

# Chapter 5

## Mixed Interaction Models

### 5.1 Introduction

This chapter introduces *mixed interaction models*, a class of models for discrete and continuous variables that combine log-linear models for discrete variables (described in Chap. 2) with graphical Gaussian models for continuous variables (described in Chap. 4). The exposition given here is restricted to *homogeneous mixed interaction models*. Homogeneity in this context means that the covariance matrix of the Gaussian variables does not depend on the values of discrete variables. More general types of mixed interaction models that do not assume homogeneity are described in Lauritzen (1996) and Edwards (2000). An important advantage of the homogeneous models is that they can be specified using model formulae that are similar to the model formulae for log-linear models and for graphical Gaussian models.

### 5.2 Example Datasets

To introduce the models we consider three datasets that are in **gRbase**. The first dataset, `milkcomp1`, comes from a study comparing the composition of sow milk in terms of fat, protein and lactose content under 8 different diets. The control diet consisted of soybean meal, barley and wheat. The other diets added 8% fat to this basis diet: animal fat, rapeseed oil, fish oil, coconut oil, palm oil or sunflower oil. Sow milk was analysed for the concentration of dry matter, protein, fat and lactose: here we consider the data recorded four days after farrowing (i.e., giving birth). For further details see Lauridsen and Danielsen (2004). The first rows of the dataset are:

```
> data(milkcomp1, package='gRbase')
> head(milkcomp1)

  treat fat protein   dm lactose
1     d 6.16     6.65 18.55    5.06
2     c 4.06     5.44 18.32    5.23
3     f 9.25     5.67 20.68    5.15
```

```

4    b 5.82    5.62 17.57    5.74
5    a 4.98    5.37 16.38    5.55
6    b 9.06    5.08 20.21    5.29

```

The second dataset, `wine`, contains the results of a study of the chemical constituents of three varieties of grape, grown in the same region in Italy. There are 178 observations on 14 variables, of which one is discrete (grape variety) and the rest (chemical constituents) are continuous. For more information on this dataset see <http://archive.ics.uci.edu/ml/datasets/Wine>.

```

> data(wine, package='gRbase')
> head(wine)

  Cult  Alch Mlca  Ash Aloa Mgns Ttlp Flvn Nnfp Prnt Clri Hue Oodw
1   v1 14.23 1.71 2.43 15.6 127 2.80 3.06 0.28 2.29 5.64 1.04 3.92
2   v1 13.20 1.78 2.14 11.2 100 2.65 2.76 0.26 1.28 4.38 1.05 3.40
3   v1 13.16 2.36 2.67 18.6 101 2.80 3.24 0.30 2.81 5.68 1.03 3.17
4   v1 14.37 1.95 2.50 16.8 113 3.85 3.49 0.24 2.18 7.80 0.86 3.45
5   v1 13.24 2.59 2.87 21.0 118 2.80 2.69 0.39 1.82 4.32 1.04 2.93
6   v1 14.20 1.76 2.45 15.2 112 3.27 3.39 0.34 1.97 6.75 1.05 2.85

  Prln
1 1065
2 1050
3 1185
4 1480
5 735
6 1450

```

The third dataset, `Nutrimouse`, stems from a study of the effect of nutrition on lipid levels and gene expression in mice. Forty mice were each assigned one of five different diets, with different fatty acid compositions. Two strains of mice were used, one with the  $\text{PPAR}\alpha$  gene knocked out and the other was wild-type (i.e. the  $\text{PPAR}\alpha$  gene was present). The  $\text{PPAR}\alpha$  gene is known to affect fatty acid metabolism. The concentrations of 21 lipids (fatty acids) in the liver were recorded. In addition the data include the expression levels of 120 genes in the liver: these 120 were selected from a much greater number as potentially relevant for nutrition. Thus the dataset contains  $N = 40$  observations of 143 variables: two discrete design variables—genotype (with two levels) and diet (with five levels), 120 gene expression values and 21 lipid values. For more details see Martin et al. (2007).

The following code fragment lists a small subset of the data.

```

> data(Nutrimouse, package='gRbase')
> head(Nutrimouse[,c(1:5,123:126)])

  genotype diet X36b4 ACAT1 ACAT2 C14.0 C16.0 C18.0 C16.1n.9
1      wt  lin -0.42 -0.65 -0.84  0.34 26.45 10.22   0.35
2      wt  sun -0.44 -0.68 -0.91  0.38 24.04  9.93   0.55
3      wt  sun -0.48 -0.74 -1.10  0.36 23.70  8.96   0.55
4      wt  fish -0.45 -0.69 -0.65  0.22 25.48  8.14   0.49
5      wt  ref -0.42 -0.71 -0.54  0.37 24.80  9.63   0.46
6      wt  coc -0.43 -0.69 -0.80  1.70 26.04  6.59   0.66

```

### 5.3 Mixed Data and CG-densities

Suppose that  $N$  observations of  $d$  discrete variables and  $q$  continuous variables are available. We denote the set of discrete variables by  $\Delta$ , the set of continuous variables by  $\Gamma$ , and the combined variable set by  $V = \Delta \cup \Gamma$ .

An observation has the form  $x = (i, y) = (i_1, \dots, i_d, y_1, \dots, y_q)$ . This combines the notation of Chap. 2 and Chap. 4. As in Chap. 2 we write the set of possible cells  $i = (i_1, \dots, i_d)$  as  $\mathcal{I}$ .

We construct a homogeneous *conditional Gaussian density*, or *CG-density* for short, for  $x = (i, y)$  in the following way. Firstly, the probability of the discrete variables falling in cell  $i$  is denoted  $p(i)$ . We assume that  $p(i) > 0$  for all  $i \in \mathcal{I}$ . Secondly, the conditional distribution of the continuous variables given that the discrete variables fall in cell  $i$  is multivariate Gaussian  $\mathcal{N}\{\mu(i), \Sigma\}$ . Observe that the mean may depend on  $i$  but the variance does not. The density takes the form

$$f(i, y) = p(i)(2\pi)^{-q/2} \det(\Sigma)^{-1/2} \exp\left[-\frac{1}{2}\{y - \mu(i)\}^\top \Sigma^{-1}\{y - \mu(i)\}\right] \quad (5.1)$$

The parameters  $\{p(i), \mu(i), i \in \mathcal{I}; \Sigma\}$ , that is, the cell probability and mean vector for each cell  $i$  and the common covariance matrix, are called the *moment parameters*.

It is convenient to represent (5.1) in *exponential family* form as

$$\begin{aligned} f(i, y) &= \exp\left\{g(i) + \sum_u h^u(i)y_u - \frac{1}{2} \sum_{uv} y_u y_v k_{uv}\right\} \\ &= \exp\left\{g(i) + h(i)^\top y - \frac{1}{2} y^\top K y\right\} \end{aligned} \quad (5.2)$$

The parameters  $\{g(i), h(i), i \in \mathcal{I}; K\}$  are called the *canonical parameters*. Note that the canonical parameters have the same dimensions as the moment parameters: for each  $i$ ,  $g(i)$  is a scalar (the discrete canonical parameter) and  $h(i)$  is a  $q$ -vector (the linear canonical parameter); also, the *concentration matrix*  $K$  is a symmetric positive definite  $q \times q$  matrix.

Occasionally it is convenient to use the *mixed parameters* which are given as  $\{p(i), h(i), i \in \mathcal{I}; K\}$ . We allow ourselves to write the parameters briefly as  $\{p, \mu, \Sigma\}$ ,  $\{g, h, K\}$  and  $\{p, h, K\}$ .

We can transform back and forth between the different parameterizations using the relations

$$\begin{aligned} K &= \Sigma^{-1} \\ h(i) &= \Sigma^{-1} \mu(i) \\ g(i) &= \log p(i) - \frac{1}{2} \log \det(\Sigma) - \frac{1}{2} \mu(i)^\top \Sigma^{-1} \mu(i) - \frac{q}{2} \log(2\pi), \end{aligned}$$

and

$$\begin{aligned}\Sigma &= K^{-1} \\ \mu(i) &= K^{-1}h(i) \\ p(i) &= (2\pi)^{\frac{q}{2}} \det(K)^{-\frac{1}{2}} \exp\left\{g(i) + \frac{1}{2}h(i)^\top K^{-1}h(i)\right\}.\end{aligned}\quad (5.3a)$$

## 5.4 Homogeneous Mixed Interaction Models

The *homogeneous mixed interaction models*, which we for brevity here refer to as *MI-models*, are defined by constraining the canonical parameters of CG-densities so as follow factorial expansions.

For example, let  $\Delta = \{A, B\}$  and  $\Gamma = \{X, Z\}$  and let the levels of the factors  $A$  and  $B$  be denoted  $j$  and  $k$ . So in this case  $i = (j, k)$  and  $y = (x, z)$ . The joint density can be written

$$f(i, y) = \exp\left\{g(i) + h^x(i)x + h^z(i)z - \frac{1}{2}(k_{xx}x^2 + 2k_{xz}xz + k_{zz}z^2)\right\} \quad (5.4)$$

and we can write the unrestricted (or saturated) model as

$$g(i) = u + u_j^a + u_k^b + u_{jk}^{ab} \quad (5.5)$$

$$h^x(i) = v + v_j^a + v_k^b + v_{jk}^{ab} \quad (5.6)$$

$$h^z(i) = w + w_j^a + w_k^b + w_{jk}^{ab} \quad (5.7)$$

$$K = \begin{pmatrix} k_{xx} & k_{xz} \\ k_{xz} & k_{zz} \end{pmatrix} \quad (5.8)$$

where the  $u$ 's,  $v$ 's and  $w$ 's are interaction terms. In this model  $g(i)$ ,  $h^x(i)$  and  $h^y(i)$  are unrestricted functions of the cells  $i = (j, k)$ . To estimate the interaction terms uniquely would require some further constraints but we do not bother about this here. This is because we use the *factorial expansions* to constrain the way canonical parameters vary over  $\mathcal{I}$ , but are not usually interested in their values *per se*.

Models are defined by setting certain interaction terms to zero. The usual hierarchical rule, that if a term is set to zero then all higher-order terms must also be zero, is respected. So by this rule, if we set  $v_j^a$  to zero for all  $j$ , we must also set  $v_{jk}^{ab}$  to zero for all  $j$  and  $k$ .

Conditional independence constraints can be imposed by setting interaction terms to zero. For example, to make  $A \perp\!\!\!\perp X \mid (B, Z)$  we must set all terms involving  $A$  and  $X$  in (5.4) to zero, that is,  $v_j^a = v_{jk}^{ab} = 0, \forall j, k$ . To make  $A \perp\!\!\!\perp B \mid (X, Z)$  we must set all terms involving  $A$  and  $B$  to zero, i.e.,  $u_j^a = u_{jk}^{ab} = w_{jk}^{ab} = 0, \forall j, k$ . Finally, to obtain  $X \perp\!\!\!\perp Z \mid (A, B)$  we set  $k_{xz} = 0$ .

For example, consider the `milkcomp1` data:

```
> head(milkcomp1,3)
  treat fat protein    dm lactose
1     d 6.16     6.65 18.55    5.06
2     c 4.06     5.44 18.32    5.23
3     f 9.25     5.67 20.68    5.15
```

The `CGstats()` function calculates the number of observations and the means of the continuous variables for each cell  $i$ , as well as (by default) a common covariance matrix:

```
> library(gRim)
> SS <- CGstats(milkcomp1, varnames=c("treat","fat","protein",
                                     "lactose"))
> SS

$n.obs
treat
a b c d e f g
8 8 8 8 8 7 8

$center
      a      b      c      d      e      f      g
fat    6.641 8.010 7.053 7.401 8.134 7.519 6.974
protein 5.487 5.287 5.475 5.817 5.263 5.296 5.580
lactose 5.491 5.489 5.468 5.314 5.406 5.383 5.415

$cov
      fat protein lactose
fat    2.31288 0.19928 -0.07028
protein 0.19928 0.12289 -0.03035
lactose -0.07028 -0.03035 0.04530
```

Note that the mean of fat (and to a lesser extent of protein) varies over the treatments whereas the lactose means seem to be more or less constant. The coefficients of variation are:

```
> apply(SS$center,1,sd) / apply(SS$center,1,mean)
      fat protein lactose
0.07416 0.03656 0.01187
```

The corresponding canonical parameters are

```
> can.parms<-CGstats2mmodParms(SS,type="ghk")
> print(can.parms, simplify=FALSE)
Mlparms: form=ghk
$g
treat
      a      b      c      d      e      f      g
-745.5 -729.4 -740.5 -743.6 -712.7 -710.5 -740.2

$h
      a      b      c      d      e      f      g
[1,]  0.787  1.628  0.9976  0.8736  1.686  1.344  0.8642
[2,] 88.221 85.006 87.6318 90.1511 84.137 84.817 88.5107
[3,] 181.555 180.651 180.9626 179.0642 178.338 177.745 180.1856
```

```

$K
      [,1] [,2] [,3]
[1,] 0.5056 -0.7503 0.2817
[2,] -0.7503 10.8649 6.1158
[3,] 0.2817 6.1158 26.6104

```

Let  $j$  refer to a level of the treatment factor. Then  $h(j)$  takes the form

$$h(j) = (h^{\text{fat}}(j), h^{\text{protein}}(j), h^{\text{lactose}}(j)).$$

The coefficients of variation for  $h$  are

```

> apply(can.parms$h, 1, sd) / apply(can.parms$h, 1, mean)
[1] 0.324840 0.026150 0.007934

```

which suggests that  $h^{\text{lactose}}(j)$  is constant as a function of  $j$ ; that is

$$h(j) = (h^{\text{fat}}(j), h^{\text{protein}}(j), h^{\text{lactose}}).$$

If we insert this in (5.2) and use the factorization criterion 1.1 we find that

$$\text{lactose} \perp\!\!\!\perp \text{treat} \mid (\text{fat}, \text{protein}).$$

The partial correlation matrix is more informative than the concentration matrix:

```

> conc2pcor(can.parms$K)
      [,1] [,2] [,3]
[1,] 1.00000 0.3201 -0.07679
[2,] 0.32014 1.0000 -0.35968
[3,] -0.07679 -0.3597 1.00000

```

This suggests that the partial correlation between `fat` and `lactose` is zero. If we set  $k_{\text{fat}, \text{lactose}} = 0$  in (5.2) and use the factorization criterion we find that

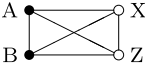
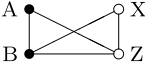
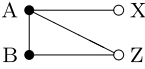
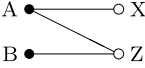

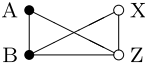

$$\text{lactose} \perp\!\!\!\perp \text{fat} \mid (\text{treat}, \text{protein}).$$

## 5.5 Model Formulae

In this section we describe how to specify MI-models using *model formulae* and show how they may be represented as *dependence graphs*. Here and later we refer to the models and graphs shown in Table 5.1.

As we have seen above in Sect. 5.4, we define an MI-model by constraining  $g(i)$  and the  $h^u(i)$  for  $u \in \Gamma$  to satisfy factorial expansions, and by constraining some off-diagonal elements of  $K$  to zero. So in principle we can define an MI-model by giving a list of generating classes—one for  $g(i)$  and one for  $h^u(i)$  for each  $u \in \Gamma$ —together with list of off-diagonal elements of  $K$  that are allowed to be non-zero. Together these specifications define an MI-model, although some restrictions in the different components are necessary, as we describe below.

**Table 5.1** Some homogeneous mixed interaction models

Model	Formula	Graph	Graphical	Decomposable
(a)	$A*B*X*Z$		true	true
(b)	$A*B*Z+B*X*Z$		true	true
(c)	$A*B*Z+A*X$		true	true
(d)	$A*Z+B*Z+A*X$		true	false
(e)	$A*X+A*Z+B*X+B*Z$		true	false
(f)	$A*B+A*Z+B*X*Z$		false	false
(g)	$A*X+B*X$		true	false

To give all these generating classes would be very cumbersome, however. It is much more convenient to specify a model using a single generating class  $\mathcal{C} = \{G_1, \dots, G_m\}$ , with  $G_j \subseteq V$  for each  $j = 1 \dots m$ . We now explain how this is done.

We use the following convention. We write a generator  $G$  as a pair  $(a, b)$  where  $a = G \cap \Delta$  are discrete variables and  $b = G \cap \Gamma$  are continuous variables. For  $a \subset \Delta$ , by  $g_a(i_a)$  we mean a function which depends on an index  $i$  only through  $i_a$ . Let  $q$  be the number of variables in  $\Gamma$ . Suppose that  $y$  is a  $q$ -vector. For  $b \subset \Gamma$  we write the corresponding subvector of  $y$  as  $y^b$ . Furthermore, we take  $[y^b]$  to mean the  $q$ -vector obtained by padding  $y^b$  with zeros in the right places to obtain full dimension.

Using this convention we can define the restrictions which a generating class  $\mathcal{C}$  imposes on a general (homogeneous) CG-density.

1. The discrete canonical parameter  $g(i)$  is constrained to follow the factorial expansion

$$g(i) = \sum_{(a,b) \in \mathcal{C}} g_a(i_a)$$

That is to say, the generators for  $g(i)$  are the maximal elements of  $\{a \mid (a, b) \in \mathcal{C}\}$ , which we write compactly as  $\max(\{a \mid (a, b) \in \mathcal{C}\})$ . These are called the *discrete generators* of the model.

2. The linear canonical parameter  $h$  is constrained to follow the factorial expansion

$$h(i) = \sum_{(a,b) \in \mathcal{C}} [h_a^b(i_a)].$$

It follows that  $h(i)^\top y = \sum_{(a,b) \in \mathcal{C}} h_a^b(i_a)^\top y_b$ . For each  $u \in \Gamma$ , the generators for  $h^u(i)$  are  $\mathcal{C}^u = \max(\{a \mid (a, b) \in \mathcal{C} \wedge u \in b\})$ ; that is, the discrete components of those generators containing  $u$ . These are termed the *linear generators* of the model.

3. Finally, the quadratic canonical parameter  $K$  is constrained as follows: elements  $k_{uv}$  of  $K$  are set to zero unless  $\{u, v\} \subset b$  for some generator  $(a, b) \in \mathcal{C}$ . The sets  $\{b \mid (a, b) \in \mathcal{C}\}$  induce a graph whose edges correspond to those  $k_{uv}$  which are not set to zero. The cliques of the graph are called the *quadratic generators* of the model.

For example, the last model in Table 5.1 has the generating class

$$\{(A, B), (A, Z), (B, X, Z)\}.$$

The derived formulae for  $g(i)$ ,  $h^x(i)$  and  $h^z(i)$  are  $\{(A, B)\}$ ,  $\{(B)\}$ , and  $\{(A), (B)\}$ , respectively. Hence  $g(i)$  is unrestricted,  $h^x(i)$  satisfies  $h^x(i) = v + v_k^b$  for all  $i = (j, k)$  and  $h^z(i)$  satisfies  $h^z(i) = w + w_j^a + w_k^b$  for all  $i = (j, k)$ . Since  $(X, Z) \subset (B, X, Z)$ ,  $k_{xz}$  is not set to zero.

It can be shown that to ensure location and scale invariance, the formula for  $g(i)$  must be “larger” than the formulae for each  $h^u(i)$  in the sense that each generator for  $h^u(i)$  must be contained in a generator for  $g(i)$ . This constraint is automatically fulfilled by the above construction.

The model formula notation for MI-models used here has the disadvantage that distinct formulae can specify the same model. For example, if  $\Delta = \{I\}$  and  $\Gamma = \{X, W, Z\}$  then the formulae  $I * X * W + X * W * Z$  and  $I * X * W + X * Z + W * Z$  give identical models. This is not usually problematic, but it can impact the efficiency of the iterative estimation procedure, as we describe later. We can define a particular representation, termed the *maximal form* of the model. This has generators defined as the maximal sets  $\mathcal{A} \subseteq \Delta \cup \Gamma$  such that:

1.  $\mathcal{A} \cap \Delta$  is contained in a generator of  $g(i)$ ,
2. for each  $u \in \mathcal{A} \cap \Gamma$ ,  $\mathcal{A} \cap \Delta$  is contained in a generator of  $h^u(i)$ , and
3. for each  $x, y \in \mathcal{A} \cap \Gamma$ , with  $u \neq v$ ,  $k_{uv}$  is not set to be zero.

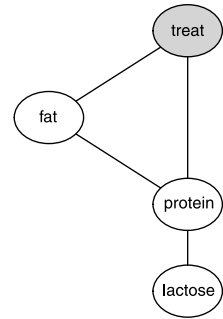
For example,  $I * X * W + X * W * Z$  is of maximal form but  $I * X * W + X * Z + W * Z$  is not.

The `mmod()` function in the **gRim** package allows MI-models to be defined using model formulae. For example, to define the model for the milk composition dataset with the conditional independences arrived at in Sect. 5.4, we specify the generating class with generators `{treat,fat,protein}` and `{protein,lactose}`, as follows:

```
> milkmod <- mmod(~treat*fat*protein + protein*lactose, data=milkcomp1)
```



**Fig. 5.1** Mixed interaction model for milk composition data. Discrete variables are shown as *grey* nodes while continuous variables are *white*



```

Model: A mModel with 4 variables
graphical : TRUE decomposable : TRUE
-2logL    :      428.47 mdim :    26 aic :      480.47
ideviance :      18.97 idf  :    14 bic :      532.66
deviance  :         2.11 df   :     7
  
```

The discrete, linear and quadratic generators of the model are

```

> str(milkmod$modelinfo$dIq)
List of 3
 $ discrete :List of 1
  ..$ : chr "treat"
 $ linear   :List of 2
  ..$ : chr [1:2] "fat" "treat"
  ..$ : chr [1:2] "protein" "treat"
 $ quadratic:List of 2
  ..$ : chr [1:2] "fat" "protein"
  ..$ : chr [1:2] "protein" "lactose"
  
```

To construct the dependence graph of an MI-model defined using such a formula, we connect with an edge all variable pairs appearing in the same generator. By convention, discrete variables are drawn with filled circles and continuous variables with hollow circles. The *global Markov property* (Sect. 1.3) can be used for reading conditional independencies from the dependence graph in the usual way. For example, the dependence graph for the model `milkmod` just discussed is shown in Fig. 5.1. It can be obtained using the `plot` function:

```

> plot(milkmod)
  
```

## 5.6 Graphical and Decomposable MI-models

Suppose we are given an undirected graph with vertex set  $\Delta \cup \Gamma$  and consider the MI-model for  $\Delta \cup \Gamma$  whose generators are the cliques of the graph. An MI-model that can be formed in this way is termed a *graphical MI-model*. Table 5.1 shows some graphical MI-models.

As with log-linear models, it is possible to set higher-order interactions to zero, without introducing new conditional independence relations. Such models are called

*non-graphical*. For example, consider model (b) in Table 5.1. Since the generators of the formula correspond to the cliques of the graph, the model is graphical. The model implies that the term  $h^y(i)$  is unrestricted, say as

$$h^y(i) = w + w_j^a + w_k^b + w_{jk}^{ab}.$$

If we constrain  $w_{jk}^{ab} = 0$ ,  $\forall j, k$ , then  $h^y(i)$  has the additive form  $h^y(i) = w + w_j^a + w_k^b$ ,  $\forall j, k$ . This does not correspond to a conditional independence restriction, but results in model (f) in Table 5.1. So model (f) is non-graphical. Since no further conditional independence restrictions have been added model (f) has the same dependence graph as model (b).

We now turn to a subclass of the graphical MI-models, the *decomposable MI-models*. These build on a more basic concept, that of a *decomposition*, which we describe first.

The notion of a decomposition of a graph  $\mathcal{G}$  with mixed variables relates to the question of how and when the analysis of a graphical MI-model may be broken down into analyses of smaller models. This notion is slightly more elaborate than in the purely discrete and purely continuous cases. Let  $A$ ,  $B$  and  $S$  be disjoint non-empty subsets of  $V$  such that  $A \cup B \cup S = V$ . We define  $(A, B, S)$  to be a *decomposition* of  $\mathcal{G}$  if the following conditions hold:

1.  $A$  and  $B$  are separated by  $S$  in  $\mathcal{G}$ ,
2.  $S$  is complete in  $\mathcal{G}$ , and
3.  $S \subset \Delta$  or  $B \subset \Gamma$ .

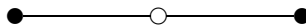
It can be shown that when  $(A, B, S)$  is a decomposition of  $\mathcal{G}$ , the maximum likelihood estimator  $\hat{f}$  of the density of the graphical MI-model with dependence graph  $\mathcal{G}$  is given by

$$\hat{f} = \frac{\hat{f}_{[A \cup S]} \hat{f}_{[B \cup S]}}{\hat{f}_{[S]}}$$

where  $\hat{f}_{[A \cup S]}$ ,  $\hat{f}_{[B \cup S]}$ ,  $\hat{f}_{[S]}$  are the estimates of densities based on the models corresponding to the relevant induced subgraphs and based on marginal data only. Indeed they are weak marginals of  $\hat{f}$ , see Sect. 5.7.5.1 below.

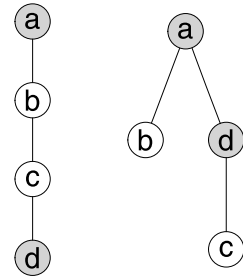
A graph with mixed variables  $\mathcal{G}$  is called decomposable if it is complete or it can be successively decomposed into complete graphs.

Various characterizations of graphs with this property are useful. One is based on the forbidden path property: a *forbidden path* is a path between two non-adjacent discrete vertices that passes through only continuous vertices. It can be shown that a graph is decomposable if and only if it is triangulated and has no forbidden paths. A simple example of a graph with mixed variables that is not decomposable is:



Another characterization is that the cliques of a decomposable graph with mixed variables can be ordered as  $(C_1, \dots, C_k)$  with a modified version of the *running in-*

**Fig. 5.2** Decomposable graphs with mixed variables. If  $a$  and  $d$  are discrete and  $b$  and  $c$  are continuous then the first graph is not decomposable whereas the second graph is



*tersection property.* For  $j > 1$  define  $H_j = \bigcup_{i=1}^{j-1} C_i$  and  $S_j = C_j \cap H_j$ . The modified condition is that

1. for each  $j > 1$ ,  $S_j \subset C_i$  for some  $i < j$ , and
2. for each  $j > 1$  it holds that  $C_j \setminus S_j \subseteq \Gamma$  or  $S_j \subseteq \Delta$ .

The additional condition (2) states that continuous variables cannot be prior to discrete ones. A graph with mixed variables is decomposable if and only there exists an ordering of its cliques fulfilling conditions (1) and (2).

A decomposable MI-model is a graphical MI-model whose dependence graph is decomposable. For such a model, the maximum likelihood estimates take the closed form

$$\hat{f}(x) = \prod_{j=1}^k \frac{\hat{f}_{[C_j]}(x_{C_j})}{\hat{f}_{[S_j]}(x_{S_j})} \quad (5.9)$$

where we have let  $S_1 = \emptyset$  and  $\hat{f}_{\emptyset} = 1$ .

To check whether a graph with mixed variables is decomposable, the so-called *star graph* construction can be used. That is, let  $\mathcal{G}^*$  be a new graph obtained by adding an extra vertex,  $\star$ , to  $\mathcal{G}$  and adding edges between  $\star$  and all discrete variables. Then  $\mathcal{G}^*$  is triangulated (which can be checked with maximum cardinality search) if and only if  $\mathcal{G}$  is decomposable.

It can also be shown that a graph  $\mathcal{G}$  with mixed variables is decomposable if and only if the vertices of  $\mathcal{G}$  can be given a *perfect ordering*. For such graphs this is defined as an ordering  $\{v_1, v_2, \dots, v_T\}$  such that (i)  $S_k = \text{ne}(v_k) \cap \{v_1, v_2, \dots, v_{k-1}\}$  is complete in  $\mathcal{G}$  and (ii)  $S_k \subset \Delta$  if  $v_k \in \Delta$ . The `mcsmarked()` function is based on constructing  $\mathcal{G}^*$  as described above and returns a perfect ordering if the graph is decomposable.

As an example consider the following two graphs shown in Fig. 5.2. If  $a$  and  $d$  are discrete and  $b$  and  $c$  are continuous then the graph on the left is not decomposable whereas the graph on the right is. Note that since a graph object contains no information about whether the nodes are discrete or continuous, `mcsmarked()` has to be supplied this information explicitly.

```

> uG1 <- ug(~a:b+b:c+c:d)
> uG2 <- ug(~a:b+a:d+c:d)
> mcsmarked(uG1, discrete=c("a", "d"))

```

```

character(0)
> mcsmarked(uG2, discrete=c("a", "d"))
[1] "a" "d" "b" "c"

```

## 5.7 Maximum Likelihood Estimation

In this section we derive expressions for the likelihood and describe algorithm(s) the maximize this.

### 5.7.1 Likelihood and Deviance

In this section we derive some expressions for the *likelihood* and the *deviance*. The log density can be written as

$$\log f(i, y) = \log p(i) - q \log(2\pi)/2 - \log \det(\Sigma)/2 - \{y - \mu(i)\}^\top \Sigma^{-1} \{y - \mu(i)\}/2,$$

so the log-likelihood of a sample  $(i^\nu, y^\nu)$ ,  $\nu = 1, \dots, N$  is

$$\begin{aligned} \ell = & \sum_i n(i) \log p(i) - Nq \log(2\pi)/2 - N \log \det(\Sigma)/2 \\ & - \sum_{\nu=1}^N \{y^\nu - \mu(i^\nu)\}^\top \Sigma^{-1} \{y^\nu - \mu(i^\nu)\}/2. \end{aligned}$$

We can simplify the last term using that

$$\begin{aligned} & \sum_i \sum_{\nu: i^\nu=i} \{y^\nu - \mu(i)\}^\top \Sigma^{-1} \{y^\nu - \mu(i)\} \\ & = N \operatorname{tr}(S\Sigma^{-1}) + \sum_i n(i) \{\bar{y}(i) - \mu(i)\}^\top \Sigma^{-1} \{\bar{y}(i) - \mu(i)\}. \end{aligned}$$

So an alternative expression for the log likelihood is

$$\begin{aligned} \ell = & \sum_i n(i) \log p(i) - Nq \log(2\pi)/2 - \sum_i n(i) \log \det(\Sigma)/2 \\ & - N \operatorname{tr}(S\Sigma^{-1})/2 - \sum_i n(i) \{\bar{y}(i) - \mu(i)\}^\top \Sigma^{-1} \{\bar{y}(i) - \mu(i)\}/2. \end{aligned}$$

The full homogeneous model has MLEs  $\hat{p}(i) = n(i)/N$ , (so that  $\hat{m}(i) = N\hat{p}(i)$ ),  $\hat{\mu}(i) = \bar{y}(i)$ , and  $\hat{\Sigma} = S = \sum_i n(i)S_i/N$ , so the maximized log likelihood for this model is

$$\hat{\ell}_s = \sum_i n(i) \log\{n(i)/N\} - Nq \log(2\pi)/2 - N \log \det(S)/2 - Nq/2, \quad (5.10)$$

and the deviance of a homogeneous model  $\mathcal{M}$  with MLEs  $\hat{p}(i)$ ,  $\hat{\mu}(i)$ , and  $\hat{\Sigma}$  with respect to the full homogeneous model simplifies to

$$D = 2 \sum_i n(i) \log\{n(i)/\hat{m}(i)\} - N \log \det(S\hat{\Sigma}^{-1}) + N\{\text{tr}(S\hat{\Sigma}^{-1}) - q\} \\ + \sum_i n(i) \{\bar{y}(i) - \hat{\mu}(i)\}^\top \hat{\Sigma}^{-1} \{\bar{y}(i) - \hat{\mu}(i)\}.$$

Note that in contrast to the models considered in Chap. 4, we do not necessarily have  $\text{tr}(S\hat{\Sigma}^{-1}) = q$  so the term  $N \log \det(S\hat{\Sigma}^{-1})$  does not disappear.

### 5.7.2 Dimension of MI-models

The dimension of a mixed interaction model may be simply calculated by adding the dimensions of the component models for  $g(i)$  and each  $h^u(i)$  to the number of free elements of the covariance matrix, and finally subtract one for the normalisation constant.

### 5.7.3 Inference

Under  $\mathcal{M}$ , the deviance  $D$  is asymptotically  $\chi^2(k)$  where the degrees of freedom  $k$  is the difference in dimension (number of free parameters) between the saturated model and  $\mathcal{M}$ . Similarly, for two nested models  $\mathcal{M}_1 \subseteq \mathcal{M}_2$ , the deviance difference  $D_1 - D_2$  is asymptotically  $\chi^2(k)$  where the degrees of freedom  $k$  is the difference in dimension (number of free parameters) between the two models.

### 5.7.4 Likelihood Equations

Suppose we have a sample of  $N$  independent, identically distributed observations  $(i^v, y^v)$  for  $v = 1 \dots N$ . Let  $(n(i), t(i), \bar{y}(i))_{i \in \mathcal{I}}$  be the observed counts, variate totals and variate means, for cell  $i$ , and  $SS$  and  $S$  be the uncorrected sums of squares and sample covariance matrices, i.e.,

$$n(i) = \#\{v : i^v = i\},$$

$$\begin{aligned}
t(i) &= \sum_{v:i^v=i} y^v, \\
\bar{y}(i) &= t(i)/n(i), \\
SS &= \sum_v y^v (y^v)^\top, \\
SSD &= \sum_{i \in \mathcal{I}} \sum_{v:i^v=i} \{y^v - \bar{y}(i)\} \{y^v - \bar{y}(i)\}^\top = SS - n(i) \sum_{i \in \mathcal{I}} \bar{y}(i) \{\bar{y}(i)\}^\top \\
S &= SSD/N
\end{aligned}$$

For  $a \subseteq \Delta$ , we write the marginal cell corresponding to  $i$  as  $i_a$  and likewise for  $b \subseteq \Gamma$ , we write the subvector of  $y$  as  $y_b$ . Similarly, we write the marginal cell counts as  $\{n(i_a)\}_{i_a \in \mathcal{I}_a}$ , marginal variate totals as  $\{t^b(i_a)\}_{i_a \in \mathcal{I}_a}$  and marginal variate means as  $\{\bar{y}_b(i_a)\}_{i_a \in \mathcal{I}_a}$ . Define

$$SSD_a^b(i_a) = \sum_{v:i_a^v=i_a} \{y_b^k - \bar{y}_b(i_a)\} \{y_b^k - \bar{y}_b(i_a)\}^\top$$

and let

$$SSD_a^b = \sum_{i_a \in \mathcal{I}_a} SSD_a^b(i_a) = SS^b - \sum_{i_a \in \mathcal{I}_a} n(i_a) \bar{y}_b(i_a) \bar{y}_b(i_a)^\top$$

where  $SS^b$  is the  $b$ -submatrix of the sums-of-squares matrix  $SS$ .

The log-likelihood for the sample is

$$\begin{aligned}
l &= \sum_{(a,b) \in \mathcal{C}} \sum_{i_a \in \mathcal{I}_a} n(i_a) g_a(i_a) + \sum_{(a,b) \in \mathcal{C}} \sum_{i_a \in \mathcal{I}_a} h_a^b(i_a)^\top t^b(i_a) \\
&\quad - \sum_{u \in \Gamma} SS^{uu} k_{uu}/2 - \sum_{\{u,v\} \in \Gamma} SS^{uv} k_{uv}
\end{aligned} \tag{5.11}$$

where in the last term there is a contribution from  $SS^{uv}$  only if  $k_{uv} \neq 0$ , that is if  $\{u, v\} \in b$  for some generator  $(a, b) \in \mathcal{C}$ .

Consider now a given model with generators  $\mathcal{C} = \{G_1, \dots, G_m\}$  and derive the formulae for  $g(i)$  and each  $h^u(i)$  as described in Sect. 5.5. Then a set of *minimal sufficient statistics* is given by

1. A set of marginal tables of cell counts  $\{n(i_a)\}_{i_a \in \mathcal{I}_a}$  for each discrete generator  $a$ .
2. For each  $u \in \Gamma$ , a set of marginal variate totals  $\{t^u(i_a)\}_{i_a \in \mathcal{I}_a}$  for each linear generator  $a$  of  $u$ .
3. A set of marginal tables of uncorrected sums and squares  $\{SS^b\}$  for each quadratic generator  $b$ .

From exponential family theory, we know that the MLE of  $\{p(i), \mu(i), \Sigma\}$  can be found by equating the expectations of these minimal sufficient statistics to their observed values. Equating the minimal sufficient statistics to their observed values for a generator  $(a, b)$  yields:

$$n(i_a) = Np(i_a), \quad \forall i_a \in \mathcal{I}_a, \quad (5.12)$$

$$t^b(i_a) = N \sum_{j:j_a=i_a} p(j)\mu^b(j), \quad \forall i_a \in \mathcal{I}_a \quad (5.13)$$

$$SS^b = N \left\{ \Sigma^b + \sum_{j \in \mathcal{I}} p(j)\mu^b(j)\mu^b(j)^\top \right\}. \quad (5.14)$$

Each generator  $(a, b) \in \mathcal{C}$  defines a set of equations of the form (5.12)–(5.14) and the collection of these equations are the *likelihood equations* for the model. The MLEs, when they exist, are the unique solution to these equations that also satisfy the model constraints.

For example, for the saturated model on  $V = \Delta \cup \Gamma$ , we set  $a = \Delta$  and  $b = \Gamma$ . Here there are no model constraints, and from the equations we find that the MLEs are given as  $\hat{p}(i) = n(i)/N$ ,  $\hat{\mu}(i) = \bar{y}(i)$  and  $\hat{\Sigma} = S$ .

### 5.7.5 Iterative Proportional Scaling

As with discrete log-linear models and graphical Gaussian models, iterative methods to find the maximum likelihood parameter estimates are generally necessary. The iterative proportional scaling algorithm for mixed interaction models proceeds by equating observed and expected margins, in much the same way as with discrete and continuous models. An important conceptual difference, however, relates to marginalization. Whereas multinomial and Gaussian distributions are preserved under marginalization, the same is not generally true in the mixed case: the marginal distribution of a CG-distribution is not necessarily CG. For this reason the concept of *weak marginals* is needed.

#### 5.7.5.1 Weak Marginals

Consider a CG-density  $f_V$  defined over the variables  $V = \Delta \cup \Gamma$ . Letting  $a \subset \Delta$  and  $b \subset \Gamma$  we wish to obtain the marginal density  $f_{a \cup b}$ . This density is obtained by first integrating over  $y_{\Gamma \setminus b}$  to produce  $f_{\Delta \cup b}$  which again is a CG-density. The next step is to sum over  $i_{\Delta \setminus a}$  to form  $f_{a \cup b}$ . This summation may involve forming a mixture of normal densities, which does not generally have the form of a CG-density. However, even though  $f_{a \cup b}$  is not in general a CG-density we can find the moments of  $f_{a \cup b}$  using standard formulae, namely

$$p_{[a]}(i_a) = p(I_a = i_a) = p(i_a) = \sum_{j:j_a=i_a} p(j)$$

$$\mu_{[a]}^b(i_a) = E(Y^b | I_a = i_a) = \sum_{j:j_a=i_a} \frac{p(j)}{p_{[a]}(i_a)} \mu^b(j), \quad \text{and}$$

$$\begin{aligned}\Sigma_{[a]}^b(i_a) &= \mathbf{V}(Y^b | I_a = i_a) \\ &= \Sigma^b + \sum_{j:j_a=i_a} \frac{p(j)}{p_{[a]}(i_a)} \{\mu^b(j) - \mu_{[a]}^b(i_a)\} \{\mu^b(j) - \mu_{[a]}^b(i_a)\}^\top.\end{aligned}$$

These moments  $\{p_{[a]}(i_a), \mu_{[a]}^b(i_a), \Sigma_{[a]}^b(i_a)\}_{i_a \in \mathcal{I}_a}$  define a CG density  $f_{[a \cup b]}$  denoted the *weak marginal density* (which is not homogeneous).

Furthermore, we define the *homogeneous weak marginal variance* to be:

$$\begin{aligned}\Sigma_{[a]}^b &= \sum_{i_a \in \mathcal{I}_a} p_{[a]}(i_a) \Sigma_{[a]}^b(i_a) \\ &= \Sigma^b + \sum_{i_a \in \mathcal{I}_a} \sum_{j:j_a=i_a} p(j) \{\mu^b(j) - \mu_{[a]}^b(i_a)\} \{\mu^b(j) - \mu_{[a]}^b(i_a)\}^\top.\end{aligned}$$

The moments  $\{p_{[a]}(i_a), \mu_{[a]}^b(i_a), \Sigma_{[a]}^b(i_a)\}_{i_a \in \mathcal{I}_a}$  define a CG density  $f_{[a \cup b]}^h$  which is denoted the *homogeneous weak marginal density*.

The weak marginal density is the CG-density which best approximates the true marginal  $f_{a \cup b}$  in the sense of minimizing the Kullback–Leibler distance, see Lauritzen (1996), p. 162. The same proof yields that the analogous statement holds for the homogeneous weak marginal.

### 5.7.5.2 Likelihood Equations Revisited

It is illustrative to rewrite the likelihood equations as follows. Observe that

$$\begin{aligned}Q_a^b &= \sum_{i_a \in \mathcal{I}_a} \sum_{j:j_a=i_a} p(j) \{\mu^b(j) - \mu_{[a]}^b(i_a)\} \{\mu^b(j) - \mu_{[a]}^b(i_a)\}^\top \\ &= \sum_{i \in \mathcal{I}} p(i) \mu^b(i) \{\mu^b(i)\}^\top - \sum_{i_a \in \mathcal{I}_a} p_{[a]}(i_a) \mu_{[a]}^b(i_a) \{\mu_{[a]}^b(i_a)\}^\top\end{aligned}\quad (5.15)$$

Using the definitions of the parameters of weak marginal models, (5.12) and (5.13) imply that

$$n(i_a)/N = p_{[a]}(i_a), \quad \bar{y}^b(i_a) = t^b(i_a)/n(i_a) = \mu_{[a]}^b(i_a).\quad (5.16)$$

Using (5.15) and (5.16) we get from (5.14) that

$$\begin{aligned}SSD_a^b &= SS^b - \sum_{i_a \in \mathcal{I}_a} n(i_a) \bar{y}^b(i_a) \bar{y}^b(i_a)^\top \\ &= N \left[ \Sigma^b + \sum_{i_a \in \mathcal{I}_a} \sum_{j:j_a=i_a} p(j) \{\mu^b(j) - \mu_{[a]}^b(i_a)\} \{\mu^b(j) - \mu_{[a]}^b(i_a)\}^\top \right] \\ &= N(\Sigma^b + Q_a^b) = N \Sigma_{[a]}^b\end{aligned}$$



The MLEs under the saturated MI-model for the variables  $a \cup b$  (whose density is denoted  $\tilde{f}_{a \cup b}$ ) are  $\{\tilde{p}_a(i_a), \tilde{\mu}_a^b(i_a), \tilde{\Sigma}_a^b\}_{i_a \in \mathcal{I}_a}$  where

$$\tilde{p}_a(i_a) = n(i_a)/N, \quad \tilde{\mu}_a^b(i_a) = \bar{y}^b(i_a) \quad \text{and} \quad \tilde{\Sigma}_a^b = SSD_a^b/N.$$

In other words, the likelihood equations are:

$$\tilde{p}_a(i_a) = n(i_a)/N = p_{[a]}(i_a) \quad (5.17)$$

$$\tilde{\mu}_a^b(i_a) = \bar{y}^b(i_a) = \mu_{[a]}^b(i_a) \quad (5.18)$$

$$\tilde{\Sigma}_a^b = SSD_a^b/N = \Sigma_{[a]}^b \quad (5.19)$$

thus the homogeneous weak marginal model on  $a \cup b$  should be identical to the saturated MI-model on  $a \cup b$ , i.e.  $f_{[a \cup b]}^h = \tilde{f}_{a \cup b}$ .

### 5.7.5.3 General IPS Update Step

Here we describe the iterative algorithm for general MI-models implemented in **gRim** and MIM (Edwards 2000). Equations (5.17)–(5.19) suggest the following IPS update step for a generator  $(a, b)$ :

$$f^*(i, y) \propto f(i, y) \frac{f_{a \cup b}^{\text{sat}}(i_a, y_b)}{f_{[a \cup b]}^h(i_a, y_b)} \quad (5.20)$$

Note that the right-hand side of (5.20) will not in general be a density: Integrating over  $y_{\Gamma \setminus b}$  and summing over  $i_{\Delta \setminus a}$  gives

$$f_{a \cup b}(i_a, y^b) f_{a \cup b}^{\text{sat}}(i_a, y^b) / f_{[a \cup b]}^h(i_a, y^b)$$

which will not be a density unless the marginal density  $f_{a \cup b}(i_a, y_b)$  equals the homogeneous weak marginal density  $f_{[a \cup b]}^h(i_a, y_b)$ .

It is convenient to perform the update (5.20) on log-densities using the canonical parametrisation, since it just involves to addition and subtraction of canonical parameters. From (5.17)–(5.19), to update  $(g, h, K)$  we first transform the moment parameters  $\{\tilde{p}_a, \tilde{\mu}_a^b, \tilde{\Sigma}_a^b\}$  and  $\{p_{[a]}, \mu_{[a]}^b, \Sigma_{[a]}^b\}$  of  $\tilde{f}_{a \cup b}$  and  $f_{[a \cup b]}^h$  to canonical parameters  $(\tilde{g}_a, \tilde{h}_a^b, \tilde{K}_a^b)$  and  $(g_{[a]}, h_{[a]}^b, K_{[a]}^b)$ . Then we

1. Update  $g$  as follows: For each  $i_a \in \mathcal{I}_a$  do for all  $j$  for which  $j_a = i_a$ :

$$g(j) \leftarrow g(j) + \{\tilde{g}_a(i_a) - g_{[a]}(i_a)\}. \quad (5.21)$$

2. Update the  $b$  subvector of  $h$  as follows: For each  $i_a \in \mathcal{I}_a$  do for all  $j$  for which  $j_a = i_a$ :

$$h^b(j) \leftarrow h^b(j) + \{\tilde{h}_a^b(i_a) - h_{[a]}^b(i_a)\}. \quad (5.22)$$

3. Update the  $b$  submatrix  $K^{bb}$  of  $K$  as follows:

$$K^{bb} \leftarrow K^{bb} + \{\tilde{K}_a^b - K_{[a]}^b\}. \quad (5.23)$$

After the update steps (5.21)–(5.23) we know  $h$  and  $K$  and hence the conditional distribution of  $y$  given  $i$ . To complete the update we must transform  $(g, h, K)$  to moment form  $(p, \mu, \Sigma)$ , normalize  $p$  to sum to one and transform back to canonical form  $(g, h, K)$  again before moving on to the next generator. Running through the generators  $(a_1, b_1), (a_2, b_2), \dots, (a_M, b_M)$  as described above constitutes one cycle of the iterative fitting process.

A measure of how much the updates (5.21)–(5.23) change the parameter estimates may be obtained by comparing the moments of  $\tilde{f}_{a \cup b}$  and  $f_{[a \cup b]}^h$ . Following Edwards (2000) we use the quantity:

$$\text{mdiff}(a, b) = \max_{i_a \in \mathcal{I}_a, u, v \in b} \left\{ \frac{N|p_{[a]}(i_a) - \tilde{p}_a(i_a)|}{\sqrt{Np_{[a]}(i_a) + 1}}, \frac{|\mu_{[a]}^u(i_a) - \tilde{\mu}_a^u(i_a)|}{\sqrt{(\Sigma_{[a]}^b)_{uu}}}, \right. \\ \left. \frac{|(\Sigma_{[a]}^b)_{uv} - (\tilde{\Sigma}_a^b)_{uv}|}{\sqrt{(\Sigma_{[a]}^b)_{uu}(\Sigma_{[a]}^b)_{vv} + (\Sigma_{[a]}^b)_{uv}^2}} \right\} \quad (5.24)$$

It sometimes happens that the updates (5.21)–(5.23) lead to a decrease in the likelihood. To avoid this situation we first calculate  $\text{mdiff}(a, b)$  in (5.24). If  $\text{mdiff}(a, b)$  is smaller than some prespecified criterion we do not update the model but proceed to the next generator. If this is true for all generators we exit the iterative process, as it essentially only happens when we are close to the MLE.

Since the estimation algorithm in the `mmod()` function is based on the model formula, which is not unique, there will be efficiency differences between the different representations of the same model. The maximal form is the most efficient.

#### 5.7.5.4 Step-Halving Variant

It can happen that the updates (5.21)–(5.23) fail to increase the likelihood, or lead to a  $K$  that is not positive definite. The step-halving variant of the algorithm (currently not implemented in `gRim`) replaces the three update steps in (5.21)–(5.23) with:

$$g(j) \leftarrow g(j) + \kappa\{\tilde{g}_a(i_a) - g_{[a]}(i_a)\}, \\ h^b(j) \leftarrow h^b(j) + \kappa\{\tilde{h}_a^b(i_a) - h_{[a]}^b(i_a)\}, \\ K^{bb} \leftarrow K^{bb} + \kappa\{\tilde{K}_a^b - K_{[a]}^b\}.$$

Initially  $\kappa = 1$ . The update is attempted and it is then checked if (1)  $K$  is positive definite and (2) the likelihood is increased. If either of these conditions fail,  $\kappa$  is halved and the update is attempted again. The step-halving variant is therefore slower than the general algorithm. Edwards (2000, p. 312) shows an example with contrived data where step-halving is necessary.

### 5.7.5.5 Mixed Parameterisation Variant

If the model is collapsible onto the discrete parameters, the estimate  $\hat{p}(i)$  is identical to the estimate obtained in the log-linear model with the same discrete generator. This permits another variant based on the mixed parametrisation to be used. It has the following update scheme

$$\begin{aligned} p(j) &\leftarrow p(j)\{p(i_a)/p_{[a]}(i_a)\}, \\ h^b(j) &\leftarrow h^b(j) + \kappa\{\tilde{h}_a^b(i_a) - h_{[a]}^b(i_a)\}, \\ K^{bb} &\leftarrow K^{bb} + \kappa\{\tilde{K}_a^b - K_{[a]}^b\}. \end{aligned}$$

The model is collapsible onto  $\Delta$  if and only every connected component of the sub-graph induced by the continuous variables has a complete boundary in the subgraph induced by the discrete variables (Frydenberg 1990b). This variant is currently not implemented in **gRim**.

## 5.8 Using **gRim**

The function `mmod()` in the **gRim** package allows homogeneous mixed interaction models to be defined and fitted to data.

```
> glist <- ~treat:fat:protein+protein:lactose
~treat:fat:protein + protein:lactose
> milk <- mmod(glist, data=milkcomp1)
Model: A mModel with 4 variables
graphical : TRUE decomposable : TRUE
-2logL    :      428.47 mdim :    26 aic :      480.47
ideviance :      18.97 idf  :    14 bic :      532.66
deviance  :       2.11 df   :     7
```

This model is shown in Fig. 5.1. More details about the model are obtained with

```
> summary(milk)
Mixed interaction model:
Generators:
  : "treat" "fat" "protein"
  : "protein" "lactose"
Discrete: 1 Continuous: 3
Is graphical: TRUE Is decomposable: TRUE
logL: -214.233011, iDeviance: 241.774364
```

The parameters are obtained using `coef()` where the desired parameterization can be specified. For example, the canonical parameters are

```
> coef(milk, type="ghk")
```

```

MIparms: form=ghk
      a          b          c          d          e          f
[1,] -676.055 -666.0859 -675.0546 -690.992 -664.9730 -666.7805
[2,]  -1.135  -0.2838  -0.9179  -1.022  -0.2012  -0.5375
[3,]  84.349  81.3414  83.8954  86.851  81.0040  81.8196
[4,] 164.634 164.6335 164.6335 164.634 164.6335 164.6335
      g
[1,] -680.022      NA      NA      NA
[2,]  -1.043  0.5026 -0.815  0.000
[3,]  84.953 -0.8150 10.762  5.667
[4,] 164.634  0.0000  5.667 24.646

```

### 5.8.1 Updating Models

Models are changed using the `update()` method. A list with one or more of the components `add.edge`, `drop.edge`, `add.term` and `drop.term` is specified. The updates are made in the order given. For example:

```

> milk2 <- update(milk, list(add.edge=~fat:lactose,
                             drop.edge=~treat:protein))

Model: A mModel with 4 variables
graphical : TRUE decomposable : TRUE
-2logL    :      446.17 mdim :    21 aic :      488.17
ideviance :      10.12 idf  :     9 bic :      530.33
deviance  :       10.96 df   :    12

```

### 5.8.2 Inference

Functions such as `ciTest()`, `testInEdges()`, `testOutEdges()`, etc. behave more or less as for pure discrete and pure continuous variables. For example

```

> ciTest(milkcomp1)

Testing treat _|_ fat | protein dm lactose
Statistic (DEV):    4.371 df: 6 p-value: 0.6266 method: CHISQ

```

and

```

> testInEdges(milk, getInEdges(milk$glist))

  statistic df p.value    aic    V1    V2 action
1     5.530  6 0.47780 -6.470   fat  treat    +
2     9.345  6 0.15510 -2.655 protein treat    +
3     4.139  1 0.04191  2.139 protein   fat     -
4     5.123  1 0.02362  3.123 lactose protein -

> testOutEdges(milk, getOutEdges(milk$glist))

  statistic df p.value    aic    V1    V2 action
1     1.9464 6  0.9246 10.054 lactose treat    -
2     0.4914 1  0.4833  1.509 lactose   fat     -

```

or

```
> milk3 <- update(milk, list(drop.edge=~treat:protein))

Model: A mModel with 4 variables
graphical : TRUE decomposable : TRUE
-2logL    :          447.16 mdim :    20 aic :          487.16
ideviance :           9.63 idf  :     8 bic :          527.30
deviance  :           11.45 df   :    13

> compareModels(milk, milk3)

Large:
:"treat" "fat" "protein"
:"protein" "lactose"
Small:
:"protein" "lactose"
:"treat" "fat"
:"fat" "protein"
-2logL:    18.69 df: 6 AIC(k= 2.0):    6.69 p.value: 0.155100

and

> testdelete(milk, c("treat", "protein"))

dev:    9.345 df: 6 p.value: 0.15510 AIC(k=2.0):  -2.7 edge:
      treat:protein
Notice: Test perfomed by comparing likelihood ratios

> testadd(milk, c("treat", "lactose"))

dev:    1.946 df: 6 p.value: 0.92456 AIC(k=2.0):  10.1 edge:
      treat:lactose
Notice: Test perfomed by comparing likelihood ratios
```

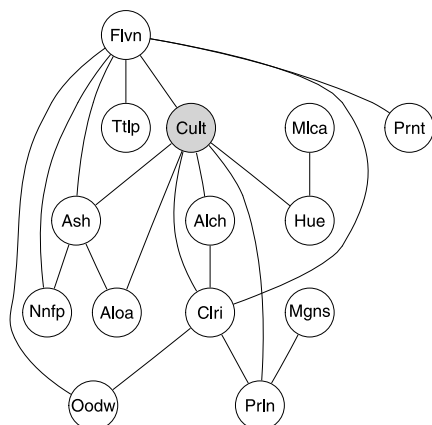
### 5.8.3 Stepwise Model Selection

The `stepwise()` function in the **gRim** package implements stepwise selection for mixed interaction models. The functionality is very similar to that described above in Sect. 2.4 and Sect. 4.4.1, for discrete graphical models and undirected graphical Gaussian models respectively. We refer to those sections for further details, and illustrate using the wine dataset described in Sect. 5.2. We start from the saturated model and use the BIC criterion:

```
> data(wine, package='gRbase')
> mm <- mmod(~.^. , data=wine)
> mm2 <- stepwise(mm, k=log(nrow(wine)), details=0)
```

The selected model is shown below:

```
> plot(mm2)
```



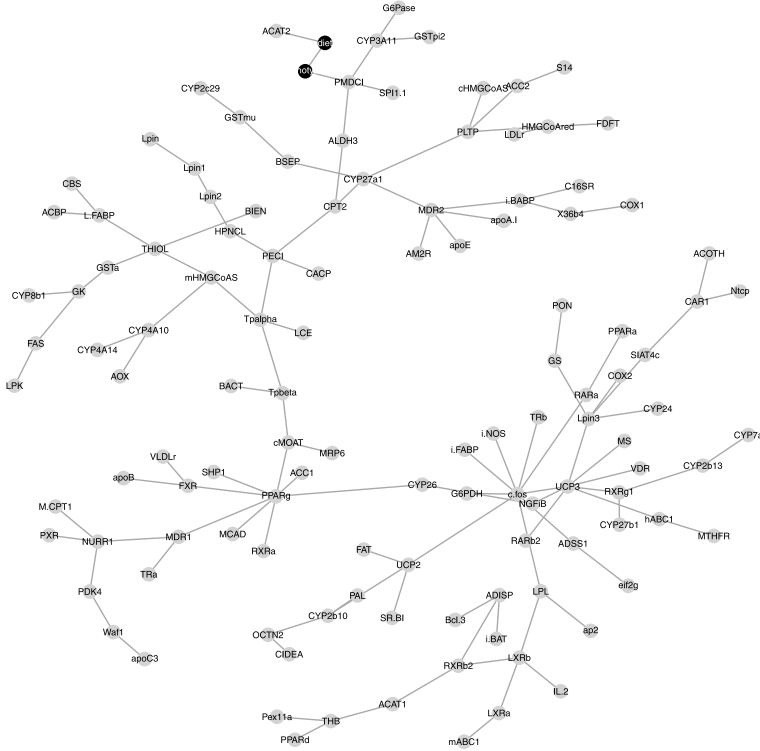
We note that the model is non-decomposable, since there are several chordless four-cycles in the graph. Since the graph is connected, it appears that all constituents differ over the grape varieties. Seven constituents are adjacent to the discrete variable. The model implies that these seven are sufficient to predict grape variety, since the remaining six are independent of variety given the seven, and so would not increase predictive ability.

## 5.9 An Example of Chain Graph Modelling

In this section we illustrate an approach that is appropriate when there is a clear overall response to the data, that is, when some variables are prior or explanatory to others, that are themselves prior or explanatory to others, and so on. The variables can *a priori* be divided into blocks, whose mutual ordering in this respect is clear. The goal of the analysis is to model the data, respecting this ordering between blocks, but not assuming any ordering within blocks. Chain graph models fit this purpose well.

The `Nutrimouse` dataset described above in Sect. 5.2 is here used as example. Here, the variables fall into three blocks: two discrete design variables (genotype and diet), 120 gene expression variables, and 21 lipid measurements. Clearly the design variables, which are subject to the control of the experimenter, are causally prior to the others. It is also natural as a preliminary working hypothesis to suppose that the gene expression measurements are causally prior to the lipid measurements, and this is the approach taken here. More advanced methods would be necessary to study whether there is evidence of influence in the opposite direction.

The chain graph is constructed using two graphical models: the first is for the gene expressions (block 2) given the design variables (block 1), and the second is for the lipids (block 3) given blocks 1 and 2. We use the **gRapHD** package described in Chap. 7. This package supports decomposable mixed models, both homogeneous and heterogeneous, exploiting the closed-form expressions for the MLEs (5.9). This restriction also means that models can simply be specified as graphs, rather than using model formulae.



**Fig. 5.3** A tree model for the gene expression variables (block 2) given the design variables (block 1)

To model the conditional distribution of block 2 given block 1 we restrict attention to models in which block 1 is complete, that is, there is an edge between the two design variables. See Fig. 4.33. The following code first finds the minimal BIC forest containing this edge, and then uses this as initial model in a forward selection process to find the minimal BIC decomposable model. This takes a few seconds.

```

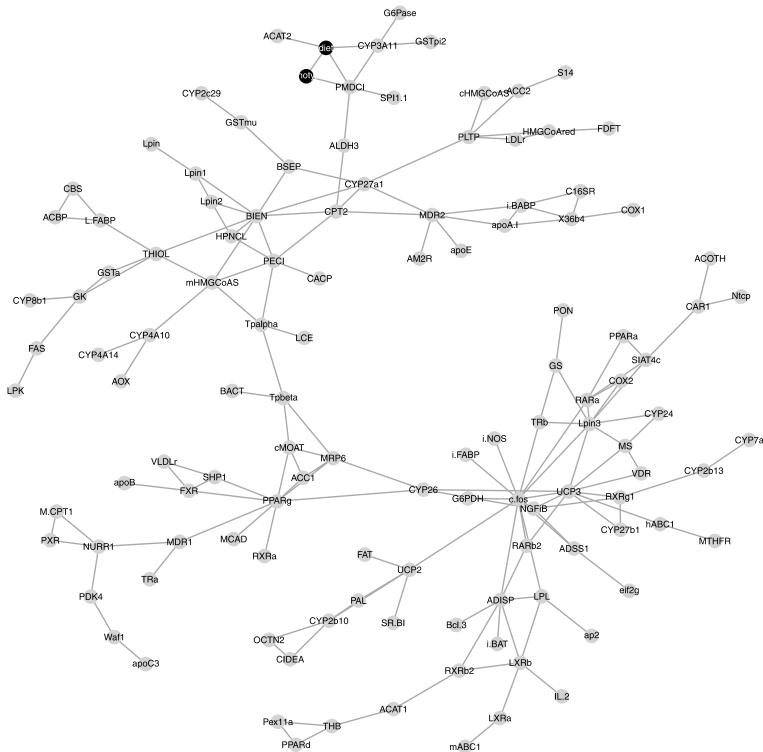
> data(Nutrimouse, package='gRbase')
> library(gRapHD)
> block2 <- Nutrimouse[,1:122]
> gF1 <- minForest(block2, cond=list(1:2))
> gD1 <- stepw(gF1, data=block2)

> xyD1 <- plot(gD1, numIt=5000, disp=F)
> plot(gF1, numIt=0, coord=xyD1)

> plot(gD1, numIt=0, coord=xyD1)
    
```

We display the two graphs in Figs. 5.3 and 5.4, using the same vertex coordinates for clarity. The vertex coordinates are saved in a matrix xyD1.

We now turn to modelling the conditional distribution of block 3 variables given the prior blocks. We adopt the same approach as before, first finding a minimal BIC



**Fig. 5.4** A decomposable model for the gene expression variables (block 2) given the design variables (block 1)

forest and then using this as start model in a forward selection process. As before we restrict the search space to conditional models by including all edges between prior variables in the models considered. The forward selection process seeks the decomposable MI-model with minimum BIC in this search space.

```
> gF2 <- minForest(Nutrimouse, cond=list(1:122))
> gD2 <- stepw(gF2, data=Nutrimouse)
```

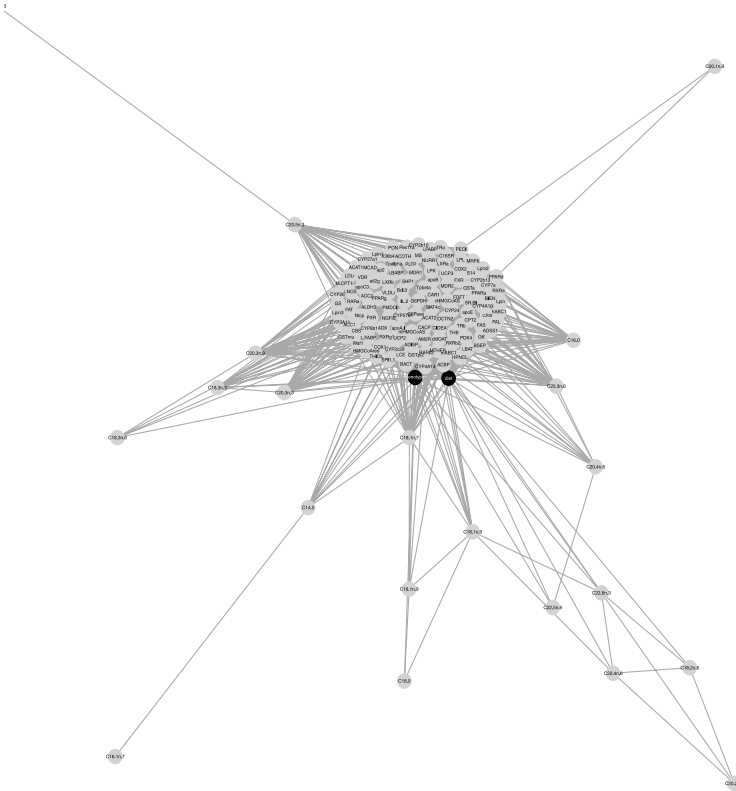
The `stepw()` function is computationally intensive, taking around 10 minutes on an ordinary laptop running Windows. We display the decomposable model in Fig. 5.5.

```
> plot(gD2, numIt=1000)
```

Now we construct a graph `gD3` by adding to `gD1` those edges in `gD2` that have a vertex in block 3:

```
> E2 <- data.frame(gD2@edges)
> names(E2) <- c("v1", "v2")
> E2 <- as.matrix(E2[(E2$v1>122) | (E2$v2>122),])
> E3 <- rbind(gD1@edges, E2)
> gD3 <- gD2
> gD3@edges <- E3
```

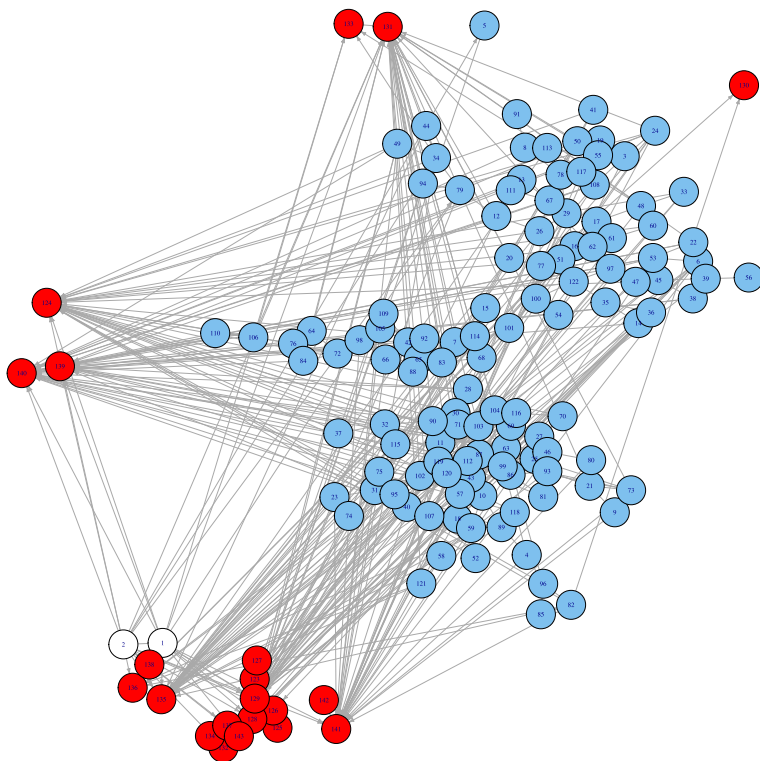




**Fig. 5.5** A decomposable model for the lipid variables given the gene expression and design variables. The subgraph induced by the gene expression and design variables is complete, and is shown as a compact splot

Note that `gD3` is an undirected graph rather than a chain graph. We use the **igraph** package to display it as a chain graph using different colours for the blocks, interblock edges displayed as arrows, in a layout in which the different blocks are separated for clarity. See Fig. 5.6

```
> # Define the blocks
> blk <- c(rep(1,2),rep(2,120),rep(3,21))
> # Derive the layout from the graph with only intrablock edges
> E <- gD3@edges
> E1 <- cbind(blk[E[,1]],blk[E[,2]])
> intrablock <- E1[,1]==E1[,2]
> tG3 <- gD3; tG3@edges <- E[intrablock,]
> itG3 <- as(as(tG3, "graphNEL"),"igraph")
> xy.coord <- piecewise.layout(itG3)
> # Use this for the chain graph
> igD3 <- graph.edgelist(E-1, directed=T)
> V(igD3)$label <- as.character(1:143)
> V(igD3)[blk==1]$color <- "white"
> V(igD3)[blk==2]$color <- "SkyBlue2"
```



**Fig. 5.6** A chain graph model for the nutr mouse data. The design variables are shown as *open circles*, the gene expression variables as *blue circles*, and the lipid variables as *red circles*. The variables are shown as column numbers

```
> V(igD3)[blk==3]$color <- "red"
> V(igD3)$size <- 8
> V(igD3)$label.cex <- 0.5
> E(igD3)[intrablock]$arrow.mode <- "-"
> E(igD3)[!intrablock]$arrow.mode <- "->"
> E(igD3)$arrow.size <- 0.3
> plot(igD3, layout=xy.coord)
```

## 5.10 Various

Several other R packages are designed for graphical modelling with mixed discrete and Gaussian variables. The package **CoCo** (Badsberg 1991) implements undirected graphical (and hierarchical) models with mixed variables. The package **deal** (Bøttcher and Dethlefsen 2003) allows a Bayesian analysis using models for mixed

variables based on DAGs, based on the conditional Gaussian distribution. Prior distributions for the model parameters are set and posterior distributions given data are derived. A heuristic search strategy for structural learning is also supported. The package **RHugin** also supports the use of Bayesian network models with mixed variables: see Chap. 3.