **popgen**, **poplab**, **qdg**, **r2lUniv**, **rankreg**, **spatclus**, **staRt**, **sublogo**, **tdist**, **tutoR**, **vabayelMix**, and **varmixt**.

- The following packages were resurrected from the Archive: **ProbForecastGOP**, **SciViews**, **apTreeshape**, **far**, **norm**, **vardiag** and **xlsReadWrite**.

- Package **atm** was renamed to **amba**.

Kurt Hornik
WU Wirtschaftsuniversität Wien, Austria
`Kurt.Hornik@R-project.org`

Achim Zeileis
Universität Innsbruck, Austria
`Achim.Zeileis@R-project.org`

News from the Bioconductor Project

Bioconductor Team
Program in Computational Biology
Fred Hutchinson Cancer Research Center

We are pleased to announce Bioconductor 2.6, released on April 23, 2010. Bioconductor 1.6 is compatible with R 2.11.1, and consists of 389 packages. There are 37 new packages, and enhancements to many others. Explore Bioconductor at http://bioconductor.org, and install packages with

```
source("http://bioconductor.org/biocLite.R")
biocLite() # install standard packages...
biocLite("IRanges") # ...or IRanges
```

New and revised packages

Sequence analysis packages address infrastructure (**GenomicRanges**, **Rsamtools**, **girafe**); ChIP-seq (**BayesPeak**, **CSAR**, **PICS**); digital gene expression and RNA-seq (**DESeq**, **goseq**, **segmentSeq**); and motif discovery (**MotIV**, **rGADEM**).

Microarray analysis packages introduce new approaches to pre-process and technology-specific assays (**affyILM**, **frma**, **frmaTools**, **BeadDataPackR**, **MassArray**); analysis of specific experimental protocols (**charm**, **genoCN**, **iChip**, **methVisual**); and novel statistical methods (**ConsensusClusterPlus**, **ExpressionView**, **eisa**, **GSRI**, **PROMISE**, **tigre**).

Flow cytometry packages include **SamSPECTRAL**, **flowMeans**, **flowTrans**, and **iFlow**.

Annotation and integrative analysis packages facilitate interfacing with GEO (**GEOsubmission**), the Sequence Read Archive (**SRAdb**), and tabulation of genome sequence project data (**genomes**); the **GSRI** package to estimate differentially expressed genes in a gene set; PCA and CCA dependency modeling (**pint**); and updated access to exon array annotations (**xmapcore**).

Further information on new and existing packages can be found on the Bioconductor web site, which contains 'views' that identify coherent groups of packages. The views link to on-line package descriptions, vignettes, reference manuals, and use statistics.

Other activities

Training courses and a European developer conference were important aspects of Bioconductor, with recent successful offerings ranging from microarray and sequence analysis through advanced R programming (http://bioconductor.org/workshops). The active Bioconductor mailing lists (http://bioconductor.org/docs/mailList.html) connect users with each other, to domain experts, and to maintainers eager to ensure that their packages satisfy the needs of leading edge approaches. The Bioconductor team continues to ensure quality software through technical and scientific reviews of new packages, and daily builds of released packages on Linux, Windows (32- and now 64-bit), and Macintosh platforms.

Looking forward

BioC2010 is the Bioconductor annual conference, scheduled for July 29–30 at the Fred Hutchinson Cancer Research Centre in Seattle, Washington, and with a developer day on July 28. The conference features morning scientific talks and afternoon practical sessions (https://secure.bioconductor.org/BioC2010/). European developers will have an opportunity to meet in Heidelberg, Germany, on 17–18 November.

Contributions from the Bioconductor community play an important role in shaping each release. We anticipate continued efforts to provide statistically informed analysis of next generation sequence data, as well as development of algorithms for advanced analysis of microarrays. Areas of opportunity include the ChIP-seq, RNA-seq, rare variant, and structural variant domains. The next release cycle promises to be one of active scientific growth and exploration!

R Foundation News

by Kurt Hornik

Donations and new members

Donations

Vies Animales, France

Owen Jones, Robert Maillardet, and Andrew Robinson, Australia: donating a proportion of the royalties for the book "Introduction to Scientific Programming and Simulation using R", recently published (2009) by Chapman & Hall/CRC

Jogat Sheth, USA

New benefactors

OpenAnalytics, Belgium

Ritter and Danielson Consulting, Belgium

Saorstat Limited, Ireland

New supporting institutions

Marine Scotland Science (UK)

DFG Research Group "Mind and Brain Dynamics", Universität Potsdam (Germany)

Chaire d'actuariat, Université Laval (Canada)

New supporting members

José Banda, USA

Kapo Martin Coulibaly, USA

Andreas Eckner, USA

Marco Geraci, UK

David Monterde, Spain

Guido Möser, Germany

Alexandre Salvador, France

Ajit de Silva, USA

Wen Hsiang Wie, Taiwan

Tomas Zelinsky, Slovakia

Kurt Hornik
WU Wirtschaftsuniversität Wien, Austria
`Kurt.Hornik@R-project.org`