How to use gRchain

Thomas Rusch

2015-06-29

This is a short document intended to introduce the reader into the usage of the gRchain package. Currently, it supports block recursive chain graph models (Cox & Wermuth, 1996) by via the main function `coxwer()`.

Introduction

In complex research problems that involve a large number of potentially important variables or feature a complicated dependence structure, the joint probability distribution of the involved random variables can only be unsatisfactorily modelled with classical statistical models. Often some structure in the joint distribution allows to factorize it into conditionally independent components which gave rise to a class of models known as Graphical Models. By representing the joint distribution as a graph with nodes and edges, Graphical Models can exploit possible conditional independence structures in the joint distribution and allow restoring the joint distribution from the components.

A subclass of Graphical Models, Chain Graph Models, can be of particular interest for problems. A chain graph is a graph which may have both directed and undirected edges but is devoid of any directed cycles. In these models, a researcher can use substantive knowledge to categorize the variables as purely explanatory (predictor), purely dependent or target (response) or intermediate (response and predictor in turn). Each of these variables is assigned to a certain block, based on a partial ordering of the variables, meaning that the ordering is present between blocks but not within blocks. This approach leads to a chain of relationships between the different variables.

The challenging task of fitting a full Chain Graph Model to the data is facilitated by the factorization property which allows maximizing the joint likelihood by reducing the problem to maximizing the likelihood for each factorized submodel. However, in case of different variable types in the same block (e.g., metric and categorical variables) ML estimation using the direct factorization strategy often does not converge or can computationally be very expensive. As a remedy, Cox & Wermuth (1996) propose the heuristic usage of a system of univariate models for each factorized component.

In this vignette we introduce an R implementation of the Cox-Wermuth selection strategy that allows to fit a Chain Graph Model for metric and categorical random variables from exponential families and thus incorporates the class of Generalized Linear Models in the chain. For illustration we apply the procedure to a highly multivariate data set that features many different types of variables.

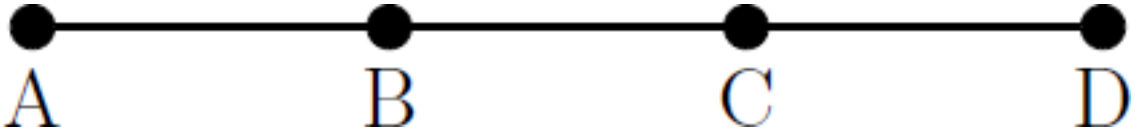
Graphical Models

Graphical models (GM) allow multivariate analysis of complex dependency structures. They are probability distributions over a multidimensional space encoded by graphs (as a set of vertices/variables, V, and a set of edges/relationships between variables, E). There are different types:

- Undirected GM (e.g., Markov random fields)
- Directed GM (e.g., Bayesian Networks, DAG)
- Chain GM

GM represent multivariate dependencies by conditional dependence and independence statements. Thus they can help in reducing overall complexity and allow model formulation, identification and selection.

A simple graphical model (a Markov random field):



In GM the Markov property of graphs allows to factorize the distribution F_V into a set of conditional distributions, e.g., for $V = \{A, B, C, D\}$ by way of densities: $f_V = f_{A|B} \times f_{B|C} \times f_{C|D} \times f_D$. Thus the problem of fitting graphical models effectively reduces to estimating a series of conditional distributions.

Chain Graph Models

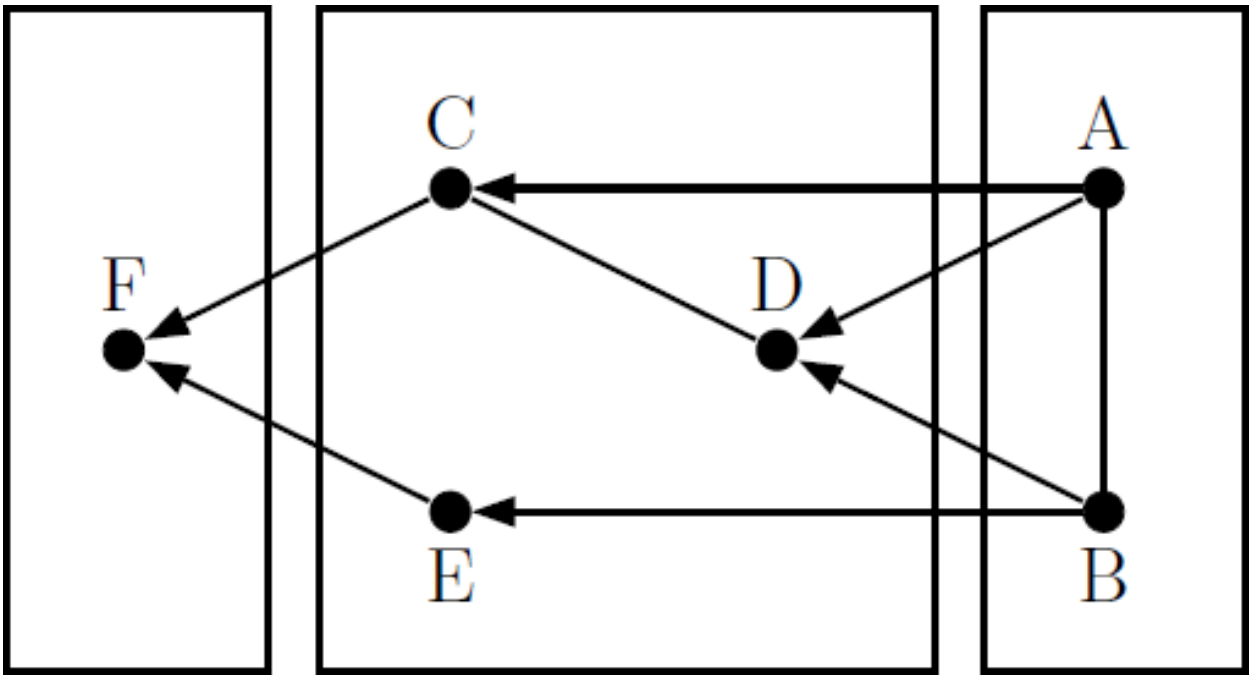
Chain graph models (CGM) are a mixture of directed and undirected graphical models. They are particularly interesting for social and behavioral sciences (observational studies, complex multivariate dependencies, existing substantive knowledge). In CGM, all variables are assigned to **boxes** (disjoint variable subsets $V_t, V = \bigcup_t V_t$ by theory or substantive knowledge. Between boxes exist **directed** edges, within boxes the edges are **undirected**,

Two types of CGM:

- Univariate recursive regression graph model (URRG; one variable per block)
- Joint response chain graph model (JRCG; more than one variable per block)

Factorization

A joint response chain graph model:



In CGM factorization happens at least **recursively between blocks**: $f_V = f_{V_T|V_{T-1},\ldots,V_1} \times f_{V_{T-1}|V_{T-2},\ldots,V_1} \times \cdots \times f_{V_1}$. Possibly additional conditional independence by missing edges, e.g., for the above graph $f_V = f_{F|C,E,D,A,B} \times f_{C,E,D|A,B} \times f_{A,B} = f_{F|C,E} \times f_{C,D|A,B} \times f_{E|B} \times f_{A,B}$.

Estimation

For CGM there are no theoretical restrictions on the form of the conditional distributions (though usually conditional Gaussian distributions; Lauritzen & Wermuth, 1989). In particular variable types can be of **mixed type within and between boxes** (discrete and continuous components). General algorithms for computing estimates in every CGM under every possible variable type specification is challenging. One can attempt to fit the conditional distributions of the factorization with a **series of multiple univariate conditional regressions** (Wermuth & Cox, 2001). These are called traceable regressions.

Cox & Wermuth (1996; see also Caputo et al., 1997) lay out ideas for a heuristic selection strategy to approximate the CGM by univariate conditional regressions which is implemented in `coxwer`.

The `coxwer` Selection Algorithm:

- Start in the block with the lowest number
- Take one variable from that block. Fit main effects model with all the variables in the same block or higher block.
- Screen for quadratic effects (metric variables) and two-way interactions by adding of single terms. Retain the ones with an associated p-value $<$ `signif`.
- Fit the model with main and retained effects.
- Use backward selection to reduce the model. We do this with an infomration Criterion (BIC by default).
- Re-enter interactions for the terms that remain in the model.
- Use backward selection.
- Re-enter quadratic terms for remaining effects.
- Use backward selection.
- If other variables in the same block: Repeat for them. Else: jump to the next higher block and repeat.

The coxwer Functionality

We implemented an algorithm based on the ideas of the Cox-Wermuth heuristic in R for approximate fitting of JRCG and URRG models.

Currently, there are the following functions intended for the user:

Object	Description
`cw-class`	S3 class for objects from a Cox-Wermuth fit
`coxwer`	Fit a JRCG or a URRG via Cox-Wermuth selection strategy
`summary, print, plot, predict`	S3 methods for class cw
`adjmatrix`	Extracts the adjacency matrix
`write_cw`	Writes and saves the graph in igraph format
`prep_coxwer`	Setup of variable frame, block membership and variable type (interactive)

Using the `coxwer` Function

There are two ways of using the coxwer function: either with a formula or with a var.frame. We will use the Contraceptive Method Choice (CMC) data set for illustration which is a subset of the 1987 National Indonesia Contraceptive Prevalence Survey (Lim et. al., 1999). Overall there are 1473 observations of married women on 10 variables:

- Age (`age`; continuous)
- Education (`wifeEdu`; ordinal 1=low, 2, 3, 4=high)
- Husband's education (`husbEdu`; ordinal 1=low, 2, 3, 4=high)

- Number of children ever born (`nrChild`; count)
- Religion (`wifeRel`; binary; 0=Non-Islam 1=Islam)
- Wife's now working? (`wifeWork`; binary 0=Yes, 1=No)
- Husband's occupation (`husbOcc`; categorical 1, 2, 3, 4)
- Standard-of-living index (`solIndex`; ordinal 1=low, 2, 3, 4=high)
- Media exposure (`mediaExp`; binary 0=Good, 1=Not good)
- Contraceptive method used (`contraceptive`; categorical 1=No-use 2=Long-term 3=Short-term)

We assume the following block structure:

- Block 1 - Purely dependent (target) variables: contraceptive, nrChild
- Block 2 - Intermediate variable: mediaExp
- Block 3 - Intermediate variable: solIndex
- Block 4 - Intermediate variables: wifeEdu, husbEdu, wifeRel, wifeWork, husbOcc
- Block 5 - Purely explanatory variable: age

Blocks must always be numbered like this: Increasing integers starting from 1. The purely target variables are always in block 1, the purely explanatory variables are always in Block K where K is the highest block number (here 5). The blocks in between are labeled in the reverse direction of dependencies. We will abreviate this with Bx which stands for block x (Vx stands for variable x; so B5V2 is Block 5 Variable 2). Variables in BK have no antecedents (but possibly undirected dependencies within the block), variables in BK-1 have all variables in BK as antecedents and variables in BK-1 as possible undirected dependencies, variables in BK-2 have all variables in BK and BK-1 as antecedents and variables in BK-2 as possible undirected dependencies and so on until variables in B1 which have all variables in higher numbered blocks as antecedents.

The model is therefore of this type (~ directed edge, + undirected edge; the brackets are there for easier readability)

```
(B1V1 + B1V2 + ...) ~ (B2V1 + B2V2 + ...) ~ (B3V1 + B3V2 + B3V3 + ...)
```

For the example it is

```
B1V1 + B1V2 ~ B2V1 ~ B3V1 ~ B4V1 + B4V2 + B4V3 + B4V4 + B4V5 ~ B5V1
```

Fitting the model with a `var.frame`

Here the `coxwer` argument is a variable frame and an observations × variables data frame. The variable frame defines the block and type of a variable. It must have the same row names as the data frame has column names.

```
cmc_prep
```

```
##                type block
## age            cont     5
## wifeEdu         ord     4
## husbEdu         ord     4
## nrChild       count     1
## wifeRel         bin     4
## wifeWork        bin     4
## husbOcc       categ     4
## solIndex        ord     3
## mediaExp        bin     2
## contraceptive categ     1
```

The `prep_coxwer` function allows to define the variable frame interactively.

```
cmc_prep <- prep_coxwer(cmc)
```

Once the var.frame is set up one ca fit the model by

```
data(cmc)
rescmc <- coxwer(var.frame=cmc_prep, data=cmc)
```

Fitting the model with a formula

The formula interface to coxwer basically allows to specifiy the directed dependencies between block with the operator `~` and the undirected within the block by `+`. The structure is therefore:

```
B1V1 + B1V2 + ... ~  B2V1 + B2V2 + ... ~  B3V1 + B3V2 + B3V3 + ...
```

Or for the example

```
contraceptive + nrChild ~ mediaExp ~ solIndex ~ wifeRel + wifeWork + husbOcc + wifeEdu + husbEdu ~ age
```

One can the use the function with

```
data(cmc)
rescmc <- coxwer(contraceptive + nrChild ~              #Block 1
                 mediaExp ~                             #Block 2
                 solIndex ~                             #Block 3
                 wifeRel + wifeWork + husbOcc + wifeEdu + husbEdu ~ #Block 4
                 age,                                   #Block 5
                 data=cmc)
```

Further arguments to `coxwer`

`coxwer` allows to specify further arguments:

- `vartype`: This is an important argument especially for using the formual interface. When using the formula interface without a specified vartype , the function attempts to automatically detect the variable type from the data frame. This works fairly well for factors but is crude for metric variables. For teh latter it will always specify an OLS model. Setting vartype allows to fine tune the model used for the variables (see below). If this argumnet is used, then the order of characters in vartype must correspond to the variable in formula form left to right, or in var.frame form top to bottom.
- `adjfile`: Save the adjacency matrix to a file.
- `automatch`: Automatically assign the data type to the variables in the data frame according to variable type in the variable frame (the reverse of the autodetection)
- `pen`, `signif`: Arguments for screening and model selection. `pen` is the penalty for the information criterion used in `stepAIC` and `signif` the significance level when screening for higher-order effects and non-linearities.

- `contrasts`: The contrasts to be used for categorical predictors. Defaults to dummy coding for ordered and unordered factors. *`silent`: Flag for whether model fitting progress should be printed.

Depending on the type of variable, `coxwer` can use different univariate models

- For binary targets (`vartype="binary"`): binomial logistic models `stats::glm(...,family=binomial,link=logit`
- For unrestricted metric targets (`vartype="metric"`,`vartype="continuous"`): OLS/Gaussian linear models `stats::glm(...,family=gaussian,link=identity)`. When autodetection is done, this is used per default for every metric variable that is not further specified (message is printed in that case).

- For positive continuous targets (`vartype="gamma"`,`vartype="invgaussian"`): gamma or inverse Gaussian GLM `stats::glm(...,family=Gamma,link=inverse)` `stats::glm(...,family=inverse.gaussian,link=1/mu`
- For count targets (`vartype="count"`,`vartype="odcount"`): Poisson or negative binomial loglinear models `stats::glm(...,family=poisson,link=log)` `MASS::glm.nb(...,link=log)`
- For categorical targets (`vartype="categorical"`,`vartype="factor"`): multinomial logistic models `nnet::multinom(...,link=logit)`
- For ordinal targets (`vartype="ordinal"`): proportional odds logistic models `MASS::polr(...,link=logit)`

Predictors we treat as metric or as ordered/unordered factors (dummy–treatment–coding by default).

Using exponential families with canonical links assures that properties of conditional Gaussian graphs are approximately retained even when fitted with the CW procedure.

Worked Example

We now fit the model to the cmc data by using the formula interface.

```
data(cmc)
rescmc <- coxwer(contraceptive + nrChild ~ mediaExp ~ solIndex ~
                 wifeRel + wifeWork + husbOcc + wifeEdu + husbEdu ~ age,
                 data=cmc)
```

```
## Some models for continuous/metric variables are not further specified. Ordinary least squares estima
```

```
## TARGET: contraceptive (multinomial logit model)
## TARGET: nrChild (ordinary least squares model)
## TARGET: mediaExp (binomial logit model)
## TARGET: solIndex (proportional odds logit model)
## TARGET: wifeRel (binomial logit model)
## TARGET: wifeWork (binomial logit model)
## TARGET: husbOcc (multinomial logit model)
## TARGET: wifeEdu (proportional odds logit model)
## TARGET: husbEdu (proportional odds logit model)
```

S3 methods are available. Print prints the adjacency matrix.

```
print(rescmc)
```

```
## Adjacency Matrix:
##
##                  1  2  3  4  5  6  7  8  9 10
##  1 contraceptive  0  1  0  0  0  0  0  0  0  0
##  2 nrChild        1  0  0  0  0  0  0  0  0  0
##  3 mediaExp       0  0  0  0  0  0  0  0  0  0
```

```
##  4 solIndex       0  0  1  0  0  0  0  0  0  0
##  5 wifeRel        0  1  0  1  0  0  1  1  1  0
##  6 wifeWork       0  1  0  0  0  0  0  1  0  0
##  7 husbOcc        0  0  0  1  1  0  0  1  1  0
##  8 wifeEdu        1  1  1  1  1  0  1  0  1  0
##  9 husbEdu        0  0  0  1  0  0  1  1  0  0
## 10 age            1  1  1  1  1  0  1  1  1  0
```

One can summarize the object by extracting the model summaries for specific target variables by name or for all with `target="all"`. If no target is given, nothing is returned.

```
summary(rescmc,target=c("contraceptive","nrChild"))
```

```
## ---------- Summary for target variable: contraceptive ----------
## Call:
## nnet::multinom(formula = y ~ nrChild + wifeEdu + age + I(poly(nrChild,
##     2)[, 2]), data = data, Hess = TRUE, model = TRUE, trace = FALSE,
##     MaxNWts = dim(data)[1] + 1)
##
## Coefficients:
##   (Intercept) nrChild wifeEdu2 wifeEdu3 wifeEdu4      age
## 2      -2.293  0.3578   0.8821   1.8373    3.096 -0.04836
## 3       1.745  0.3558   0.2366   0.6443    1.337 -0.11909
##   I(poly(nrChild, 2)[, 2])
## 2                   -25.60
## 3                   -26.44
##
## Std. Errors:
##   (Intercept) nrChild wifeEdu2 wifeEdu3 wifeEdu4     age
## 2      0.5139 0.04444   0.4047   0.3870   0.3817 0.01212
## 3      0.3756 0.04058   0.2482   0.2453   0.2462 0.01137
##   I(poly(nrChild, 2)[, 2])
## 2                    3.570
## 3                    3.224
##
## Value/SE (Wald statistics):
##   (Intercept) nrChild wifeEdu2 wifeEdu3 wifeEdu4     age
## 2      -4.462   8.051   2.1794    4.748    8.112  -3.991
## 3       4.646   8.768   0.9532    2.627    5.433 -10.476
##   I(poly(nrChild, 2)[, 2])
## 2                   -7.171
## 3                   -8.202
##
## Residual Deviance: 2708
## AIC: 2736
## ---------- Summary for target variable: nrChild ----------
##
## Call:
## stats::glm(formula = y ~ contraceptive + wifeRel + wifeWork +
##     wifeEdu + I(poly(age, 2)), family = currfamily, data = data)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
```

```
## -5.801  -1.057  -0.167   0.929  10.402
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        2.10561    0.22420    9.39  < 2e-16
## contraceptive2     1.02986    0.13158    7.83  9.5e-15
## contraceptive3     0.99954    0.11312    8.84  < 2e-16
## wifeRel1           0.82276    0.14000    5.88  5.2e-09
## wifeWork1          0.52631    0.11164    4.71  2.7e-06
## wifeEdu2          -0.00114    0.18427   -0.01  0.99508
## wifeEdu3          -0.29816    0.18113   -1.65  0.09996
## wifeEdu4          -1.11024    0.17797   -6.24  5.8e-10
## I(poly(age, 2))1 53.31834    1.94461   27.42  < 2e-16
## I(poly(age, 2))2 -6.64633    1.88877   -3.52  0.00045
##
## (Dispersion parameter for gaussian family taken to be 3.349)
##
##     Null deviance: 8188.4  on 1472  degrees of freedom
## Residual deviance: 4899.6  on 1463  degrees of freedom
## AIC: 5973
##
## Number of Fisher Scoring iterations: 2
```

We see that the model for `nrChild` is an OLS model (due to autodetection). But since `nrChild` is a count, it might be better to tell the function that.

```
data(cmc)
rescmc <- coxwer(contraceptive + nrChild ~ mediaExp ~ solIndex ~
                 wifeRel + wifeWork + husbOcc + wifeEdu + husbEdu ~ age,
                 vartype=c("cate","count","bin","ord","bin","bin",
                           "ord","ord","ord","metric"),
                 data=cmc)
```

```
## Some models for continuous/metric variables are not further specified. Ordinary least squares estima
```

```
## TARGET: contraceptive (multinomial logit model)
## TARGET: nrChild (poisson loglinear model)
## TARGET: mediaExp (binomial logit model)
## TARGET: solIndex (proportional odds logit model)
## TARGET: wifeRel (binomial logit model)
## TARGET: wifeWork (binomial logit model)
## TARGET: husbOcc (proportional odds logit model)
## TARGET: wifeEdu (proportional odds logit model)
## TARGET: husbEdu (proportional odds logit model)
```

```
summary(rescmc,target=c("contraceptive","nrChild"))
```

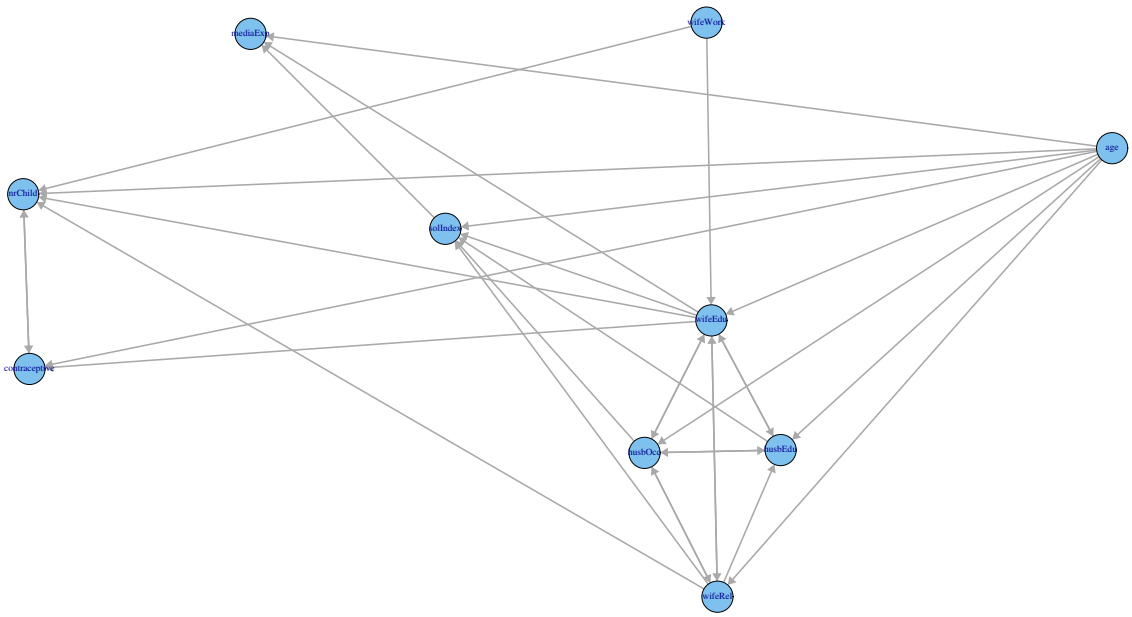
```
## ---------- Summary for target variable: contraceptive ----------
## Call:
## nnet::multinom(formula = y ~ nrChild + wifeEdu + age + I(poly(nrChild,
##     2)[, 2]), data = data, Hess = TRUE, model = TRUE, trace = FALSE,
##     MaxNWts = dim(data)[1] + 1)
```

```
##
## Coefficients:
##   (Intercept) nrChild wifeEdu2 wifeEdu3 wifeEdu4      age
## 2      -2.293  0.3578   0.8821   1.8373    3.096 -0.04836
## 3       1.745  0.3558   0.2366   0.6443    1.337 -0.11909
##   I(poly(nrChild, 2)[, 2])
## 2                   -25.60
## 3                   -26.44
##
## Std. Errors:
##   (Intercept) nrChild wifeEdu2 wifeEdu3 wifeEdu4     age
## 2      0.5139 0.04444   0.4047   0.3870   0.3817 0.01212
## 3      0.3756 0.04058   0.2482   0.2453   0.2462 0.01137
##   I(poly(nrChild, 2)[, 2])
## 2                    3.570
## 3                    3.224
##
## Value/SE (Wald statistics):
##   (Intercept) nrChild wifeEdu2 wifeEdu3 wifeEdu4     age
## 2      -4.462   8.051   2.1794    4.748    8.112  -3.991
## 3       4.646   8.768   0.9532    2.627    5.433 -10.476
##   I(poly(nrChild, 2)[, 2])
## 2                   -7.171
## 3                   -8.202
##
## Residual Deviance: 2708
## AIC: 2736
## ---------- Summary for target variable: nrChild ----------
##
## Call:
## stats::glm(formula = y ~ contraceptive + wifeRel + wifeWork +
##     wifeEdu + age + I(poly(age, 2)[, 2]), family = currfamily,
##     data = data)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -3.362  -0.648  -0.103   0.534   3.591
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -1.22834    0.11021  -11.15  < 2e-16
## contraceptive2        0.33405    0.03952    8.45  < 2e-16
## contraceptive3        0.34824    0.03575    9.74  < 2e-16
## wifeRel1              0.26392    0.04437    5.95  2.7e-09
## wifeWork1             0.17109    0.03505    4.88  1.1e-06
## wifeEdu2              0.01222    0.05007    0.24     0.81
## wifeEdu3             -0.07574    0.04964   -1.53     0.13
## wifeEdu4             -0.35135    0.04961   -7.08  1.4e-12
## age                   0.05817    0.00212   27.48  < 2e-16
## I(poly(age, 2)[, 2]) -5.16323    0.62203   -8.30  < 2e-16
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 2529.0  on 1472  degrees of freedom
```

```
## Residual deviance: 1452.1  on 1463  degrees of freedom
## AIC: 5530
##
## Number of Fisher Scoring iterations: 5
```

We can use the `igraph` facilities to plot the model. After some tweeking and ordering the vertices according to the block structure we have

```
plot(rescmc)
```



References

- Caputo, A., Heinicke, A. & Pigeot, I. (1997). A graphical chain model derived from a model selection strategy for the sociologists graduates study. *Collaborative Research Center 386, Discussion Paper 73.*
- Cox, D. & Wermuth, N. (1996). *Multivariate Dependencies: Models, Analysis, Interpretation.* Florida:Chapman & Hall/CRC.
- Cox, D. & Wermuth, N. (2001). Joint response graphs and separation induced by triangular systems. *Research Report, Australian National University.*
- Lauritzen, S. & Wermuth, N. (1989). Graphical models for associations between variables, some of which are qualitative and some quantitative. *The Annals of Statistics, 31–57.*
- Lim, T.-S., Loh, W.-Y. & Shih, Y.-S. (1999). A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-three Old and New Classification Algorithms. *Machine Learning, 40, 203–238.*