BERTIN MATRICES

AN  IMPLEMENTATION

GÜNTHER SAWITZKI
STATLAB HEIDELBERG

Contents

1. Bertin Plots

Among the rich material on graphical presentation of information, in (Bertin, 1977) (engl. (Bertin, 1999)) J. Bertin discusses the presentation of data matrices, with a particular view to seriation. (de Falguerolles et al., 1997) gives an appraisal of this aspect of Bertin's work. The methods discussed in (de Falguerolles et al., 1997) have

Date: Aug. 2010.
Revised: August 2011
Typeset, with minor revisions: September 24, 2011 from SVN *Revision* : 46.
Key words and phrases. statistical computing, S programming language, R programming, data analysis, exploratory statistics, residual diagnostics, R language.
URL: `http://bertin.r-forge.r-project.org/`.

been implemented in the Voyager system (Sawitzki, 1996). They have been partially re-implemented in R, and this paper gives an introduction to the R-implementation.

The R-implementation can be downloaded as a package *bertin* from `http://bertin.r-forge.r-project.org/`. (de Falguerolles et al., 1997) is included as bertin.pdf in the documentation section of the package.

Bertin uses a small data set on hotel occupancy data to illustrate his ideas.

	Jan	Fev	Mars	Avril	May	Juin	Juil	Aout	Sept	Oct	Nov	Dec
ClienteleFeminine	26	21	26	28	20	20	20	20	20	40	15	40
Locale	69	70	77	71	37	36	39	39	55	60	68	72
USA	7	6	3	6	23	14	19	14	9	6	8	8
AmerSud	0	0	0	0	8	6	6	4	2	12	0	0
Europe	20	15	14	15	23	27	22	30	27	19	19	17
MOrientAfrique	1	0	0	8	6	4	6	4	2	1	0	1
Asie	3	10	6	0	3	13	8	9	5	2	5	2
Business	78	80	85	86	85	87	70	76	87	85	87	80
Touristes	22	20	15	14	15	13	30	24	13	15	13	20
ResDirecte	70	70	78	74	69	68	74	75	68	68	64	75
ResAgents	20	18	19	17	27	27	19	19	26	27	21	15
EquipageAeriens	10	12	6	9	4	5	7	6	6	5	15	10
MoinsDe20	2	2	4	2	2	1	1	2	2	4	2	5
De20a55	25	27	37	35	25	25	27	28	24	30	24	30
De35a55	48	49	42	48	54	55	53	51	55	46	55	43
PlusDe55	25	22	17	15	19	19	19	19	19	20	19	22
Prix	163	167	166	174	152	155	145	170	157	174	165	156
Duree	1.65	1.71	1.65	1.91	1.9	2	1.54	1.6	1.73	1.82	1.66	1.44
Occupation	67	82	70	83	74	77	56	62	90	92	78	55
Foires	0	0	0	1	1	1	0	0	1	1	1	1

Table 1: Bertin's hotel data

2. Bertin Matrices

To repeat from (de Falguerolles et al., 1997): In abstract terms, a Bertin matrix is a matrix of displays. Bertin matrices allow rearrangements to transform an initial matrix to a more homogeneous structure. The rearrangements are row or column permutations, and groupings of rows or columns. To fix ideas, think of a data matrix, variable by case, with real valued variables. For each variable, draw a bar chart of variable value by case. Highlight all bars representing a value above some sample threshold for that variable (Figure 1 on page 3).

Variables are collected in a matrix to display the complete data set (Figure 2 on page 3). By convention, Bertin shows variables in rows and cases in columns. To make periodic structures more visible, the data are repeated cyclically.

As Bertin points out, the indexing used is arbitrary. You can rearrange rows and/or columns to reveal the information of interest. If you run a hotel, of course the percentage of hotel occupation and the duration of the visits are most interesting for you. Move these variables to the top of the display, and rearrange the other variables by similarity or dissimilarity to these target variables (Figure 3 on page 4). Time points have a natural order. No rearrangement is used here.

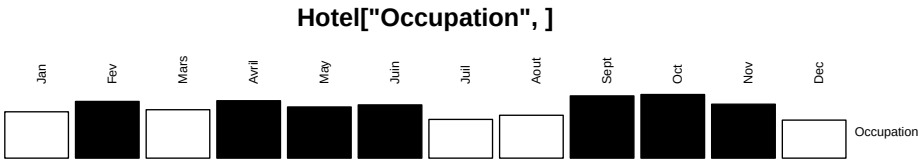


Figure 1: Display of one variable: **Hotel data**. Observations above average are highlighted.

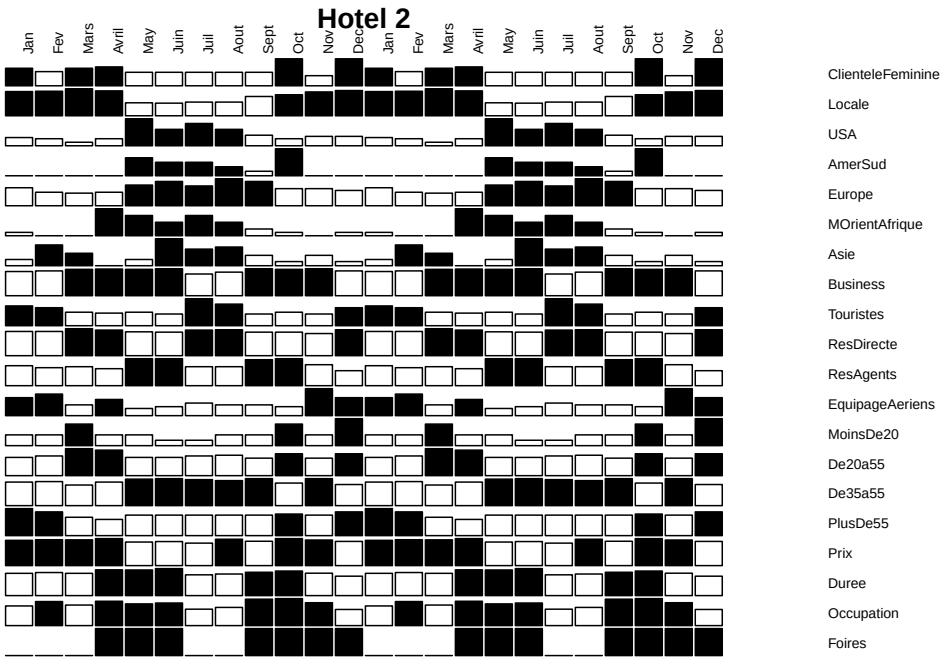


Figure 2: Display of a data matrix: **Hotel data**. Variables are shown as rows. To make periodic structures more visible, time is duplicated. Observations above average are highlighted.

As a second example, we use the the *`USJudgeRatings`* data set (Figure 4 on page 6). The data is listed in Table 2 on page 5.

Both cases (the judges) and variables (the qualities) allow for a rearrangement. Just sorting for row and column averages gives a more informative picture (Figure 5 on

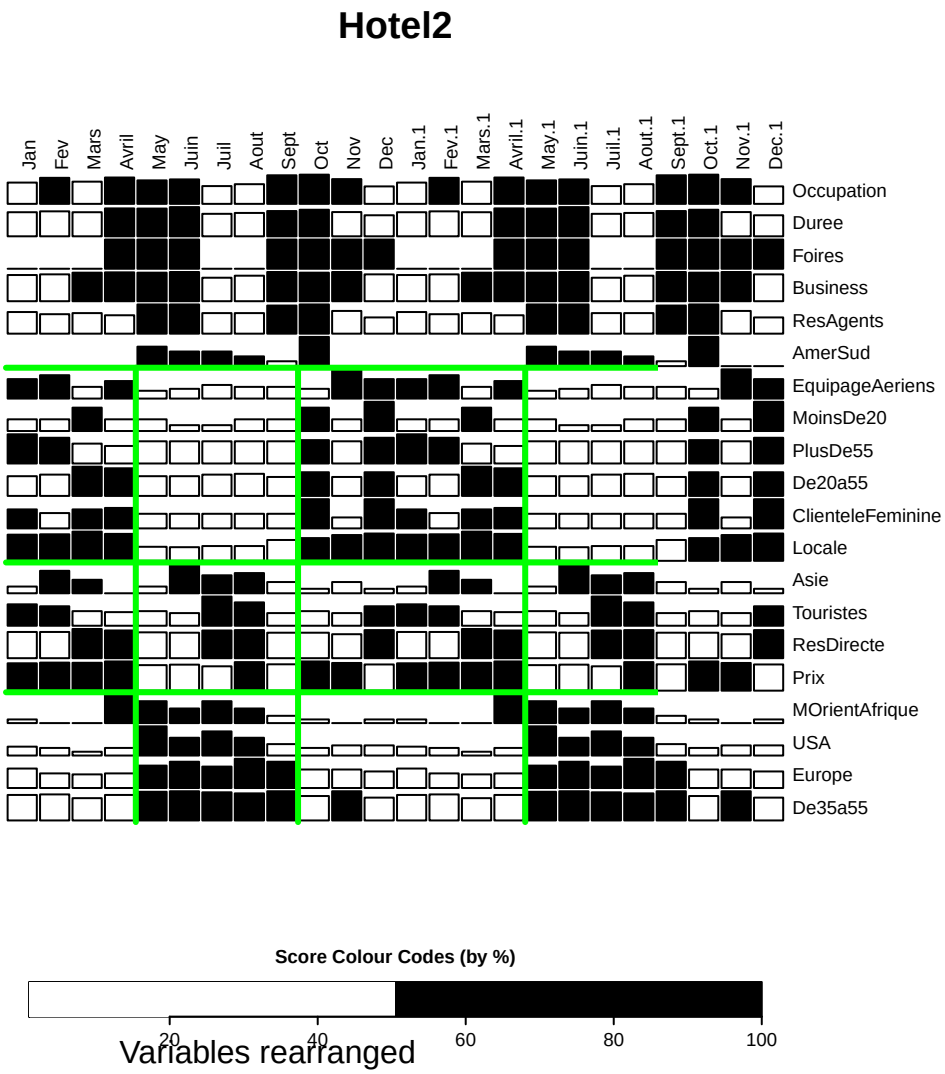


Figure 3: Display of a data matrix: Hotel data. Variables are rearranged by similarity to occupation and duration.

page 6). The number of contacts stands out - it has a different structure than the other variables. After all, this is not a rating variable at all, but ancillary information. There is little reason why it should go along with the rating variables. Judge G. A. Saden seems to be special. Most variables would rank him to the upper group, be his worth of retention is below average. The esteem of his integrity and demeanour go along with this. Overall, there is a very clear separation into an upper and a lower group.

We remove the variable *CONT* and re-run the analysis. Of course this changes the average scores per judge, and the arrangement changes (Figure 6 on page 7).

At this early point, let us put Bertin's work in place. Visualising information is but one aspect. In statistics, as we see it today, visualisation may be one part of an analysis. The outcome will be a decision leading to an action. Then there is a loss

	CONT	INTG	DMNR	DILG	CFMG	DECI	PREP	FAMI	ORAL	WRIT	PHYS	RTEN
AARONSON,L.H.	5.70	7.90	7.70	7.30	7.10	7.40	7.10	7.10	7.10	7.00	8.30	7.80
ALEXANDER,J.M.	6.80	8.90	8.80	8.50	7.80	8.10	8.00	8.00	7.80	7.90	8.50	8.70
ARMENTANO,A.J.	7.20	8.10	7.80	7.80	7.50	7.60	7.50	7.50	7.30	7.40	7.90	7.80
BERDON,R.I.	6.80	8.80	8.50	8.80	8.30	8.50	8.70	8.70	8.40	8.50	8.80	8.70
BRACKEN,J.J.	7.30	6.40	4.30	6.50	6.00	6.20	5.70	5.70	5.10	5.30	5.50	4.80
BURNS,E.B.	6.20	8.80	8.70	8.50	7.90	8.00	8.10	8.00	8.00	8.00	8.60	8.60
CALLAHAN,R.J.	10.60	9.00	8.90	8.70	8.50	8.50	8.50	8.50	8.60	8.40	9.10	9.00
COHEN,S.S.	7.00	5.90	4.90	5.10	5.40	5.90	4.80	5.10	4.70	4.90	6.80	5.00
DALY,J.J.	7.30	8.90	8.90	8.70	8.60	8.50	8.40	8.40	8.40	8.50	8.80	8.80
DANNEHY,J.F.	8.20	7.90	6.70	8.10	7.90	8.00	7.90	8.10	7.70	7.80	8.50	7.90
DEAN,H.H.	7.00	8.00	7.60	7.40	7.30	7.50	7.10	7.20	7.10	7.20	8.40	7.70
DEVITA,H.J.	6.50	8.00	7.60	7.20	7.00	7.10	6.90	7.00	7.00	7.10	6.90	7.20
DRISCOLL,P.J.	6.70	8.60	8.20	6.80	6.90	6.60	7.10	7.30	7.20	7.20	8.10	7.70
GRILLO,A.E.	7.00	7.50	6.40	6.80	6.50	7.00	6.60	6.80	6.30	6.60	6.20	6.50
HADDEN,W.L.JR.	6.50	8.10	8.00	8.00	7.90	8.00	7.90	7.80	7.80	7.80	8.40	8.00
HAMILL,E.C.	7.30	8.00	7.40	7.70	7.30	7.30	7.30	7.20	7.10	7.20	8.00	7.60
HEALEY.A.H.	8.00	7.60	6.60	7.20	6.50	6.50	6.80	6.70	6.40	6.50	6.90	6.70
HULL,T.C.	7.70	7.70	6.70	7.50	7.40	7.50	7.10	7.30	7.10	7.30	8.10	7.40
LEVINE,I.	8.30	8.20	7.40	7.80	7.70	7.70	7.70	7.80	7.50	7.60	8.00	8.00
LEVISTER,R.L.	9.60	6.90	5.70	6.60	6.90	6.60	6.20	6.00	5.80	5.80	7.20	6.00
MARTIN,L.F.	7.10	8.20	7.70	7.10	6.60	6.60	6.70	6.70	6.80	6.80	7.50	7.30
MCGRATH,J.F.	7.60	7.30	6.90	6.80	6.70	6.80	6.40	6.30	6.30	6.30	7.40	6.60
MIGNONE,A.F.	6.60	7.40	6.20	6.20	5.40	5.70	5.80	5.90	5.20	5.80	4.70	5.20
MISSAL,H.M.	6.20	8.30	8.10	7.70	7.40	7.30	7.30	7.30	7.20	7.30	7.80	7.60
MULVEY,H.M.	7.50	8.70	8.50	8.60	8.50	8.40	8.50	8.50	8.40	8.40	8.70	8.70
NARUK,H.J.	7.80	8.90	8.70	8.90	8.70	8.80	8.90	9.00	8.80	8.90	9.00	9.00
O'BRIEN,F.J.	7.10	8.50	8.30	8.00	7.90	7.90	7.80	7.80	7.80	7.70	8.30	8.20
O'SULLIVAN,T.J.	7.50	9.00	8.90	8.70	8.40	8.50	8.40	8.30	8.30	8.30	8.80	8.70
PASKEY,L.	7.50	8.10	7.70	8.20	8.00	8.10	8.20	8.40	8.00	8.10	8.40	8.10
RUBINOW,J.E.	7.10	9.20	9.00	9.00	8.40	8.60	9.10	9.10	8.90	9.00	8.90	9.20
SADEN.G.A.	6.60	7.40	6.90	8.40	8.00	7.90	8.20	8.40	7.70	7.90	8.40	7.50
SATANIELLO,A.G.	8.40	8.00	7.90	7.90	7.80	7.80	7.60	7.40	7.40	7.40	8.10	7.90
SHEA,D.M.	6.90	8.50	7.80	8.50	8.10	8.20	8.40	8.50	8.10	8.30	8.70	8.30
SHEA,J.F.JR.	7.30	8.90	8.80	8.70	8.40	8.50	8.50	8.50	8.40	8.40	8.80	8.80
SIDOR,W.J.	7.70	6.20	5.10	5.60	5.60	5.90	5.60	5.60	5.30	5.50	6.30	5.30
SPEZIALE,J.A.	8.50	8.30	8.10	8.30	8.40	8.20	8.20	8.10	7.90	8.00	8.00	8.20
SPONZO,M.J.	6.90	8.30	8.00	8.10	7.90	7.90	7.90	7.70	7.60	7.70	8.10	8.00
STAPLETON,J.F.	6.50	8.20	7.70	7.80	7.60	7.70	7.70	7.70	7.50	7.60	8.50	7.70
TESTO,R.J.	8.30	7.30	7.00	6.80	7.00	7.10	6.70	6.70	6.70	6.70	8.00	7.00
TIERNEY,W.L.JR.	8.30	8.20	7.80	8.30	8.40	8.30	7.70	7.60	7.50	7.70	8.10	7.90
WALL,R.A.	9.00	7.00	5.90	7.00	7.00	7.20	6.90	6.90	6.50	6.60	7.60	6.60
WRIGHT,D.B.	7.10	8.40	8.40	7.70	7.50	7.70	7.80	8.20	8.00	8.10	8.30	8.10
ZARRILLI,K.J.	8.60	7.40	7.00	7.50	7.50	7.70	7.40	7.20	6.90	7.00	7.80	7.10

Table 2: US judge ratings

(or gain) depending on the action taken on the one hand, and the "true" state of the world on the other. This is the common decision theoretical setting. Statistics has formulated a few standard problems, and given suggestions how to handle these. In our Hotel example, the problem can be seen as a prediction problem: find a prediction model to predict occupation and duration, based on the other variables. More specifically this is a control problem, and the statistical contribution is to find a regression model for occupation and duration, based on the other variables. The visualisation can be seen as one way to hint at a regression model. There are very few classical problems. Regression is one of them, and prediction is closely related. Classification and clustering is another, closely related pair of problems, and their relation to Bertin matrices should be obvious. The USJudgeRatings can be looked at as a classification problem.

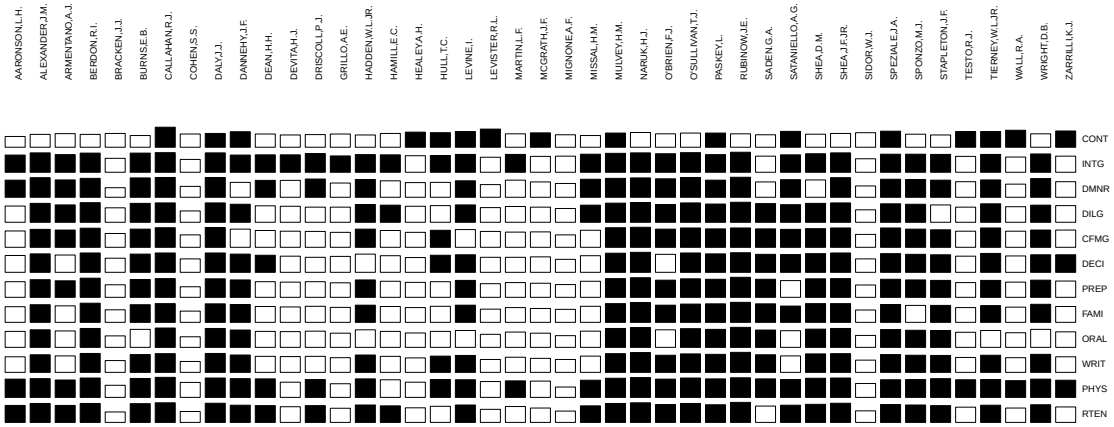


Figure 4: Display of a data matrix: USJudgeRatings data. Lawyers' ratings of state judges in the US Superior Court.

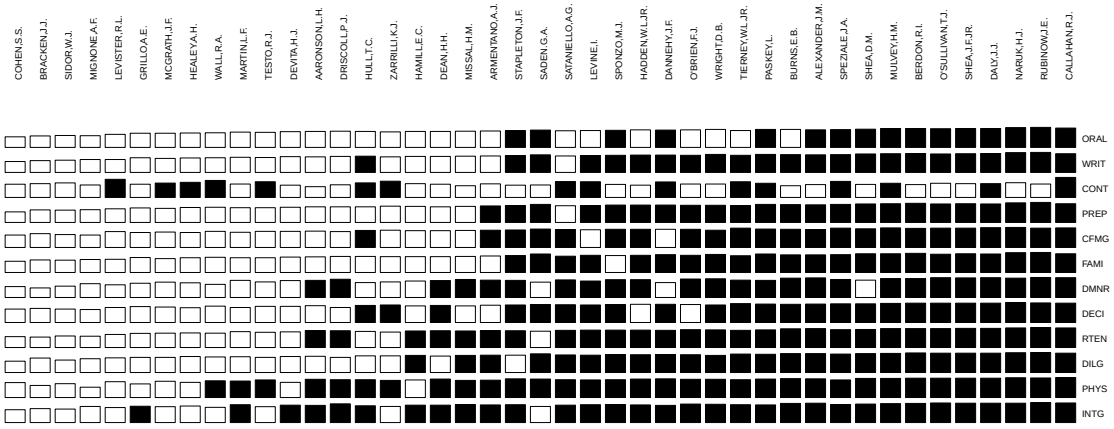


Figure 5: Display of a data matrix. **USJudgeRatings data**. Lawyers' ratings of state judges in the US Superior Court.

3. Work flow

Bertin matrices usually are part of a work flow.

In a first step, we transfer the input data to allow for common, or comparable scales. In the Hotel example, Bertin rescales by the maximum value of each variable. The dichotomous variable Faires is encoded as 0/1. Our implementation default is to rescale for `(0, max)` for positive variables, `(min, 0)` for negative variables, `(min,`

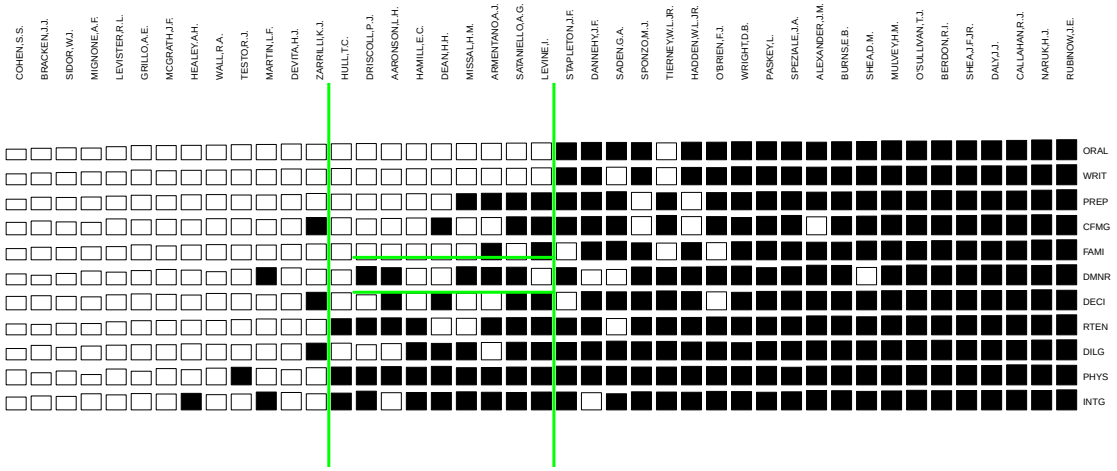


Figure 6: Display of a data matrix. **USJudgeRatings data** Lawyers' ratings of state judges in the US Superior Court. Variable *CONT* removed, and resorted.

max) for general variables. Out preferred, or recommended rescaling however is to use ranks. We use the term *score* for the rescaled variables. Orientation of the data set is critical convention in this step. Usually, rescaling should be by variable, not by case. Depending on the orientation, this can lead for example to ranks by row or by column. We allow global scaling as an additional option for those situations where all data are already on a common scale. Following Bertin, our implementation default is to expect variables in rows, but we provide the means to switch to the R convention with variables in columns. The raw data may come in data frames, or lists, or views on a data base, and the original convention should be preserved. The scores however are a matrix, or an array (which we consider a stacked list of matrices in our context.) We prefer to keep these in Bertin conventions, that is variables are in rows.

In a second step, the scores are translated to visualisation attributes. Colour is handled in two steps. The scores are translated to a colour index, which is used together with a colour palette to determine the display colour for a data element. This allows rapid experiments with various colour palettes, as long as the length of the palettes are compatible. We strongly recommend to always look at the inverted colour palette together with a chosen one to mitigate the effects of colour perception. Simple image displays limit the visualisation attributes to colour. *rect* for example allows rectangle geometry, colour, and border width. Shading and line types should be considered as an alternative for print media.

Visualisation attributes may reflect different aspects. So for example in the classical Bertin display, height of a rectangle is used to reflect the value of a data element, colour is used to show an indicator whether the value is above or below variable mean.

A third step controls the actual placement of the graphical elements. With a matrix layout, it is specified by possible permutations of rows and of columns. This may be related to information used in the first two steps, but should be considered an independent step. A vector or row orders and of column orders is the critical information from this step. Various seriation methods apply. This is where Bertin's ideas about "internal mobility" as a characteristics of modern graphics come to action. The typical situation is to select scores and display attributes, and then search for optimal or good seriations. The arrangement often leads to hard optimisation problems. Placing this step later allows to use information from score transformation and attributes, which may allow more efficient algorithms. In the end, we may be better with a good solution which helps to solve the practical problem, instead of an optimal solution to a theoretical one. These may differ considerably.

The final step is to merge these informations and render a display.

4. Scores

In principle, scores can be generated using an appropriate score function and `apply` or any of its variants. As examples, and for convenience, we provide a small collection of score functions.

As an illustration, each is applied by row to the Hotel data set, and the result is shown using a default Bertin plot. A second plot shows a poor man's regression: assuming that the hotel occupancy, variable 19, is the parameter of interest. Sort the scores by correlation to the score of occupancy.

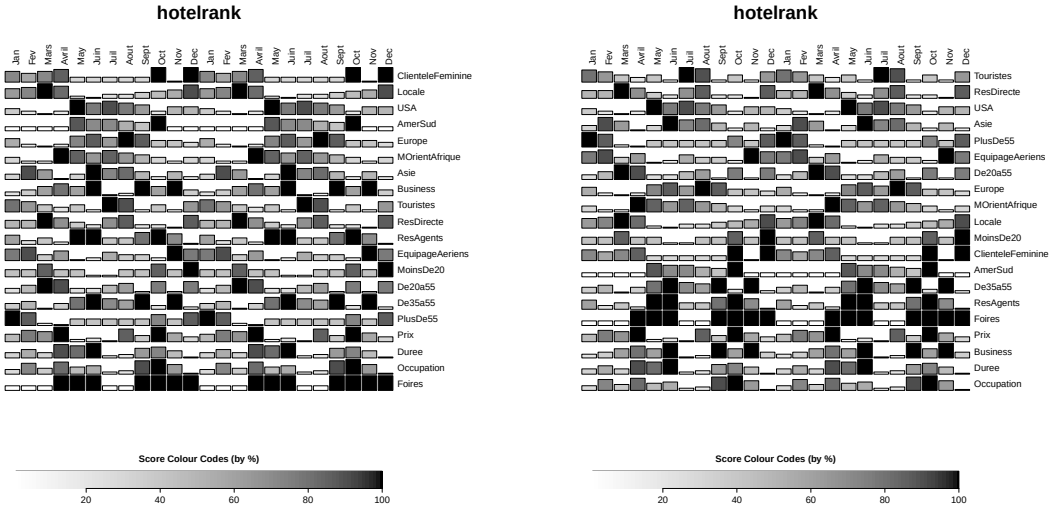
`var.orientation = "byrow"` is the default, it can be omitted. If needed in other data sets, add `var.orientation = "bycolumn"` or `var.orientation = "global"`. This allows to follow our design decision to keep the original data following the original conventions. The results here are matrices. They can be transposed at convenience.

4.1. **Ranks.**

$$x \mapsto rank(x)$$

Input

```
oldpar <-par(mfcol=c(1,2))
hotelrank <- bertinrank(Hotel2)
plot.bertin(hotelrank)
par(mfg=c(1,2)) # fix for stray display allocation
hotelrankorder <- bertin:::ordercor(hotelrank, 19)
plot.bertin(hotelrank, roworder=hotelrankorder)
par(oldpar)
```

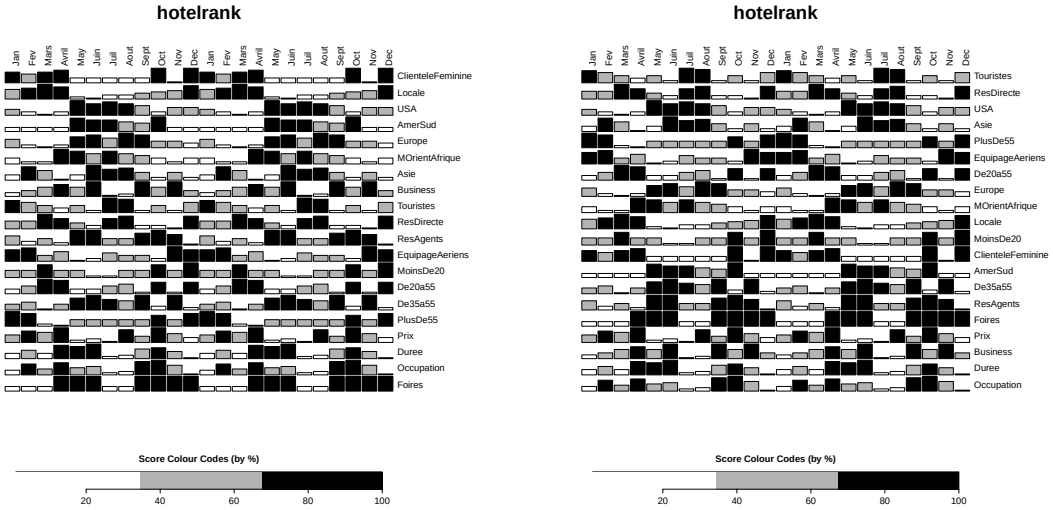


Rank scores have a sound basis. The bring us back to the range of rank statistics. However, since they scale any rank difference by unit steps, they convey order, but not quantitative differences. Preferably they are combined with colour palettes which do not suggest a quantitative scale. The default colour palette uses 256 steps of grey and suggests a quantitative order, whereas the ranks by variable provide at most 12 steps in this example.

It is preferable to use a palette with reduced resolution. For 12 rank values, a scale with 3 or 4 steps should do.

Input

```
oldpar <-par(mfcol=c(1,2))
hotelrank <- bertinrank(Hotel2)
plot.bertin(hotelrank,
        palette = gray((2:0 / 2)^0.5))
par(mfg=c(1,2)) # fix for stray display allocation
hotelrankorder <- bertin:::ordercor(hotelrank, 19)
plot.bertin(hotelrank, roworder=hotelrankorder,
        palette = gray((2:0 / 2)^0.5))
par(oldpar)
```

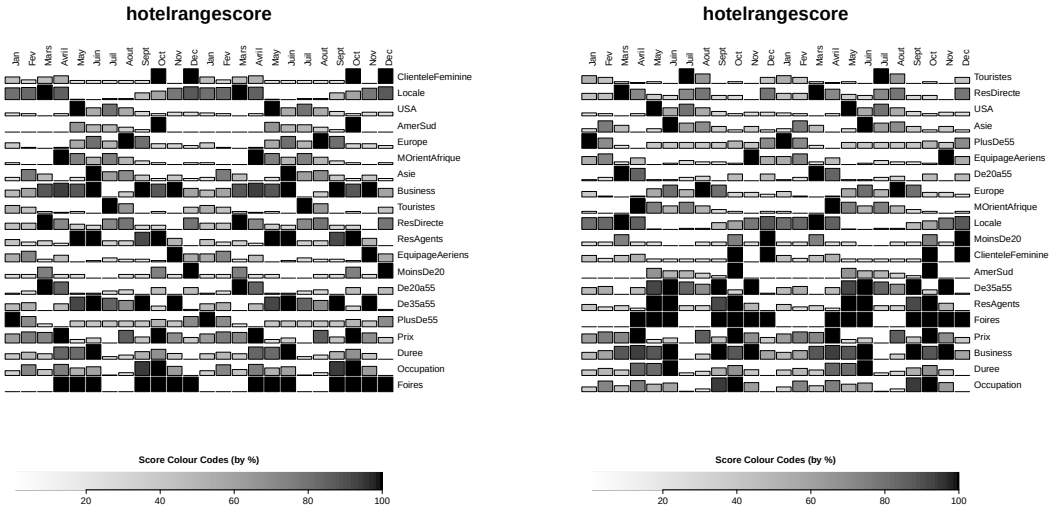

hotelrank
Jan Fev Mars Avril May Juin Juil Aout Sept Oct Nov Dec Jan Fev Mars Avril May Juin Juil Aout Sept Oct Nov Dec
ClienteleFeminine
Locale
USA
AmerSud
Europe
MOrientAfrique
Asie
Business
Touristes
ResDirecte
ResAgents
EquipageAeriens
MoinsDe20
De20a55
De35a55
PlusDe55
Prix
Duree
Occupation
Foires
Score Colour Codes (by %)
20 40 60 80 100
hotelrank
Jan Fev Mars Avril May Juin Juil Aout Sept Oct Nov Dec Jan Fev Mars Avril May Juin Juil Aout Sept Oct Nov Dec
Touristes
ResDirecte
USA
Asie
PlusDe55
EquipageAeriens
De20a55
Europe
MOrientAfrique
Locale
MoinsDe20
ClienteleFeminine
AmerSud
De35a55
ResAgents
Foires
Prix
Business
Duree
Occupation
Score Colour Codes (by %)
20 40 60 80 100

4.2. **z Scores.**

$$x \mapsto \frac{x - mean(x)}{sd(x)}$$

Input

```
oldpar <-par(mfcol=c(1,2))
hotelzscore <- bertinzscore(Hotel2)
plot.bertin(hotelzscore)
par(mfg=c(1,2)) # fix for stray display allocation
hotelzscoreorder <- bertin:::ordercor(hotelzscore, 19)
plot.bertin(hotelzscore, roworder=hotelzscoreorder)
par(oldpar)
```



Bertin uses to highlight "above average" observations. If the data is not degenerate, this corresponds to `bertinzscore > 0`.

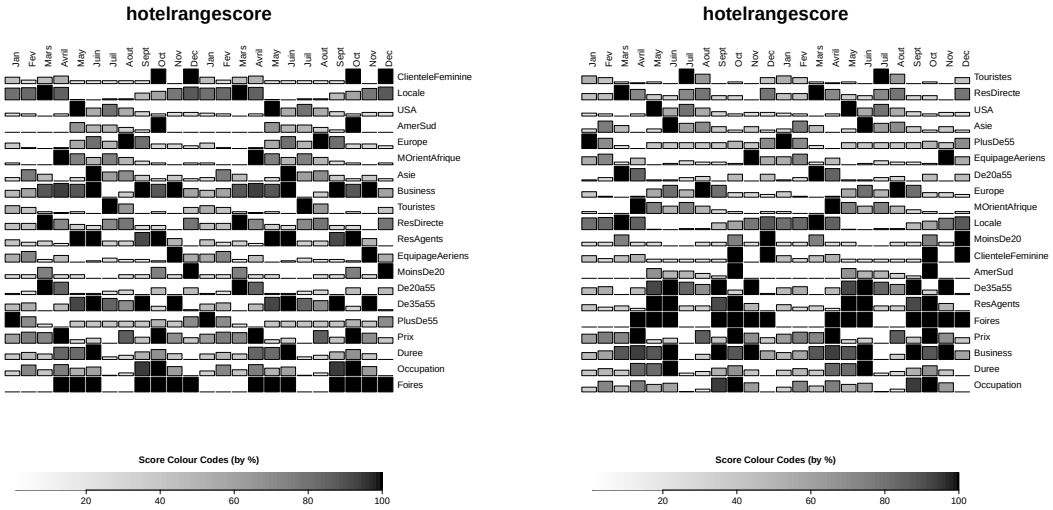
Since z Scores center the variables around the mean, they require positive and negative values for display. This leads in effect to a reduction of the display space to a half, which should be compensated by a more expressive choice of colour coding.

4.3. **Range Scores.**

$$x \mapsto \frac{x - min(x)}{max(x) - min(x)}$$

Input

```
oldpar <-par(mfcol=c(1,2))
hotelrangescore <- bertinrangescore(Hotel2)
plot.bertin(hotelrangescore)
par(mfg=c(1,2)) # fix for stray display allocation
hotelrangescoreorder <- bertin:::ordercor(hotelrangescore, 19)
plot.bertin(hotelrangescore, roworder=hotelrangescoreorder)
par(oldpar)
```



This score just rescales to $[0, 1]$. On contrast to ranks, it preserves quantitative proportions. See Figure 7 on page 13.

Input

```
USJudgeRatings1rngt <- t(bertinrangescore(USJudgeRatings1))
colmeans <- apply(USJudgeRatings1rngt, 2, mean)
colorder<- order(colmeans)
rowmeans <-  apply(USJudgeRatings1rngt, 1, mean)
roworder <- order(rowmeans)
col=ifelse(USJudgeRatings1rngt[,]>rowmeans, "black","white")
bertinrect(USJudgeRatings1rngt[roworder, colorder],col=(col)[roworder, colorder], sepw
```

5. Permutation, Seriation, Arrangement

As Bertin has pointed out,

> *Ce point est fondamental. C'est la mobilité interne de l'image qui charactérise la graphique moderne.* [Bertin 1977, p. 5]

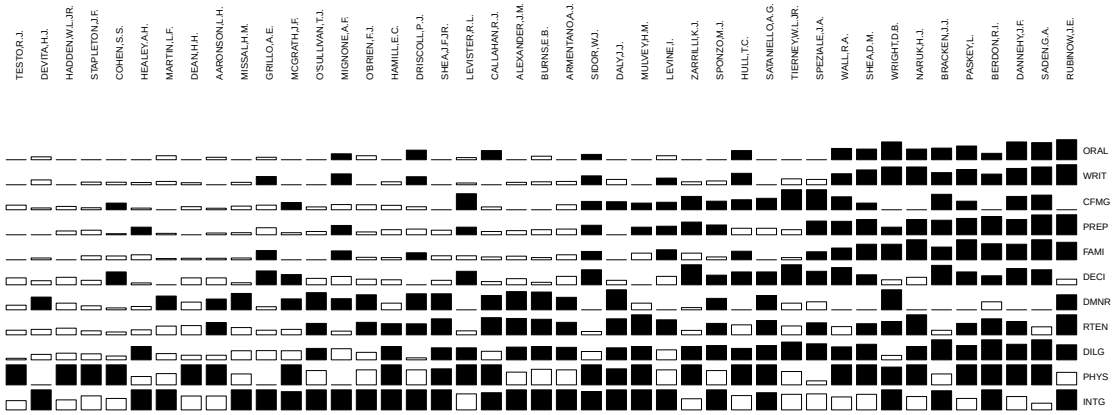


Figure 7: Data set: **USJudgeRatings**. Rescaled by range, rows sorted by row means. Scores above mean are highlighted.

Once we have solved a problem in data analysis, the problem can often be formulated as an optimisation problem. This is the end of the analysis process. In the beginning, while we are searching for a solution, experimenting is necessary. In our implementation, we separate two aspects.

Finding an adequate display is one. This amounts to building up a collection of proven models. A specific data set at hand can contribute by hinting at specific needs and simplify model choice. Building a collection of models and model choice is repeated not so often. Stability of implementation has priority over speed. We will provide a small number of basic model implementations.

The other aspect is to identify critical information. The display, or graphical method, is only giving a framework, and it needs to be filled. For a chosen display, for example, we have to compare different arrangements (seriations, for example). If we allow for interactive work, flexibility has priority. We try to cache the information that is invariant of the permutation.

The classic Bertin display shown above is one of the examples to represent certain models. Following the ideas, but deviating in the details, is to use a simple grey scale image for representation. This may be not the most informative variant. But it is most economic in the use of display space (Figure 8 on page 14).

As a final aspect, display space is limited. The number of variables and cases that can be displayed simultaneously is limited by the pixel size of the display. We can increase it by one or two magnitudes by using e series of detail displays. Any display calibration however should be constant for this series We try to allow for this global calibration.

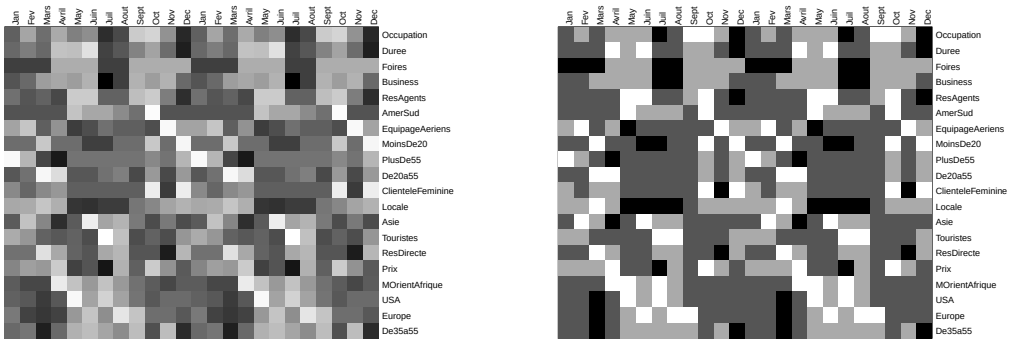


Figure 8: Display of a data matrix as grey scale image: **Hotel data**. Variables are rearranged by similarity to occupation and duration. Grey scale with 256 on the left, reduced grey scale with 4 steps on the right, applied to zscores.

6. Patch Strategy

It may be useful to think of classical statistical methods and see how they may be reflected in a bertin setting. For a start, regression and classification may serve as starting points. For both, traditional statistics has provided solutions, in the classical framework as least squares regression and as discriminance analysis. Many variations have followed.

Classification and regression trees (L. Breiman and Stone, 1984) have thrown a new light on these problems. In principle, CART is a higher dimensional strategy. In the Bertin context, we deliberately restrict ourselves to a two dimensional display context. CART suggests to look for "patches", rectangular areas that allow for an economic model. CART uses a simple strategy, splitting one variable at a point. However this leads to a fragmentation of the data material, dividing it on the average by a factor 2^k for k splits, and gives a corresponding penalty in terms of variance. Friedman (Friedman, 1996) has pointed out that this fragmentation is not necessary. Instead of one set of patches you can have two: one used for estimation, and a second one, possibly different, uses to apply the estimation for fitting. You want to keep the first one large to control variance, and the second one small to reduce bias.

This is not implemented explicitly in the R implementation, but we suggest it as a strategic guide. The examples provided in these notes follow this strategy.

The restriction to a matrix structure is arbitrary and can be omitted. Bertin has been working as a cartographer, and his main work applies to geographical data. What we call the Bertin matrices has been introduced in the very beginning of his book and are but a starting point.

The high level routines accept the usual possibilities of R for subset and index manipulation. As a convenience, the indices are accessible as function arguments. Certain actions are provided as special functions. See the reference pages for details.

7. Colour, perception and pitfalls

Colour may need some experiments to find an informative choice. To allow easy experiments, we use a two step procedure. Based on the original data or the score, we derive a colour index. From an abstract point of view, this is just another score with values in $1, \ldots nrColours$. Colours are provided as a colour palette, and the index is used to select the colour to apply. See `help(palette)` for information on colour palettes.

Some colour palettes that may be useful for Bertin displays are provided. For some palettes, there are variants to highlight the tail behaviour. They are marked by a "2" for quadratic and a "4" for a quartic flattening around the mid value which is mapped to white.

There is an abundant literature on choosing colour palettes for the visual display of quantitative information. Please read.

Be aware that not all people perceive colour the same way. In particular, if you are using red and green in you colour palette, about 6% of the male population will have difficulties.

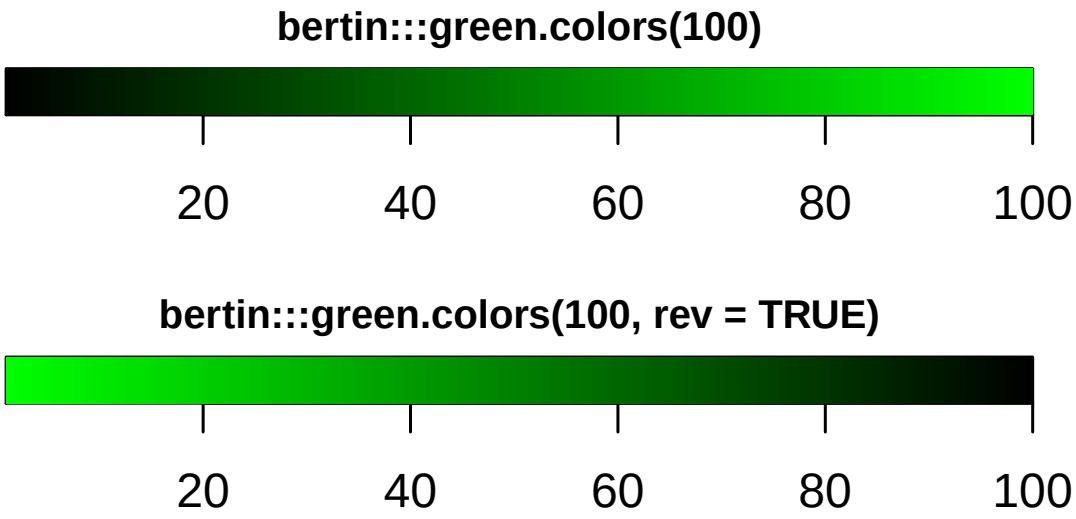
Do trivial tests. If you have a smart colour scale: give a sample to some friends and ask them to tell which shoes higher and which shows lower values.

Colour displays are a means to convey some information. Check that your choices serve this purpose. In particular: if it is of importance to recognise high or low values, chose a colour scale that does not focus on differences in neutral values.

7.1. **Colour Palettes.** Basic colour palettes.

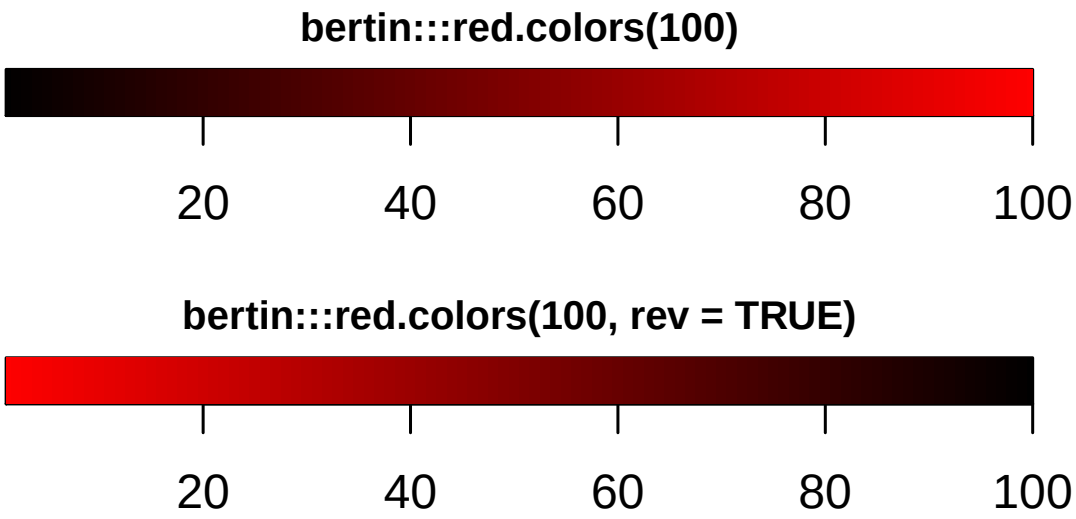
Input

```
oldpar <- par(mfrow=c(2,1), mar=c(2, 1, 2, 1) + 0.1)
colramp(bertin:::green.colors(100))
colramp(bertin:::green.colors(100, rev=TRUE))
par(oldpar)
```



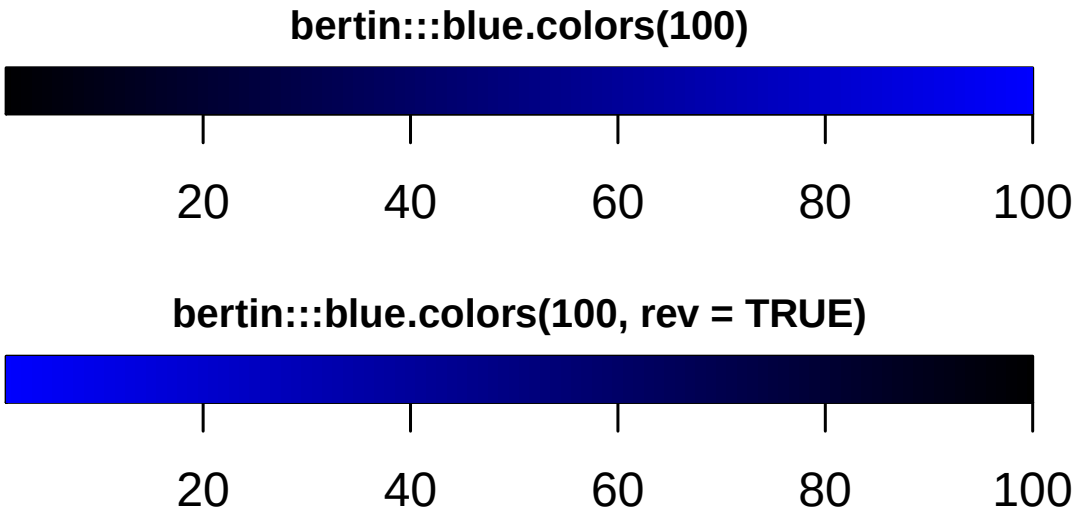
Input

```
oldpar <- par(mfrow=c(2,1), mar=c(2, 1, 2, 1) + 0.1)
colramp(bertin:::red.colors(100), horizontal=TRUE)
colramp(bertin:::red.colors(100, rev=TRUE),horizontal=TRUE)
par(oldpar)
```



Input

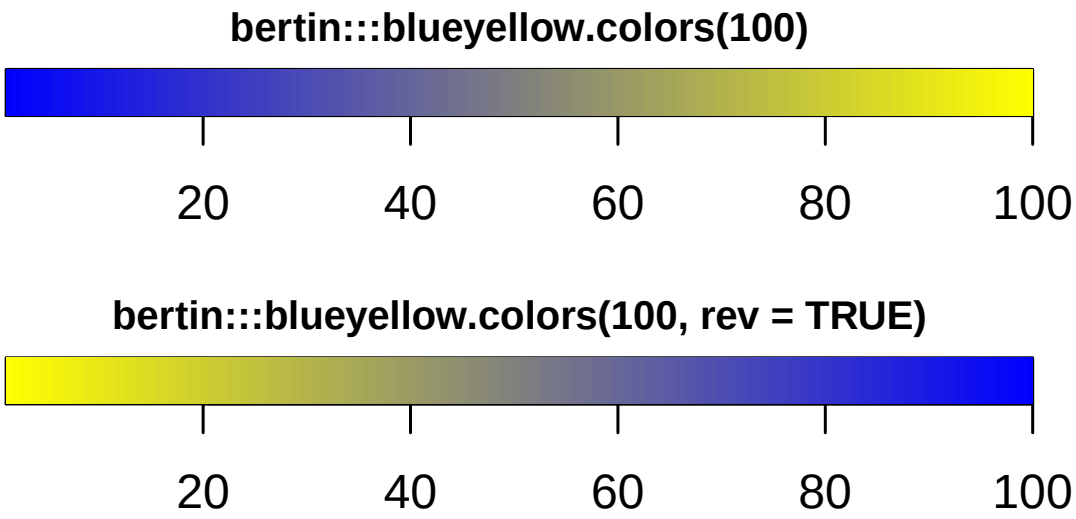
```
oldpar <- par(mfrow=c(2,1), mar=c(2, 1, 2, 1) + 0.1)
colramp(bertin:::blue.colors(100), horizontal=TRUE)
colramp(bertin:::blue.colors(100, rev=TRUE),horizontal=TRUE)
par(oldpar)
```



This is a rather stable palette, with less than 0.1%readers being affected by colour vision particularities. Scale interpretation seeing yellow as higher is also very stable.

Input

```
oldpar <- par(mfrow=c(2,1), mar=c(2, 1, 2, 1) + 0.1)
colramp(bertin:::blueyellow.colors(100),horizontal=TRUE)
colramp(bertin:::blueyellow.colors(100, rev=TRUE),horizontal=TRUE)
par(oldpar)
```

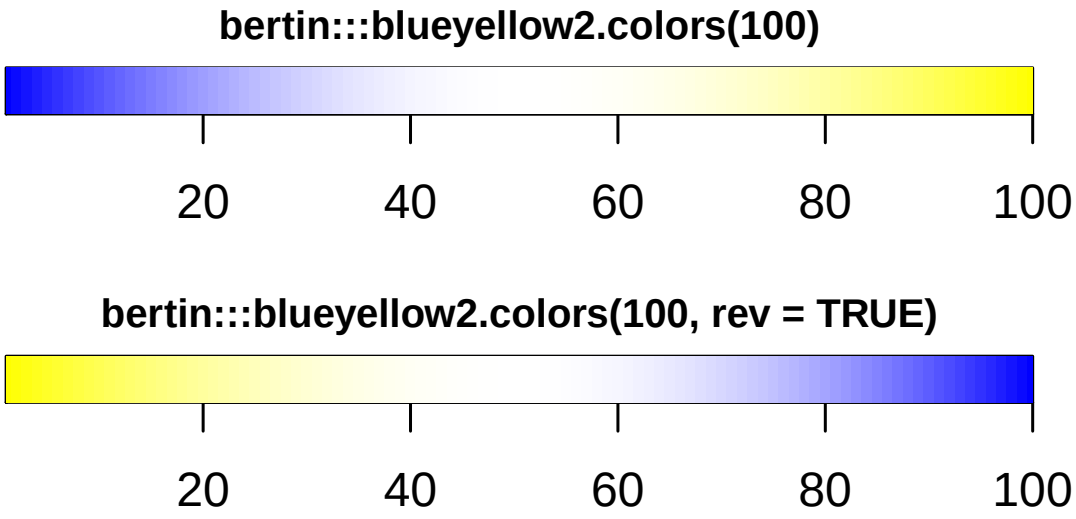


Variant of the above, highlighting the tails and attenuating the center. For an appli-

cation, see Section 9.2 Case Studies: cDNA Data (page 26). This is a rather stable palette, with less than 0.1% readers being affected by colour vision particularities. Scale interpretation seeing yellow as higher is also very stable.

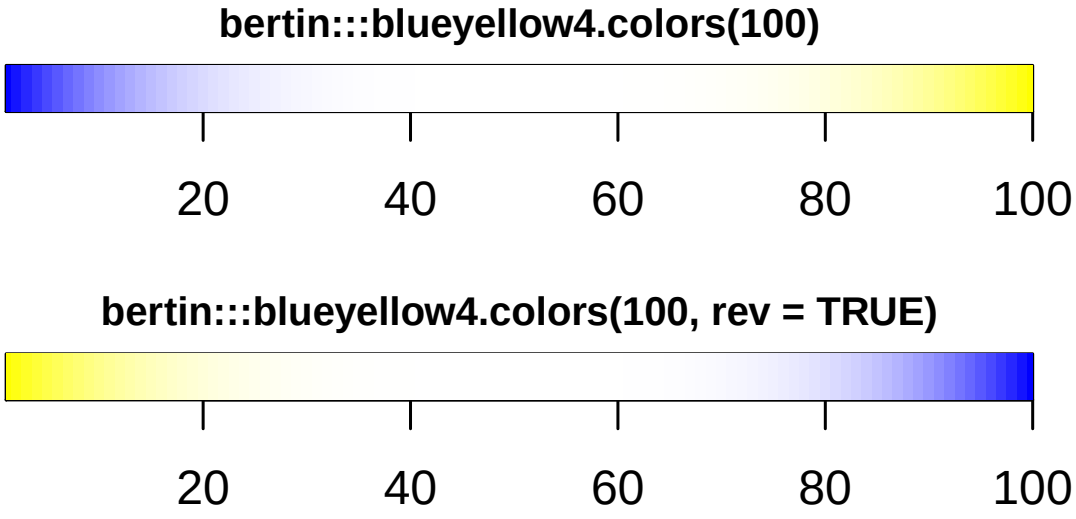
Input

```
oldpar <- par(mfrow=c(2,1), mar=c(2, 1, 2, 1) + 0.1)
colramp(bertin:::blueyellow2.colors(100),horizontal=TRUE)
colramp(bertin:::blueyellow2.colors(100,rev=TRUE),horizontal=TRUE)
par(oldpar)
```



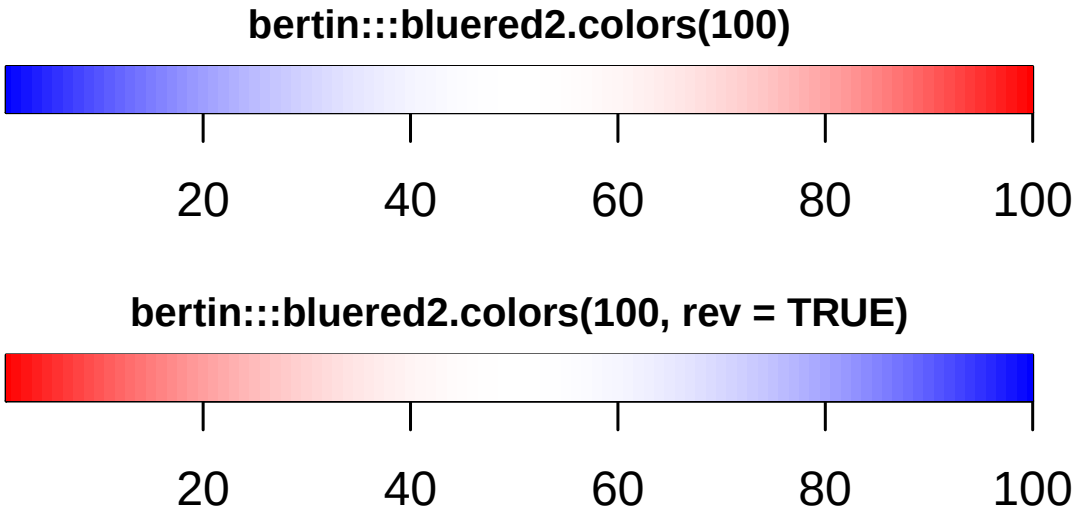
Input

```
oldpar <- par(mfrow=c(2,1), mar=c(2, 1, 2, 1) + 0.1)
colramp(bertin:::blueyellow4.colors(100),horizontal=TRUE)
colramp(bertin:::blueyellow4.colors(100,rev=TRUE),horizontal=TRUE)
par(oldpar)
```

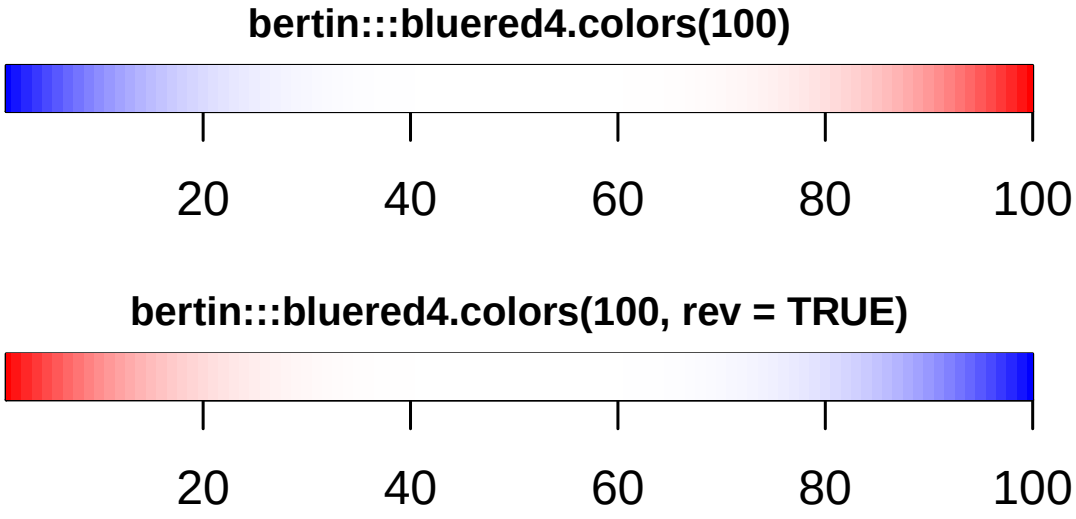


Input

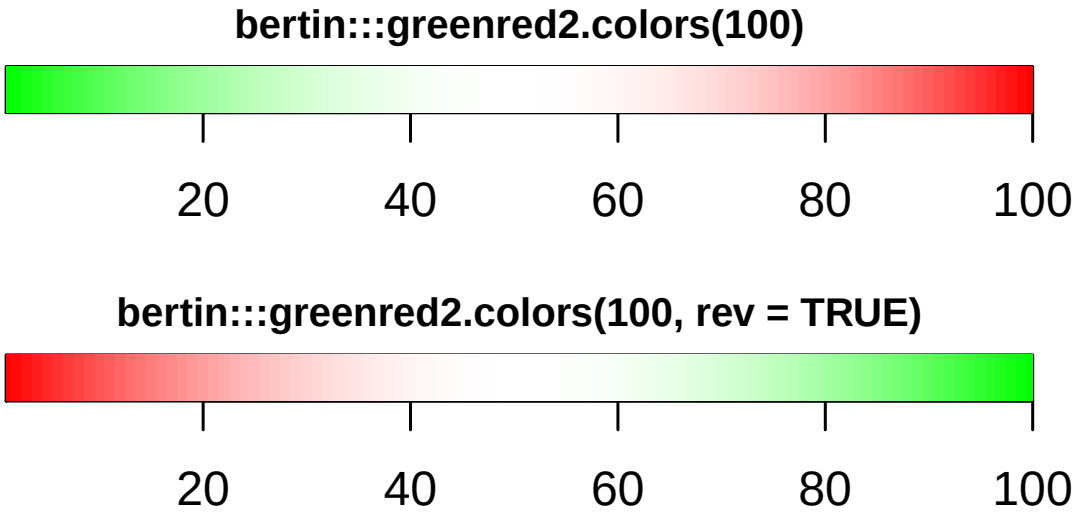
```
oldpar <- par(mfrow=c(2,1), mar=c(2, 1, 2, 1) + 0.1)
colramp(bertin:::bluered2.colors(100),horizontal=TRUE)
colramp(bertin:::bluered2.colors(100,rev=TRUE),horizontal=TRUE)
par(oldpar)
```



Input
```
oldpar <- par(mfrow=c(2,1), mar=c(2, 1, 2, 1) + 0.1)
colramp(bertin:::bluered4.colors(100),horizontal=TRUE)
colramp(bertin:::bluered4.colors(100,rev=TRUE),horizontal=TRUE)
par(oldpar)
```

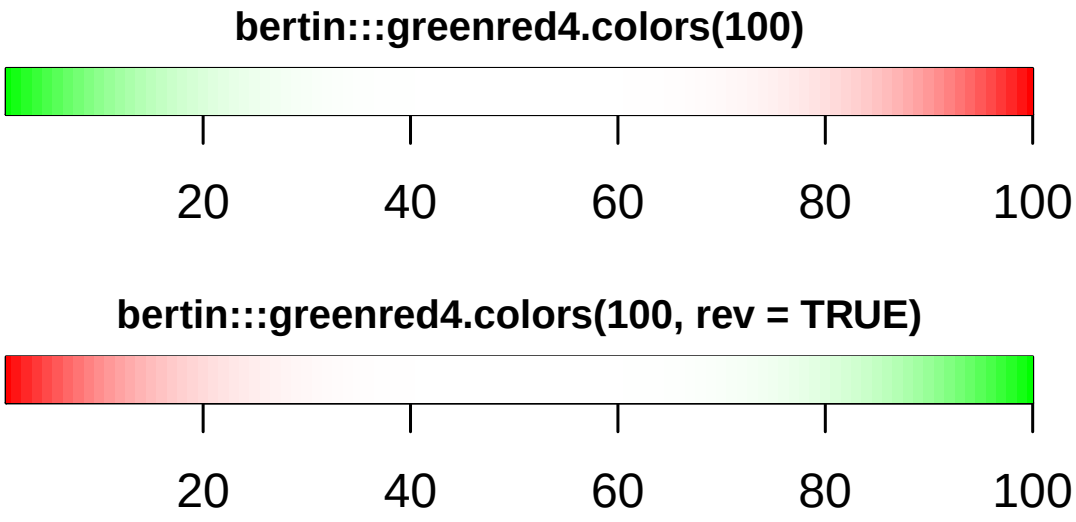


Input
```
oldpar <- par(mfrow=c(2,1), mar=c(2, 1, 2, 1) + 0.1)
colramp(bertin:::greenred2.colors(100),horizontal=TRUE)
colramp(bertin:::greenred2.colors(100,rev=TRUE),horizontal=TRUE)
par(oldpar)
```



Input

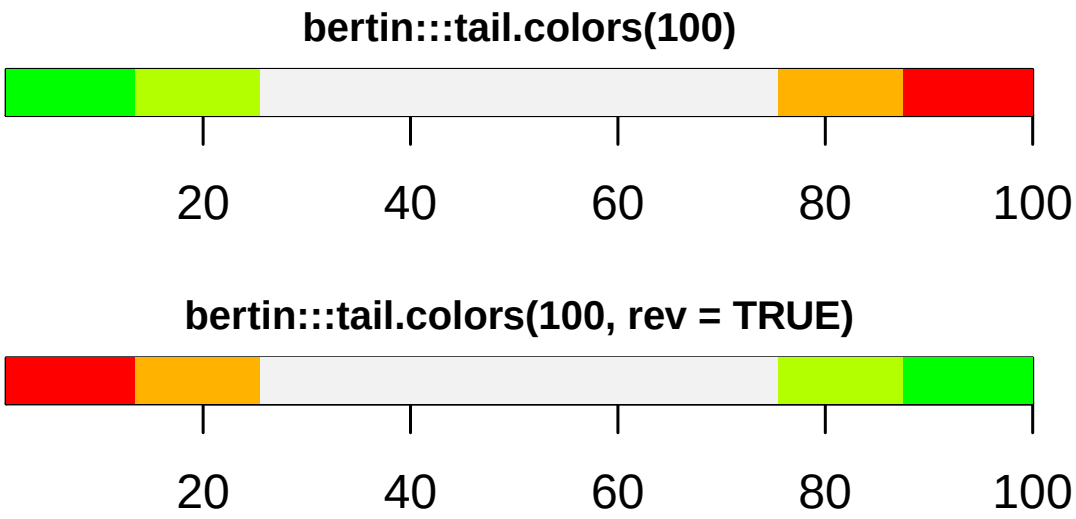
```
oldpar <- par(mfrow=c(2,1), mar=c(2, 1, 2, 1) + 0.1)
colramp(bertin:::greenred4.colors(100),horizontal=TRUE)
colramp(bertin:::greenred4.colors(100,rev=TRUE),horizontal=TRUE)
par(oldpar)
```



In (Tukey, 1991), J. Tukey suggested to use a drastic reduction of information for analysis with many covariates. The tail palettes are provided to support a graphical variant of this strategy.

Input

```
oldpar <- par(mfrow=c(2,1), mar=c(2, 1, 2, 1) + 0.1)
colramp(bertin:::tail.colors(100),horizontal=TRUE)
colramp(bertin:::tail.colors(100,rev=TRUE),horizontal=TRUE)
par(oldpar)
```



Perception is an active process, and any visual presentation may be swayed by the intricacies of perception. Colour perception is particularly complex. When working with colour (and this includes black and white), we strongly suggest to have a look at the image with inverted colours as well.

Here is a sample implementation. On the R level, provide a plotting function

Input

```
sampleimagem <- function(z,
        palette = grey((1:256)/256), xlab, ylab, main,
        colinvert=FALSE){
        if (colinvert) palette <- palette[length(palette):1]
        # x1, x2. y1, y2
        oldpar <- par(fig=c(0, 1, 0.2, 1),
                mar=c(2.5,1.5,0.5,0.5), new=FALSE)
        imagem(z, palette=palette)

        par(yaxt="n", fig=c(0, 1, 0, 0.2),
                mar=c(3.5,12.0,0.5,12.0), new=TRUE)
#        colramp(col=palette, horizontal=TRUE)
        zrange <- range(z, finite=TRUE)
        image(z=t(matrix(seq(zrange[1],zrange[2],length.out=length(palette)),
                1, length(palette))),
                zlim=zrange,main="", ylab="", xlab="", col=palette)
        par(oldpar)
}
```

and run it with `colinvert=FALSE` and `colinvert=TRUE` . If your are using `Sweave`, use two separate chunks, and place the figure output side by side in TEX.

Input

```
hotelrk <- bertinrank(Hotel)
sampleimagem(hotelrk)
```

See Figure 9 on page 23 left.

Input

```
sampleimagem(hotelrk, colinvert=TRUE)
```

See Figure 9 on page 23 right.

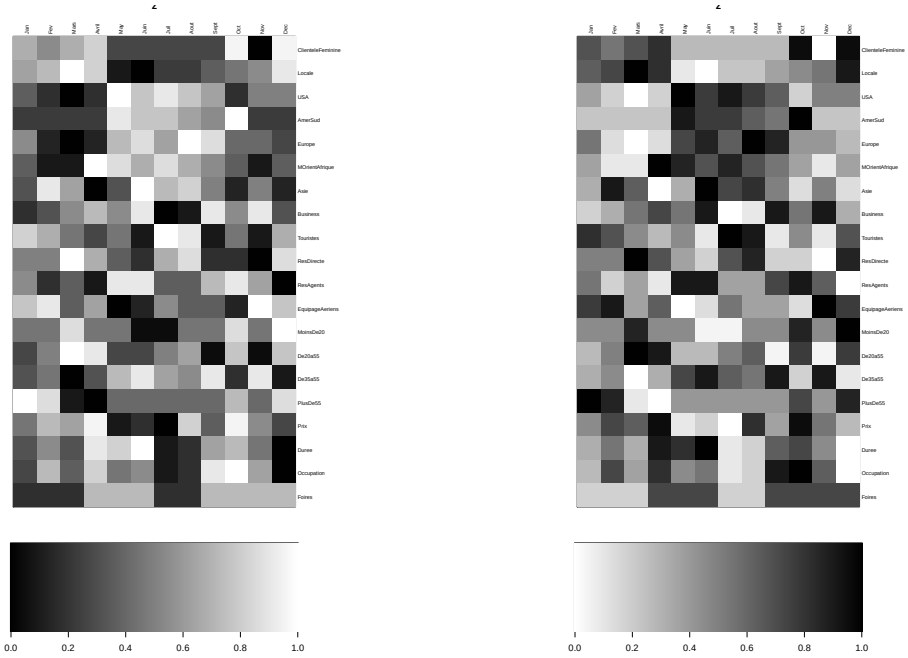


Figure 9: Test matrix: same information, but colour table inverted on the right.

8. Coordinate System and Conventions

We provide prototypes for the display of Bertin matrices. To simplify the implementation of extensions, we chose a coordinate system that allots unit square to each matrix cell. If we want to add separator lines to structure our data, we have to reserve some space `sepwd`, measured in user space.

By convention for a matrix `x` with the data `x[i, j]`, the display is a unit square with bottom left corner at `(i, j)`. Coordinates follow matrix conventions, that is the y axis is pointing downwards, and the top right corner is at `(i+1, j-1)`.

Additional graphical elements can be used immediately, using this coordinate system.

`image` uses the center of a cell as an anchor point, and shows a transposed view. To use `image` to generate an overlay, change the user coordinate system temporarily to `par("usr")-0.5` and use `t()`.

We return the relevant graphical parameters as an invisible result in the basic functions *imagem()* and *bertinrect*. This allows to generate a sequence of graphics with consistent layout.

9. Case Studies

9.1. **Borderline Data.** Figure 10 on page 25 shows the raw data from (Ebner-Priemer and Sawitzki, 2007). A colour palette has been chosen to highlight main structures. The data represent tension levels, recorded by 15 minute intervals, in for control group and for borderline patients. In this representation, to overall difference between controls (top rows) and patients (bottom rows) is obvious. But this is an obvious difference under clinical conditions. What is not obvious, but revealed in this display, is that there is a very small number of "controls" who show a tension characteristic similar to the patients. The stress characteristics in the patient group shows a higher variation. But there is a clear subgroup showing "normal" stress characteristics. This can be seen in the inverted colour plot, but is nearly hidden in the original colour plot.

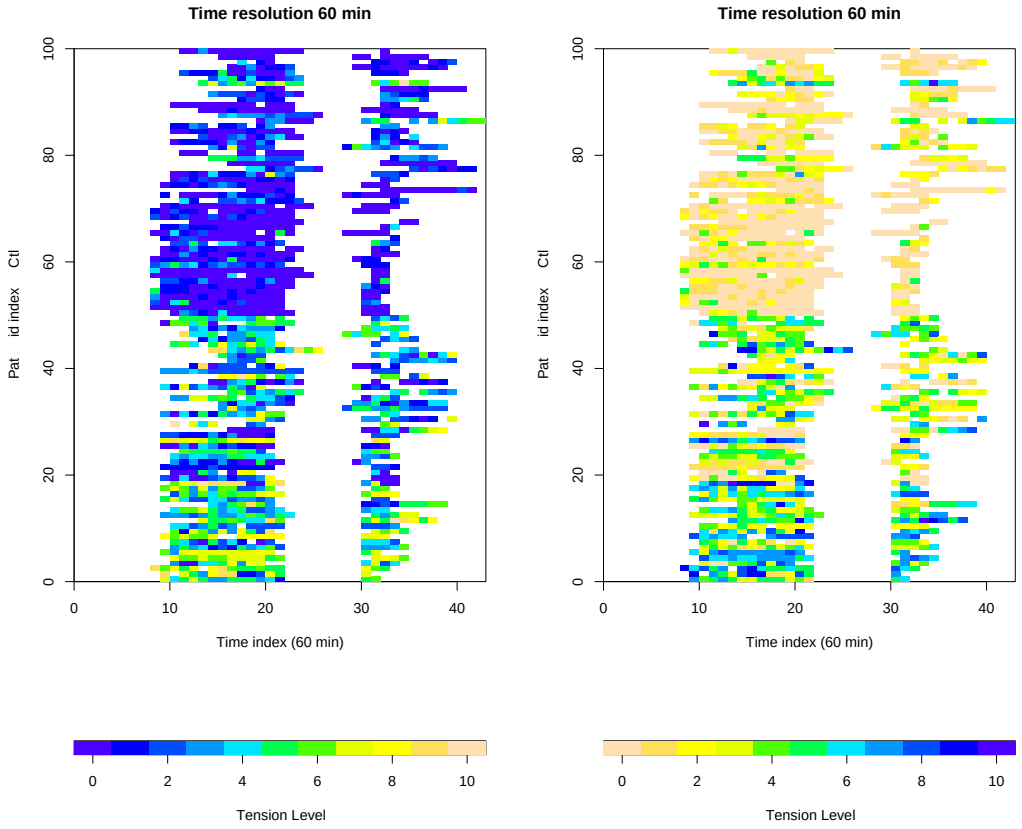


Figure 10: Borderline data. Tension level, all observations, by time.

9.2. **cDNA Data.** In 2-dye cDNA data, hybridisation results are measured using a colour scan, with two samples applied simultaneously to any one microscope slide. The dyes uses for the samples can be recovered by colour separation. Background noise is a major concern. Standard techniques from image analysis is to spot intensity and background intensity for each spot on the slide.Figure 11 on page 26 shows the colour separated channels for foreground and background for one chip (see (Sawitzki, 2002March) for details).

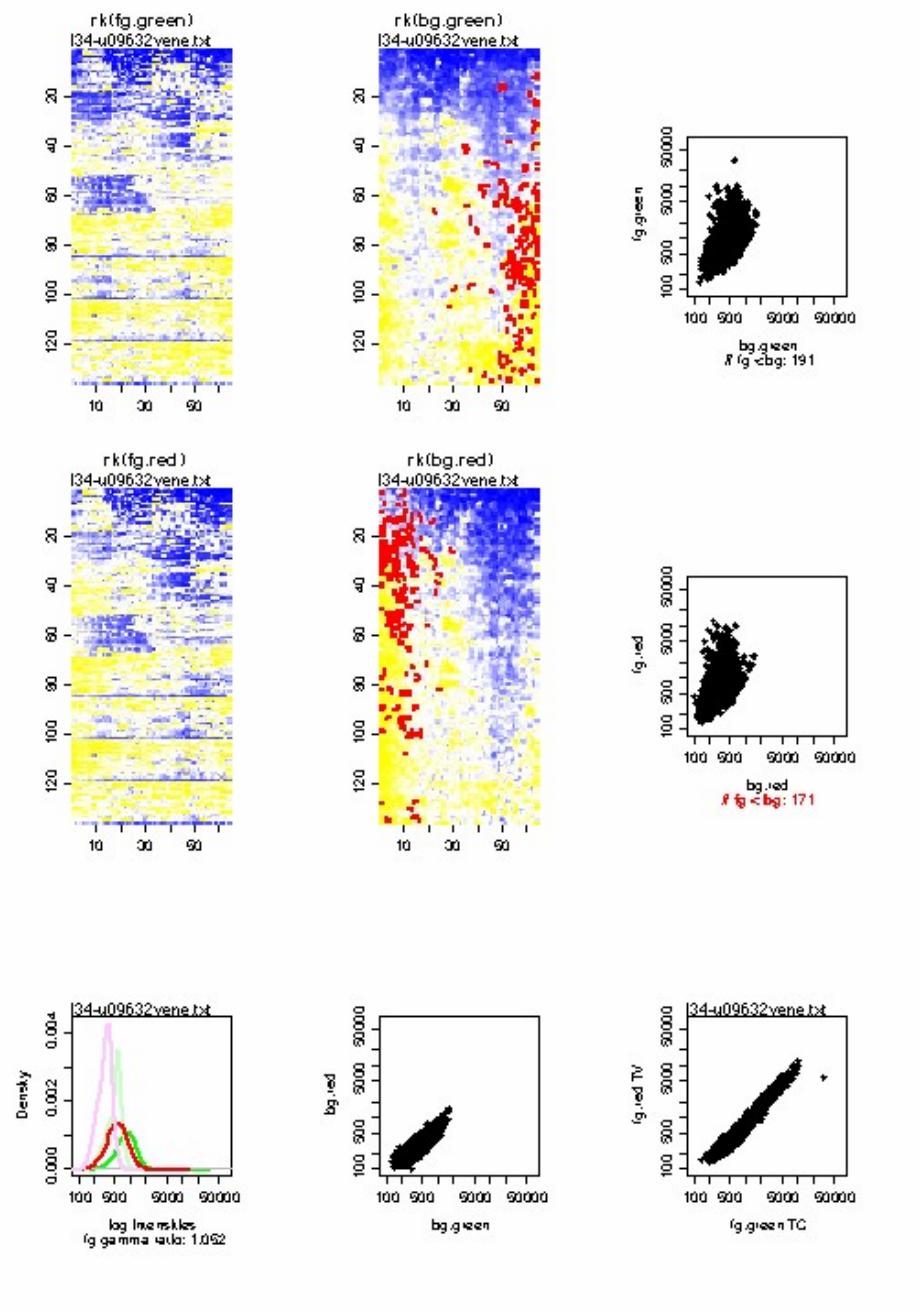


Figure 11: cDNA data. Channel intensities for red and green channel of a cDNA scan (foreground and background).

10. Test matrices

To test the implementation, a series of matrices is provided. Each matrix is shown in four displays: as an image using default setting of the low level function ***imagem***, as an image using the default setting of ***image.bertin***, as a rectangle display using the low level function ***bertinrect*** and as a display using the default setting of ***plot.bertin***.

Input

```
testplot <- function(z, main=deparse(substitute(z)))
{
        oldpar <- par(mfcol=c(2,2))
        bertinrect(z, main= main)
        title(sub="bertinrect", line=1)
par(mfg=c(1,2)) # fix for stray display allocation
        imagem(z, main= main)
        title(sub="imagem", line=1)
par(mfg=c(2,1)) # fix for stray display allocation -- this is very bad
        plot.bertin(z, main= main)
par(mfg=c(2,1)) # fix for stray display allocation -- this is very very bad
        title(sub="plot.bertin", line=1)
par(mfg=c(2,2)) # fix for stray display allocation
        image.bertin(z, main= main)
        title(sub="image.bertin", line=1)
        par(oldpar)
}
```

10.1. **Pure Vanilla Random Matrices.** The most simple case: all variables are on a common scale, and the sequence is given (no seriation possible) or irrelevant (no seriation necessary).

If we want to build test matrices, there are two free parameters to be set, for example

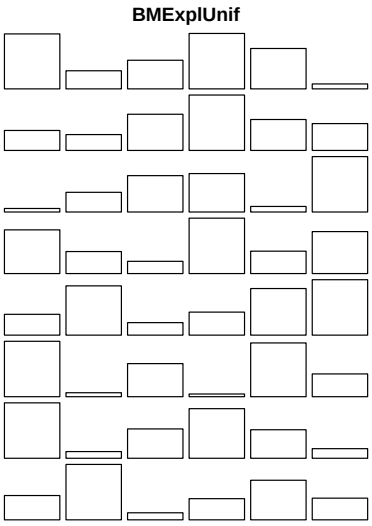
Input

```
BMExplRows=8
BMExplCols=6
```

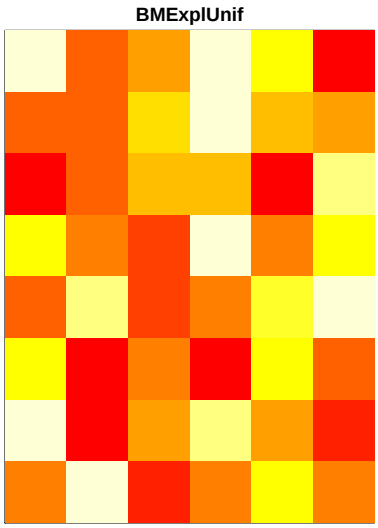
Typical cases are :

Input

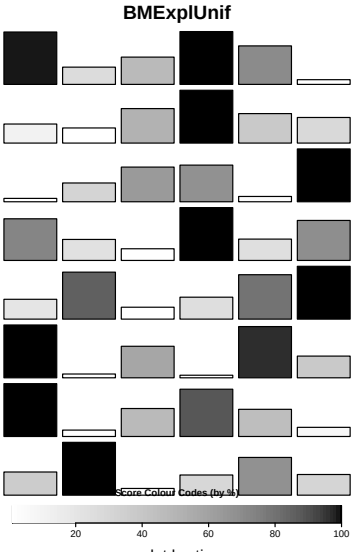
```
BMExplUnif  <- matrix( runif(BMExplRows*BMExplCols),
        nrow= BMExplRows, ncol= BMExplCols)
BMExplNorm  <- matrix( rnorm(BMExplRows*BMExplCols),
        nrow= BMExplRows, ncol= BMExplCols)
```

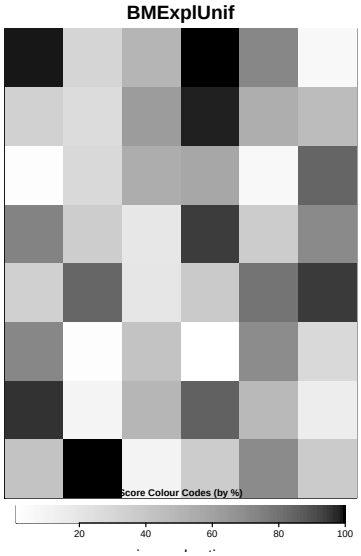


bertinrect

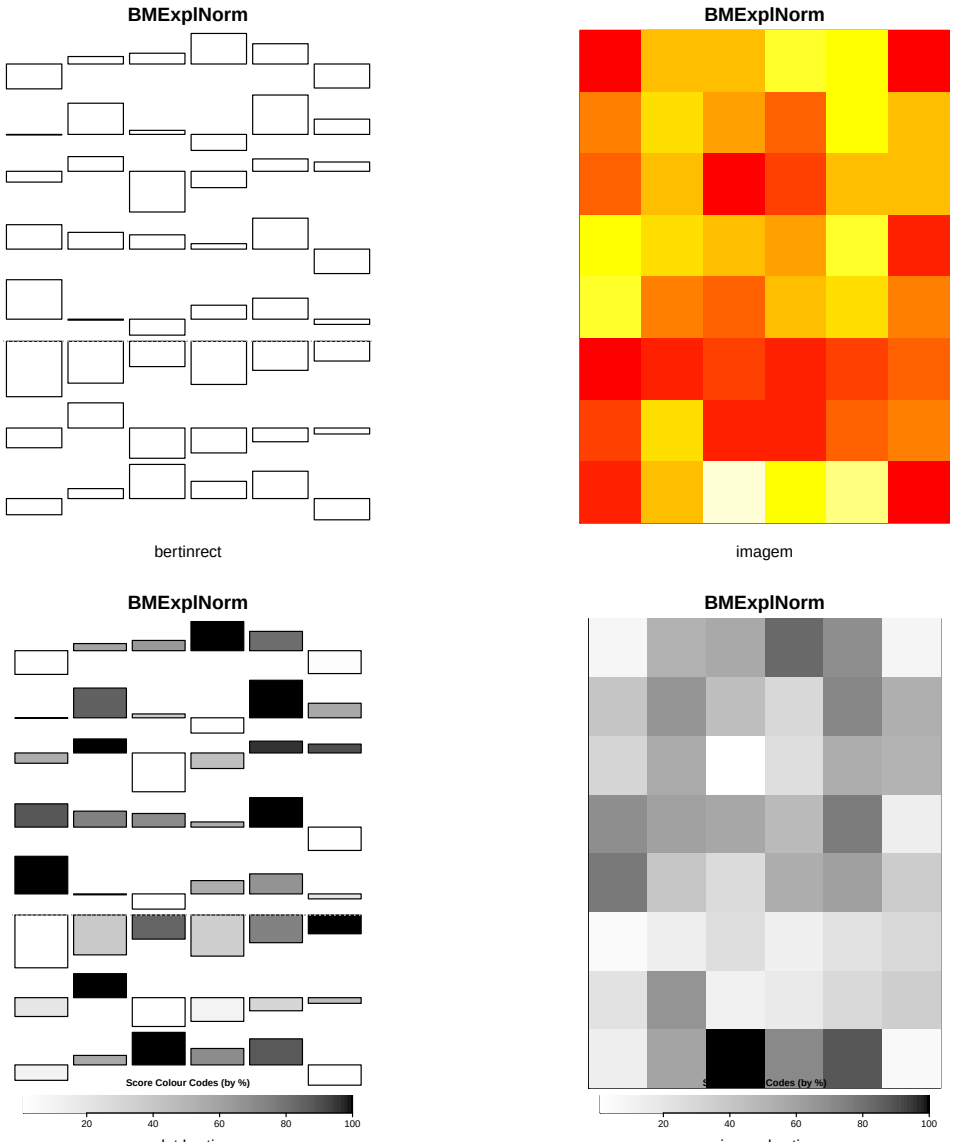


imagem





Random uniform, using default settings.



Random normal, using default settings.

10.2. **Vanilla.** The next round of test cases are numeric, but not on a common scale. We provide some test vectors which we can use to construct various test matrices.

Input

```
# Test vectors, used to build a matrix
Bzero <- rep(0, BMExplCols)
Bone <- rep(1, BMExplCols)
Bmone <- rep(-1, BMExplCols)
Binc <- (1:BMExplCols)/BMExplCols
Bdec <- (BMExplCols:1)/BMExplCols
Bstep <- c(Bmone[1:floor(BMExplCols/2)],
        Bone[(1+floor(BMExplCols/2)):BMExplCols])
Bhat <- Bone
Bhat[(floor(BMExplCols/3)+1):(BMExplCols-floor(BMExplCols/3)) ] <- 0.5
```

```
Bnazero <- rep(c(NA,0),length.out= BMExplCols)
Bnanzero <- rep(c(NaN,0),length.out= BMExplCols)
Binf <- rep(c(Inf,0,-Inf),length.out= BMExplCols)
```

10.2.1. *Basic test matrices.*

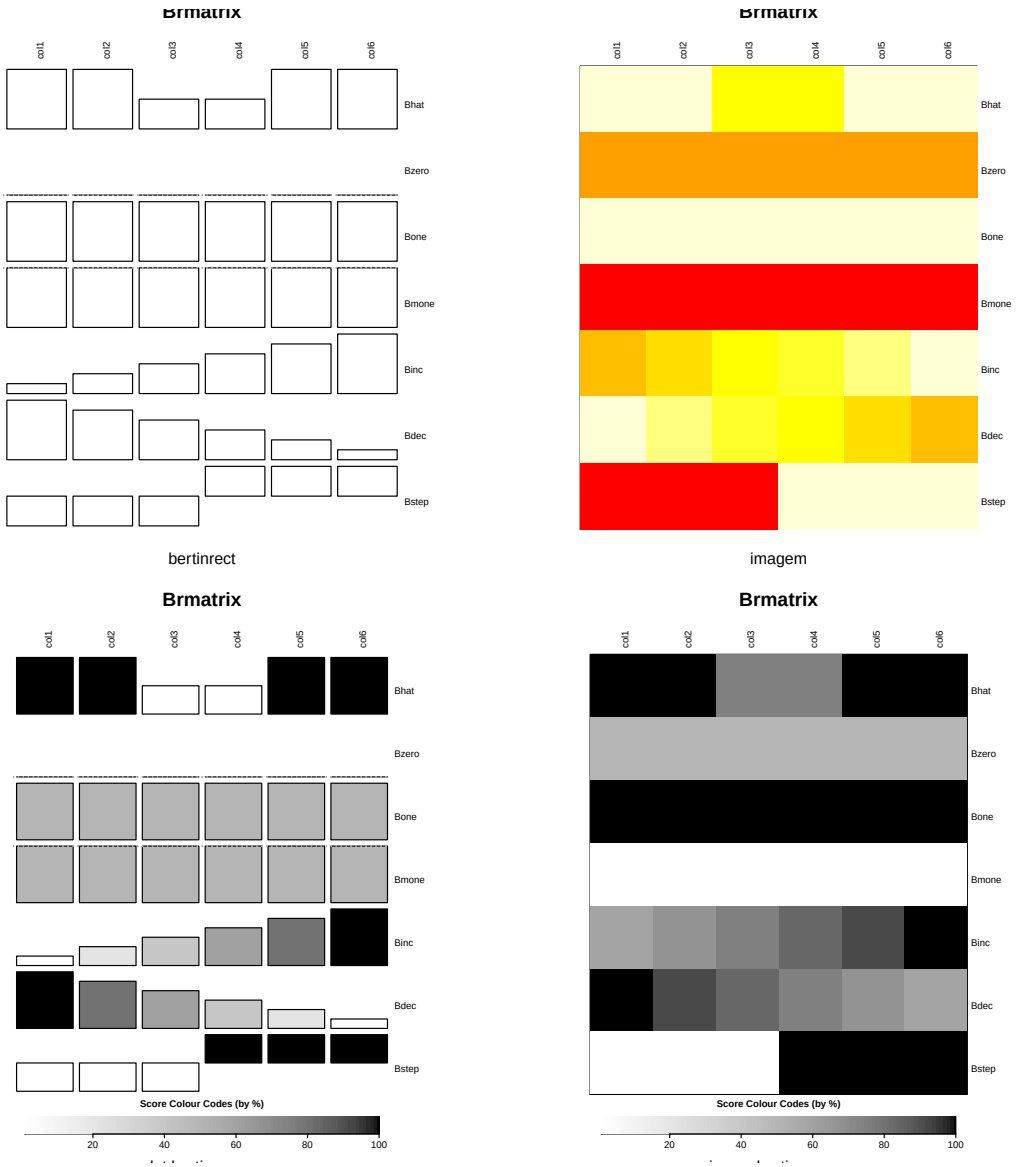
Input

```
Brmatrix <- rbind(Bhat, Bzero, Bone, Bmone, Binc, Bdec, Bstep)
colnames(Brmatrix) <- colnames(Brmatrix, FALSE)
```

See Table 3 on page 30.

	col1	col2	col3	col4	col5	col6
Bhat	1.00	1.00	0.50	0.50	1.00	1.00
Bzero	0.00	0.00	0.00	0.00	0.00	0.00
Bone	1.00	1.00	1.00	1.00	1.00	1.00
Bmone	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
Binc	0.17	0.33	0.50	0.67	0.83	1.00
Bdec	1.00	0.83	0.67	0.50	0.33	0.17
Bstep	-1.00	-1.00	-1.00	1.00	1.00	1.00

Table 3: Brmatrix: test matrix, by row.



Test matrix by row, using default settings

R may use internal housekeeping to keep matrix columns homogeneous. Check! Use row matrix and column matrix for tests.

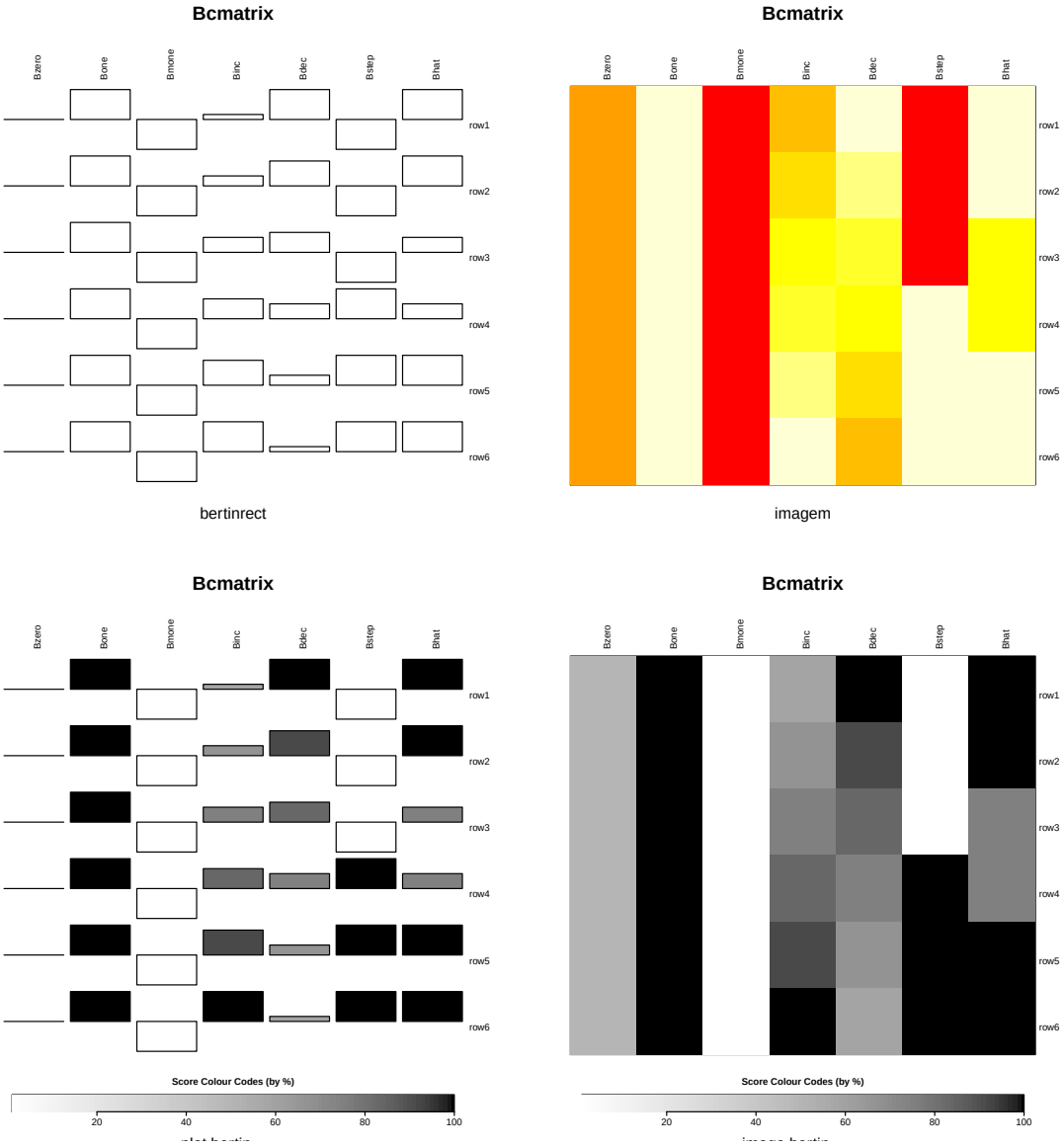
Input

```
Bcmatrix <- cbind(Bzero, Bone, Bmone, Binc, Bdec, Bstep, Bhat)
rownames(Bcmatrix) <- rownames(Bcmatrix,FALSE)
```

See Table 4 on page 32.

	Bzero	Bone	Bmone	Binc	Bdec	Bstep	Bhat
row1	0.00	1.00	-1.00	0.17	1.00	-1.00	1.00
row2	0.00	1.00	-1.00	0.33	0.83	-1.00	1.00
row3	0.00	1.00	-1.00	0.50	0.67	-1.00	0.50
row4	0.00	1.00	-1.00	0.67	0.50	1.00	0.50
row5	0.00	1.00	-1.00	0.83	0.33	1.00	1.00
row6	0.00	1.00	-1.00	1.00	0.17	1.00	1.00

Table 4: Bcmatrix: test matrix, by column.

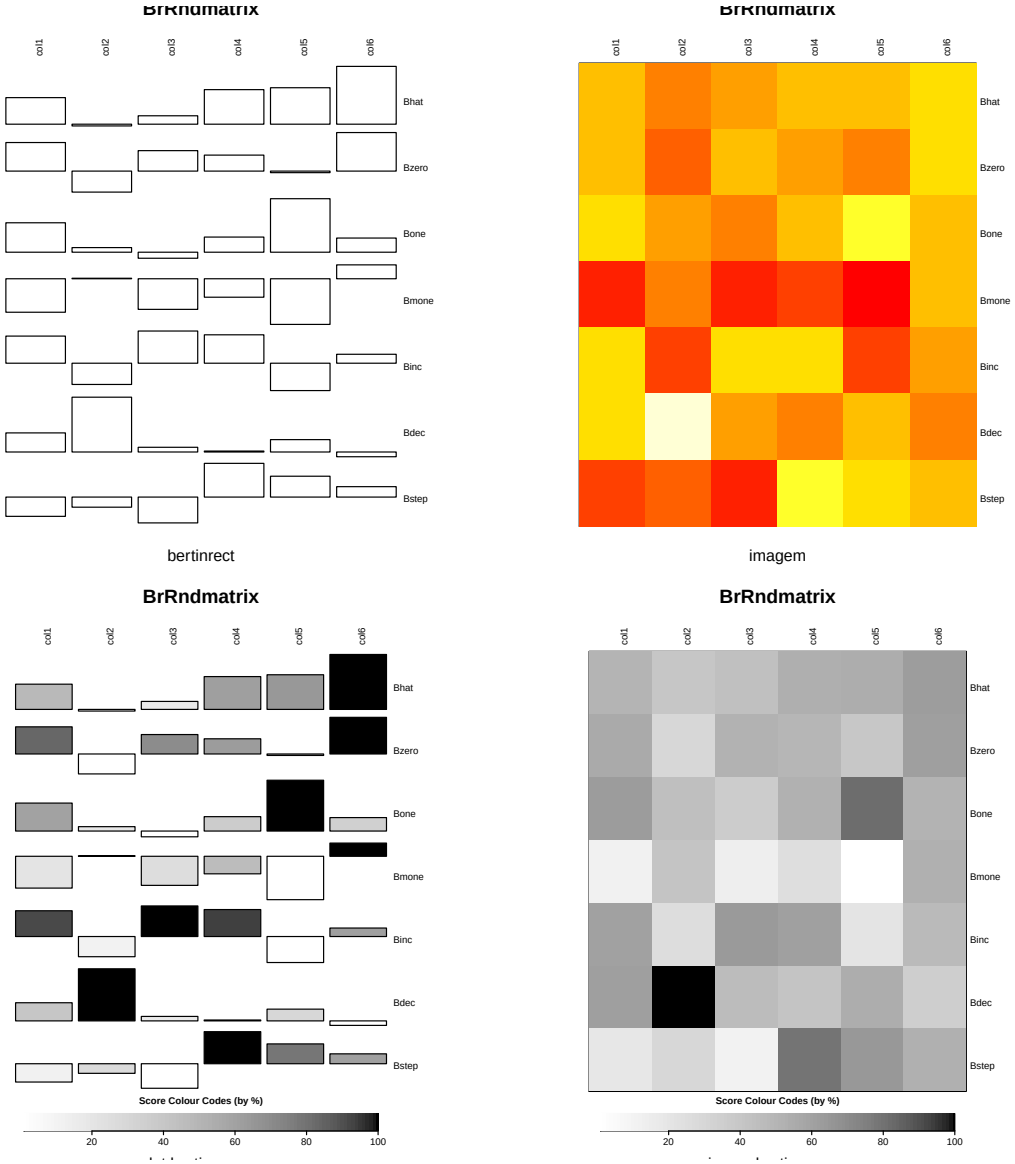


Test matrix by column, using default settings

10.2.2. *Basic test matrices with normal random error.*

Input

```
BrRndmatrix <- Brmatrix+rnorm(nrow(Brmatrix)*ncol(Brmatrix))
```



Test matrix by row with normal random error, using default settings

10.3. **Test matrices with IEEE specials.**

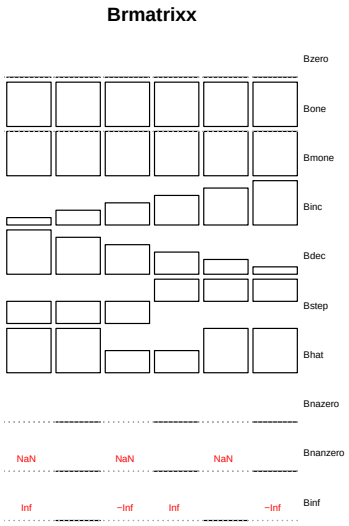
Input

```
Brmatrixx <- rbind(Bzero, Bone, Bmone, Binc, Bdec, Bstep, Bhat,
      Bnazero, Bnanzero, Binf)
```

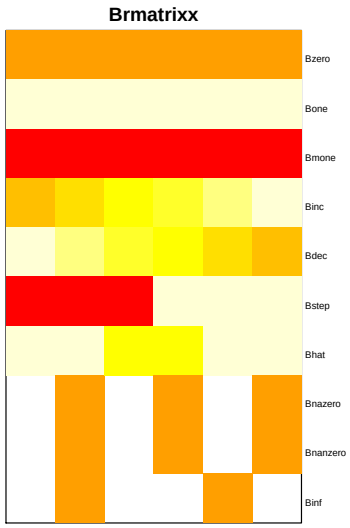
See Table 5 on page 34.

	1	2	3	4	5	6
Bzero	0.00	0.00	0.00	0.00	0.00	0.00
Bone	1.00	1.00	1.00	1.00	1.00	1.00
Bmone	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
Binc	0.17	0.33	0.50	0.67	0.83	1.00
Bdec	1.00	0.83	0.67	0.50	0.33	0.17
Bstep	-1.00	-1.00	-1.00	1.00	1.00	1.00
Bhat	1.00	1.00	0.50	0.50	1.00	1.00
Bnazero		0.00		0.00		0.00
Bnanzero		0.00		0.00		0.00
Binf	Inf	0.00	-Inf	Inf	0.00	-Inf

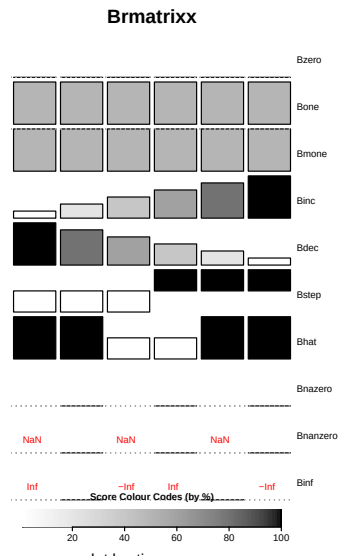
Table 5: Brmatrixx: matrix with special values, by row. NaN and NA values are not printed.

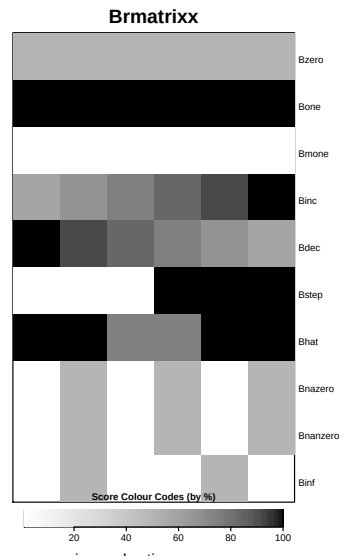


bertinrect



imagem

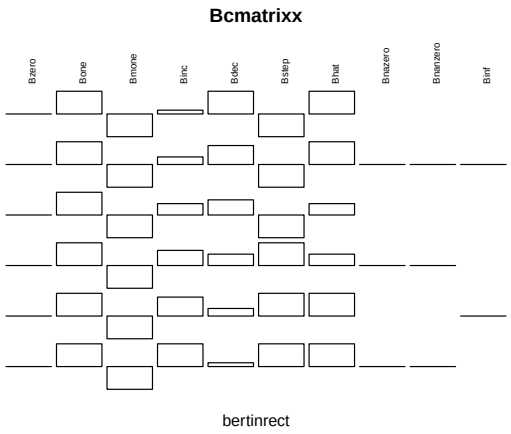


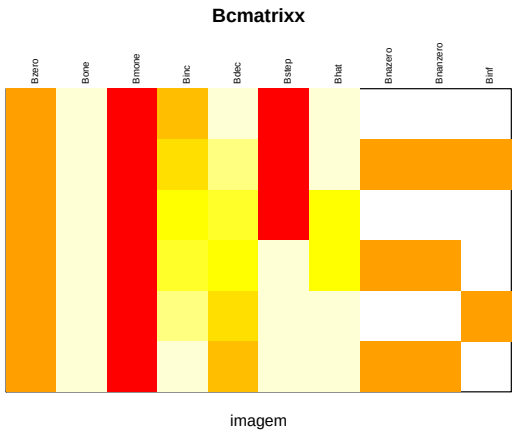


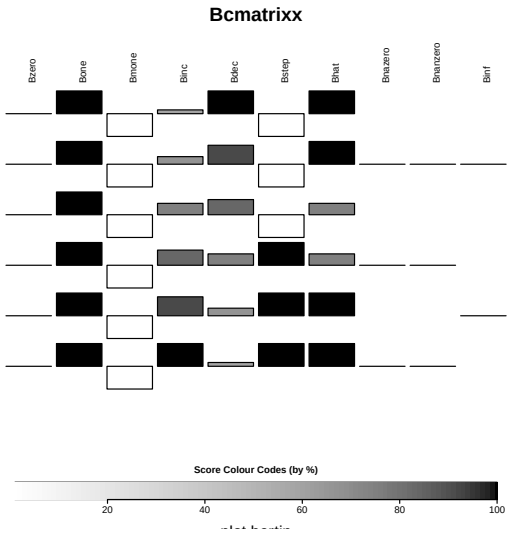
Test matrix with IEEE specials by row, using default settings

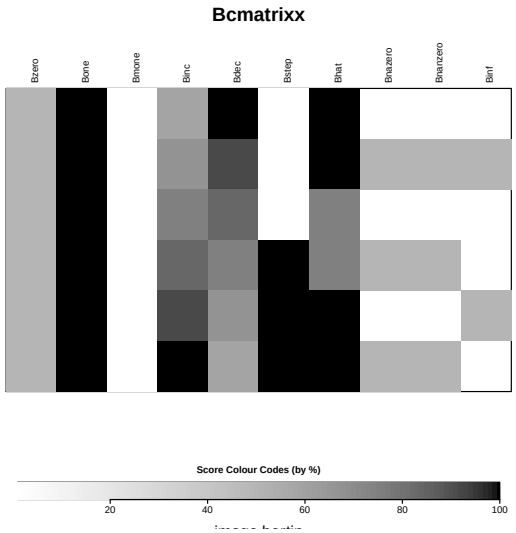
Input

```
Bcmatrixx <- cbind(Bzero, Bone, Bmone, Binc, Bdec, Bstep, Bhat,
        Bnazero, Bnanzero,Binf)
```





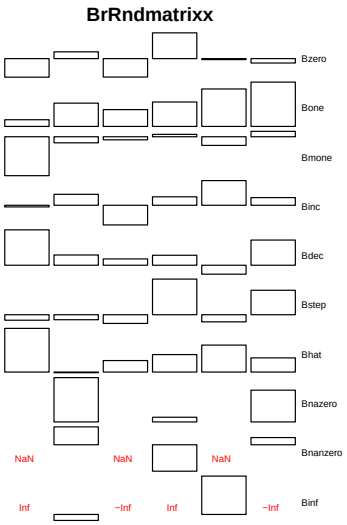




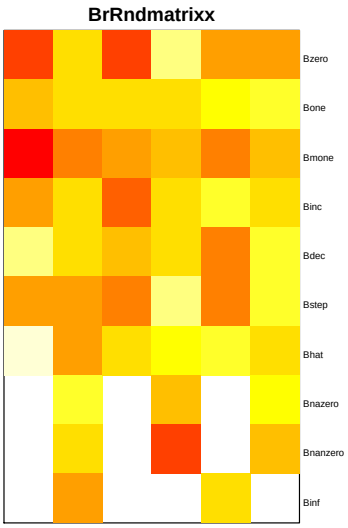
Test matrix with IEEE specials by column, using default settings.

Input

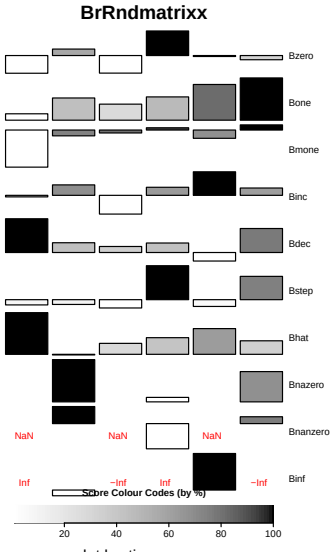
```
BrRndmatrixx <- Brmatrixx+rnorm(nrow(Brmatrixx)*ncol(Brmatrixx))
```

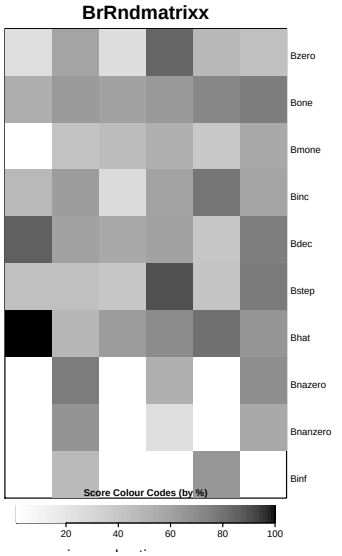


bertinrect



imagem





References

Bertin, J. 1977. *La graphique et le traitement graphique de l'information*, Flammarion, Paris.
_____. 1999. *Graphics and graphic information processing*, Readings in information visualization, pp. 62–65.
de Falguerolles, Antoine, Felix Friedrich, and Günther Sawitzki. 1997. *A tribute to J. Bertin's graphical data analysis*, Softstat '97 (advances in statistical software 6), pp. 11–20.
Ebner-Priemer, Ulrich W. and Günther Sawitzki. 2007. *Ambulatory assessment of affective instability in borderline personality disorder*, European Journal of Psychological Assessment **23**, no. 4, 238–247.
Friedman, J. H. 1996. *Local Learning Based on Recursive Covering*, Department of Statistics, Stanford University.
L. Breiman, R. Olshen, J. Friedman and C. Stone. 1984. *Classification and regression trees*, Wadsworth, Belmont, CA.
Sawitzki, Günther. 1996. *Extensible statistical software: On a voyage to oberon.*, Journal of Computational and Graphical Statistics **5**, no. 3.
_____. 2002March. *Quality Control and Early Diagnostics for cDNA Microarrays*, R News **2**, no. 1, 6–10.
Tukey, John W. 1991. *Use of many covariates in clinical trials*, International Statistical Review / Revue Internationale de Statistique **59**, no. 2, pp. 123–137 (English).

`$Id: bertinR.Rnw 46 2011-09-24 10:03:15Z gsawitzki $`
`$Revision: 46 $`
`$Date: 2011-09-24 12:03:15 +0200 (Sat, 24 Sep 2011) $`
`$Author: gsawitzki $`
textwidth: 6.00612in linewidth:6.00612in

Address: G. Sawitzki, StatLab Heidelberg, Im Neuenheimer Feld 294, D 69120 Heidelberg

E-mail address: gs@statlab.uni-heidelberg.de

URL: http://bertin.r-forge.r-project.org/