DSC 2003 Working Papers
(Draft Versions)
http://www.ci.tuwien.ac.at/Conferences/DSC-2003/

mimR – A package for graphical modelling in R

Søren Højsgaard*

Abstract

The **mimR** package for graphical modelling in **R** is introduced. We present some facilities of **mimR**, namely those relating specifying models, editing models and doing model search. We also discuss the entities needed for flexible graphical modelling in terms of an object structure.

1 Introduction and background

The **mimR** package provides facilities for graphical modelling in the statistical program **R**[1]. The statistical foundation is Mixed Interaction Models, a very general class of statistical models for mixed discrete and continuous variables. Statistical inference in mixed interaction models can be made be the stand–alone program **MIM**[2], and the core of **mimR** is simply an interface from **R** to the **MIM** program. The **mimR** package has its own homepage[3] and is furthermore a part of the gR–project (see `www.r-project.org/gR`) which is a project to make graphical models available in **R**.

There are two aims of this paper. First, to present the **mimR** package as a simple, but operational package for graphical modelling in **R**. Second, to present some tentative ideas regarding an object structure for graphical modelling in **R**.

The reader is assumed familiar with mixed interaction models, and to have a working knowledge of the **MIM** program. Edwards (2000) described both in a very clear way. For a comprehensive account of graphical models we refer to Lauritzen

*Biometry Research Unit, Danish Institute of Agricultural Sciences, Research Centre Foulum, DK–8830 Tjele, Denmark. E–mail: sorenh@agrsci.dk

[1] available from `www.r-project.org`

[2] available from `www.hypergraph.dk`

[3] at `http://www.jbs.agrsci.dk/~sorenh/mimR`

(1996). Other important references are Edwards (1990) and Lauritzen and Wermuth (1984).

2 Preliminaries

2.1 Mixed Interaction Models *à vol d'oiseau*

Mixed interaction models include as special cases log–linear models for contingency tables and covariance selection models for the multivariate normal distribution. More importantly, however, is that the models allow a simultaneous modelling of discrete *and* continuous variables. Focus in mixed interaction models is often (although not exclusively) on conditional independence restrictions.

Within mixed interaction models, one can treat problems where all variables are treated on equal footing (i.e. there are no distinction between variables as being explanatory or responses). Such models are below referred to as *undirected models*. It is however also possible to work with problems where some variables are purely explanatory, other are purely responses and others play both roles. Such models are denoted *block recursive models*. Block recursive models contain undirected models as special cases, and – perhaps more importantly – can be established by a careful combination of undirected models through conditioning.

2.2 MIM as inference engine

From the users perspective, the **MIM** stand alone program can be regarded as an "inference engine" with which the user (at least in principle) needs not be concerned with. However, in reality the **mimR** package is not yet at such a mature level, and this implies that the user in some cases should be aware that there is a separate programming running with which **R** communicates.

2.3 Getting help

In addition to the documentation in the **mimR** package, the **MIM** program itself contains a comprehensive help function which the user of **mimR** is encouraged to make use of.

3 The objects in mimR

The core in **mimR** are the `mimData`, `mimModel`and `mimFormula` objects illustrated in Figure 1.

`mimData` **objects:** A representation of a data frame which can be read by **MIM**. In addition `mimData` contains a unique name `mimData`.id which is subsequently used for linking `mimModel` objects with a `mimData` object.

`mimFormula` **objects:** A representation of a **MIM** model following the **MIM** syntax. `mimFormula` objects were introduced to separate the notion of a statistical model (as an abstract entity) from the concrete notion of a specific data set. Currently, however, the user need not work with `mimFormula` objects. A `mimFormula` specializes into an `UndirectedFormula` and a `BRFormula` representing respectively undirected and block recursive models.

`mimModel` **objects:** Links a `mimFormula` and a `mimData` object together. When the model has been fitted, the `mimModel` object also contains the fitted values, parameter estimates etc. A `mimModel` specializes into an `mimUndirModel` and a `mimBRModel` representing respectively undirected and block recursive models.

It is important to note that to each `mimData` object several `mimModel` objects can be associated. This is not only conceptually attractive, but is also time saving compared with having a data set associated with each model (as is the case with e.g. `lm()` objects in **R**. The reason is that to fit a model, the data set needs to be loaded into **MIM** which takes some time. But when working with several models each connected to the same data set, then this data set needs only be loaded into **MIM** once.

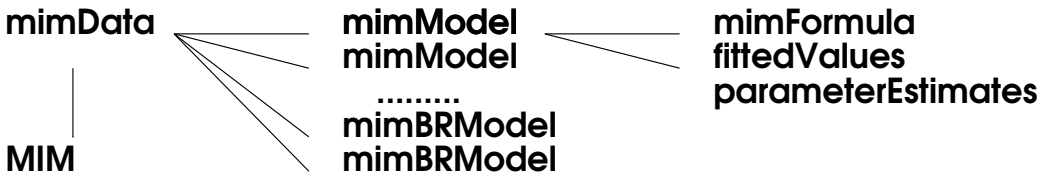

Figure 1: Overview of the object structure in **mimR**

4 The `rats` dataset

Some features of **mimR** will be illustrated in the present paper on the basis of the `rats` dataset. The `rats` dataset is from a hypothetical drug trial, where the weight losses of male and female rats under three different drug treatments have been measured after one and two weeks. See Edwards (2000) for more details. The first rows of the data are:

```
  Sex Drug W1 W2
1   M   D1  5  6
2   M   D1  7  6
3   M   D1  9  9
4   M   D1  5  4
5   M   D2  9 12
6   M   D2  7  7
7   M   D2  7  6
8   M   D2  6  8
    ............
```

5 Declaring data for mimR – `mimData` objects

A `mimData` object is a representation of a data frame which can be understood by the MIM engine. A `mimData` object is created and sent to MIM as follows:

```
data(rats)
mim.rats <- as.mimData(data=rats)
submit(mim.rats)
```

6 Specifying models in mimR

Models are specified using the `mim.model` and `mim.br.model` functions. Note: Neither of these functions check whether the model is syntactically correct in the sense of the restrictions imposed on models in MIM – this is entirely the users responsibility.

6.1 Undirected models – `mimUndirModel` objects

An undirected model is created using the `mim.model` function (which returns a `mimUndirModel` object). For example:

```
mm1 <- mim.model("Sex,Drug/Sex:W1,Drug:W1,
    Sex:W2,Drug:W2/Sex:W1:W2,Drug:W1:W2", data=mim.rats)
```

6.2 Block recursive models – `mimBRModel` objects

Block recursive models can be specified using the `mim.br.model` function (which returns a `mimBRModel` object). First, however, a block recursive structure must be defined:

```
mim.setblock(c("Sex,Drug","W1","W2"),data=mim.rats)
mm2 <- mim.br.model(c("Sex:Drug",
  "Sex:Drug/Sex:Drug:W1/Sex:Drug:W1",
  "Sex:Drug/Sex:Drug:W1,Sex:Drug:W2/Sex:Drug:W1:W2"), data=mim.rats)
```

7 Inspecting the `mimData` object

After specifying the models above, we can get an overview over the models associated with the `rats` data by typing `mim.rats`. Some of the output is:

```
mimData object
  [ mimObject.id: mimData - Wed May 07 12:45:57 2003 - 0.13801 ]
Is currently loaded in MIM: TRUE
  df.names mim.names mim.labels df.levels mim.levels
1      Sex         a        Sex       F M        1 2
2     Drug         b       Drug  D1 D2 D3      1 2 3
3       W1         c         W1
4       W2         d         W2
Mode list:
Model: 1 R-name: mm1
 MIM formula: Sex,Drug/Sex:W1,Drug:W1,Sex:W2,Drug:W2/Sex:W1:W2,Drug:W1:W2
  Is current model: FALSE
```

```
Model: 2 R-name: mm2
 Block: 1 MIM formula: Sex:Drug
 Block: 2 MIM formula: Sex:Drug/Sex:Drug:W1/Sex:Drug:W1
 Block: 3 MIM formula: Sex:Drug/Sex:Drug:W1,Sex:Drug:W2/Sex:Drug:W1:W2
  Is current model: TRUE
```

8 Interacting directly with MIM

8.1 Primitive use of MIM from R – the `mim.cmd()` function

The core of the communication between **R** and **MIM** is the `mim.cmd` function. The arguments to `mim.cmd` are simply **MIM** commands (given as strings). The `mim.cmd` function returns the result of the command submitted to **MIM**. For example:

```
>mim.cmd("fact a2 b2; statread ab; 25 2 17 8 !")
>mim.cmd("mod a,b; fit; print; print f")
```

8.2 Using MIM directly from mimR– the `mcm()` function

The `mcm` function (short for "**MIM** command mode") provides a direct interface to **MIM**, i.e. the possibility to write **MIM** commands directly. The `mcm` function returns no value to **R**, and is intended only as an easy way to submit **MIM** commands without the overhead of wrapping them into the `mim.cmd` function (or submitting the commands directly to **MIM**). To return to **R** from the `mcm` function type 'quite' or 'q'.

9 Model selection and model editing

9.1 Stepwise model selection

To a `mimModel` object the function `stepwise` applies which takes as additional arguments all arguments that the `STEPWISE` command in **MIM** does. The `stepwise` function returns a new `mimModel` object. Taking the model `mm1` as starting point we conduct a backward model selection

```
mm3 <- stepwise(mm1)
 MIM formula: Sex,Drug / Drug:W2,Drug:W1 / Drug:W1:W2
  Is current model: TRUE
  Deviance: 24.2349 DF: 17
```

Different arguments can be given to the `stepwise` function for controlling the model search, e.g. forward/backward search. These arguments are identical to those for the `STEPWISE` command in **MIM**.

9.2 Editing models directly

Models can be edited manually in different ways. One is by adding and deleting edges (interactions) from a model using the `edit.model` function:

```
mm4 <- edit.model(mm3, add="Drug:Sex", del="W1:W2")
 MIM formula: Sex:Drug / Drug:W2,Drug:W1 / Drug:W2,Drug:W1
  Is current model: FALSE
  Deviance: 48.1461 DF: 18
```

Alternatively, one can switch to the **MIM** program or submit commands to **MIM** directly from **R** as described in Section 8. In either case the changed model can be retrieved into **mimR** using the `retrieve.model` function. Suppose the edge between `W1` and `W2` in model `mm3` above has been deleted. Then the new model can be retrieved as

```
mm5 <- retrieve.model(data=mim.rats)
 MIM formula: Sex,Drug / Drug:W2,Drug:W1 / Drug:W2,Drug:W1
  Is current model: TRUE
  Deviance: 48.1461 DF: 20
```

10 Variable names

In **MIM** variables are restricted to be one character only (but **MIM** is case sensitive). To overcome this restriction, a `mimData` object contains a conversion table for converting from the names of a data frame into corresponding one–character names in **MIM**. This conversion table is used by **mimR** when turning a model specified with long variable names into a representation which can be read by **MIM**. The conversion table can be seen by printing a `mimData` object. The user needs generally not be concerned with this issue, except for on a few occasions. One is in connection with calculations on variables in **MIM**, and an illustration of this is given below.

11 Example – Mathematics marks

This dataset (taken from Mardia, Kent, and Bibby (1979)) contains the examination marks for 88 students in 5 different subjects. Data is contained the data set `mathmark` in the **mimR** package. Edwards (2000) also investigates these data.

We start out by specifying the saturated model and do a backward elimination:

```
data(mathmark)
mim.math <- as.mimData(mathmark)
submit(mim.math)
math1 <- mim.model("//mechanics:vectors:algebra:
   analysis:statistics", data=mim.math)
math2 <- stepwise(math1)
 MIM formula:  /  / mechanics:vectors:algebra,algebra:analysis:statistics
  Is current model: TRUE
  Deviance: 0.8957 DF: 4
```

The model `math2` is shown in Figures 2.

Next we consider a latent variable model: We suppose that there is a latent binary variable `L` such that the manifest variables are all conditionally independent given `L`. We fit such a model by:

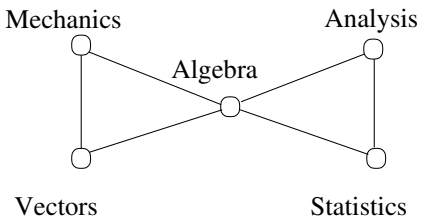

Figure 2: The "butterfly" model selected for the mathmarks data.

```
mathmark[,"L"] <- factor(1:2)
mim.math2 <- as.mimData(mathmark)
submit(mim.math2)
math3 <- mim.model("L/L:mechanics,L:vectors,L:algebra,
    L:analysis,L:statistics/L:mechanics,L:vectors,
    L:algebra,L:analysis,L:statistics",
    data=mim.math2, fit=FALSE)
mim.cmd("calc f=ln(0)")
emfit(math3, plot=TRUE)
EM algorithm: random start values.
 Cycle -2*Loglikelihood      Change
   1          3580.5111
   2          3543.7595  -36.751687
   3          3476.1469  -67.612538
   ..................
  19          3454.9348   -0.000070
Successful convergence.
```

In **MIM**, the letter `f` is the representation of the variable `L` in the data frame, and the statement `calc f=ln(0)` above only serves to set L as a missing value. Setting `plot=TRUE` in `emfit()` creates the plot in Figure 3.

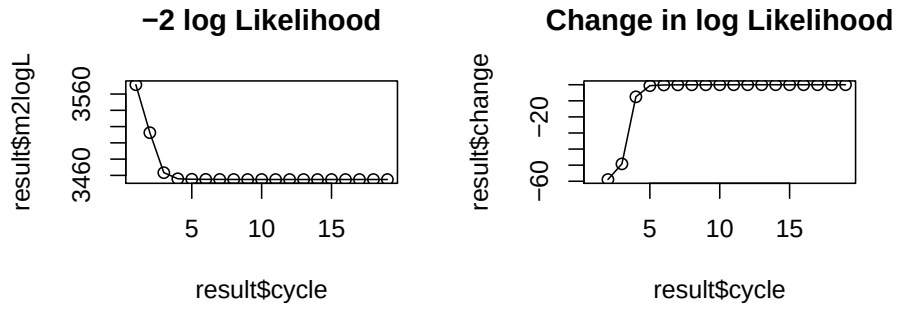

Figure 3: Convergence of the EM algorithm.

We save the predicted values of `L` and plot these against the observation number:

```
mim.cmd("impute")
```

```
m1 <- mim.print("d")
plot(m1$A)
```

The plot is shown in Figure 4. The grouping of the values of A suggests that data have been processed somehow prior to presentation. Edwards (2000), p. 181, conclude: "Certainly they (the data) have been mistreated in some way, doubtless by a statistician."

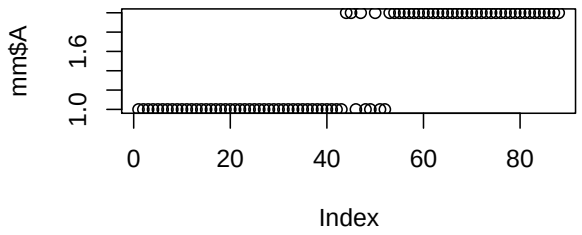

Figure 4: An index plot of the discrete latent variables.

12 Discussion

In this paper we have 1) illustrated some aspects of **mimR** package for graphical modelling in **R**, and 2) presented preliminary ideas regarding an object structure for graphical modelling in **R**. It is the hope that **mimR** will be obsolete in a not too distant future – not because of lack of relevance of being able to work with graphical models in **R**. Rather, it is the hope that a more proper package with this functionality will be implemented as an integrated part of **R**. That is one of the aims of the gR–project. Until that happens we will continue to develop **mimR**. **mimR** is currently at level of development where it is likely that significant changes (e.g. of names of functions and/or object classes) will occur.

13 Acknowledgements

David Edwards (the creator of **MIM**) is greatly acknowledged for his support in the creation of **mimR**. Also the members of the gR project (see `www.r-project.org/gR`) are acknowleded for their inspiration.

References

David Edwards. Hierarchical interaction models. *Journal of the Royal Statistical Society, Series B*, 52(1):3–20, 1990.

David Edwards. *Introduction to Graphical Modelling*. Springer Verlag, New York, 2nd edition edition, 2000.

Steffen L. Lauritzen. *Graphical models*. Oxford University Press, 1996.

Steffen Lilholt Lauritzen and Nanny Wermuth. Mixed interaction models. Technical Report R 84-8, Institute for Electronic Systems, Aalborg University, 1984.

K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis*. Academic Press, 1979.