First line of title
second line of title

Master Thesis in Biostatistics (STA495)

by

Name of student

Matriculation number

supervised by

Name of responsible supervisor (with title)

Name of supervisor (with title and affiliation if external)

Zurich, month year

p-values:

their use, abuse and proper use illustrated with seven facets

Mäxli Musterli

Version August 6, 2018

Contents

Preface

Howdy!

Max Muster
June 2018

Chapter 1

Introduction

The aim of this master thesis is to implement the variable importance measure LMG (named after the authors Lindeman, Merenda, and Gold (Grömping, 2007)) in linear models estimated with Bayesian methods.

Regression models are popular in many applied research areas (Nimon and Oswald, 2013). These models provide a tool to find an association between a response variable Y and a set of explanatory variables. The explanatory variables are also called predictors or covariates. Regression parameters provide us the information how much the response variable is expected to change when a predictor changes by one unit, given all other predictors in the model stay the same. The last subsentence is very important for the correct interpretation of the regression parameters. It shows also that the parameter value of a predictor is dependent on the other predictors in the model.

Because predictors are often correlated to some degree to each other, it is clear that it is not an easy task to find the most important predictors in a model. The first question then is: What do we mean by the importance of a predictor? A question that is not easy answered and depending on the research question. Grömping (2015) concludes that there may never be a unique accepted definition of what variable importance is. Different metrics exist to quantify the importance of predictors. These metrics focus on different aspects and with correlated predictors they lead to different conclusions. A summary of the metrics can be found in Grömping (2015).

A distinction should be made between the importance of predictors in regression models that are used to predict future data and regression models who wish to find an association between predictors and the response variable. In the former case, the aim is only to reduce the error between the predicted values and the real observed values. It does not really matter how we get there. In the other case, we are interested in the strength of the relationship between the predictors and the response variable. A predictor may explain little of the response variable given two other correlated predictors are already included in a regression model. However, this from the regression output unimportant predictor may be the main cause of the other two predictor values. It therefore may somehow be the most important predictor in this model (Grömping, 2007).

The causal relationship between the variables is missing in the regression model. Regressing conditional on other variables or using univariate regression models only provide us some parts of the bigger picture about the predictor in a model. Some authors recommend that the variable importance metric is based on both components. Which variable importance metrics are the most useful ones is still an open debate. A convincing theoretical basis is still lacking for all of them. Grömping (2015) recommends to use the existing best practices, until a more profound solution is found. For variance (or generally goodness of fit) decomposition based importance, she recommends to use LMG enhanced with joint contributions or dominance analysis (Grömping, 2007).

Chapter 2

Methods

2.1 LMG variable importance metric

The focus of this master thesis is on the LMG variable importance metric. The LMG is a metric that is based on variance decomposition. The total R^2 of a model is decomposed onto the predictors. Marginal and conditional information are incorporated (Grömping, 2015) . The following formulas are taken from Grömping (2015). The same mathematical notations are used.

The following notations for the explained variance (1) and sequentially added variance (2) simplify the notation of the LMG formula.

$$\operatorname{evar}(S) = \operatorname{Var}(Y) - \operatorname{Var}(Y \mid X_j, j \in S), \tag{2.1}$$

$$\operatorname{svar}(M \mid S) = \operatorname{evar}(M \cup S) - \operatorname{evar}(S), \tag{2.2}$$

, where S and M denote disjoint sets of predictors.

The LMG formula is given below for the first predictor only. Because of exchangeable predictors, this is no loss of generality. $\mathrm{R}^2(S)$ can be written as $\operatorname{evar}(S)/\operatorname{Var}(Y)$.

$$\begin{aligned}
\operatorname{LMG}(1) &= \frac{1}{p!} \sum_{\pi permutation} \operatorname{svar}(\{1\} \mid S_1(\pi)), \\
&= \frac{1}{p!} \sum_{S \subseteq \{2,\ldots,p\}} n(S)!\,(p - n(S) - 1)!\operatorname{svar}(\{1\} \mid S) \\
&= \frac{1}{p} \sum_{i=0}^{p-1} \left(\sum_{\substack{S \subseteq \{2,\ldots,p\} \\ n(S)=1}} \operatorname{svar}(\{1\} \mid S) \right) \bigg/ \binom{p-1}{i} \\
&= \frac{1}{p} \sum_{i=0}^{p-1} \frac{\sum_{\substack{S \subseteq \{2,\ldots,p\} \\ n(S)=1}} \operatorname{svar}(\{1\} \mid S)}{\binom{p-1}{i}},
\end{aligned}$$

where $S_1(\pi)$ is the set of predecessors of predictor 1.

The different writings of the formula help to better understand what is calculated in the LMG metric. The R^2 of the model including all predictors is decomposed. In the top formula the LMG value of predictor 1 is represented as an unweighted average over all orderings of the sequential added variance contribution of predictor 1. The middle formula shows that the calculation can be done computationally more efficient. The orderings with the same set of predecessors S are combined into one summand. Instead of $p!$ summands only 2^{p-1} summands need to be calculated. The bottom formula shows that the LMG metric can also be seen as the unweighted average over average explained variance improvements when adding predictor 1 to a model of size i without predictor 1 (Grömping, 2015). The LMG metric is implemented in the R package relaimpo (Grömping, 2006).

Chevan and Sutherland (1991) propose that instead of only using the variances, an appropriate goodness-of-fit metric can as well be used in the LMG formula. They name their proposal hierarchical partitioning. The requirements are simply: an initial measure of fit when no predictor variable is present, a final measure of fit when N predictor variables are present, all intermediate models when various combinations of predictor variables are present. The LMG component of each variable is named independent component (I). The sum of the independent components (I) results then in the overall goodness-of-fit metric. The difference between the goodness-of-fit when only the predictors itself is included in the model, compared to its independent component (I) is named the joint contribution (J). Hierarchical partitioning is implemented in the hier.part package (Walsh and Nally, 2015). The LMG values are calculated when R^2 is chosen as the goodness-of-fit measure. The partitioning function of the hier.part package is used in this master thesis. The hierarchical partitioning function accepts as input a data frame with the R^2 values of all submodels. Of note, the hier.part package is only guaranteed to work for 9 predictors and does not work at all for more than 12 predictors.

2.2 Appropriate R^2 definitions in the Bayesian framework

The focus of this master thesis is on the linear model. For the linear model the R^2 is the most widely used goodness-of-fit metric. Different formulas for R^2 exist Kvalseth (1985), all leading to the same value when and intercept is included and the model is fitted by maximum likelihood.

Two popular definitions are:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2.3}$$

$$R^2 = \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}, \qquad i = 1, \ldots, n, \tag{2.4}$$

, where $\hat{y}_i == \mathrm{E}(y \mid X_i, \hat{\theta})$. $\hat{\theta}$ is the maximum likelihood estimate of the regression coefficients.

When other estimation methods than maximum likelihood are used (2.3) can be negative and (2.4) can be bigger than 1. This is not uncommon in a Bayesian regression setting, when samples of the posterior parameter distribution are employed. A model that explaines more than 100% of the variance does not make sense. A negative R^2 is also difficult to interpret. A negative R^2 may be interpreted as a fit that is worse than the mean of the data. This can make sense for predictive purposes, e.g. when new data from a test set is predicted by leave-one-out crossvalidation (Alexander *et al.*, 2015). For non predicting purposes a negative R^2 does not make sense. The aim of the LMG formula is to gain some more information about the possible association between variables. A predictor can not explain less than zero variance in the population. To respect the non-negative share property of the LMG formula the R^2 of submodels should not decrease when adding predictors. Both classical R^2 definitions seem not to be well suited for the LMG metric in the Bayesian framework.

A more reasonable R^2 definition for the LMG formula in the Bayesian framework can be found by noting that the variance of the linear model can also be written as

$$\mathrm{Var}(y) = \mathrm{Var}(\mathbf{X}\boldsymbol{\beta}) + \sigma^2 = \boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{\mathbf{XX}} \boldsymbol{\beta} + \sigma^2, \tag{2.5}$$

where $\boldsymbol{\beta}^\top = (\beta_1 \dots \beta_p)$ are the regression parameters without the intercept. $\boldsymbol{\Sigma}_{\mathbf{XX}}$ is the covariance matrix of the regressors.

By using this variance definition Gelman *et al.* (2017) proposes to use

$$\mathrm{R}^2_{Gelman} = \frac{\mathrm{Var}(\sum_{i=1}^n \hat{y}_i^s)}{\mathrm{Var}(\sum_{i=1}^n \hat{y}_i^s) + \mathrm{Var}(\sum_{i=1}^n e_i^s)}, \qquad i = 1, \dots, n, \tag{2.6}$$

, where $\hat{y}_i^s = \mathrm{E}\left(y \mid X_i, \theta^s\right)$ and the vector of errors $e_i^s = y_i - \hat{y}_i^s$ and $\theta^s, s = 1, \dots, S$ are draws from the posterior parameter distribution. The formula is then guaranteed to be between 0 and 1. The R^2 can be interpreted as a data-based estimate of the proportion of variance explained for new data under the assumption that the predictors are held fixed (Gelman *et al.*, 2017).

In the Bayesian framework the σ^2 parameter is explicitly modeled in the standard linear regression setting. It is therefor possible to sample the σ^2 parameter from its posterior distribution instead of defining the error as in (2.6), which would lead to the following definition:

$$\mathrm{R}^2 = \frac{\mathrm{Var}(\sum_{i=1}^n \hat{y}_i^s)}{\mathrm{Var}(\sum_{i=1}^n \hat{y}_i^s) + \sigma_s^2} \tag{2.7}$$

$$= \frac{\boldsymbol{\beta}_s^\top \boldsymbol{\Sigma}_{\mathbf{XX}} \boldsymbol{\beta}_s}{\boldsymbol{\beta}_s^\top \boldsymbol{\Sigma}_{\mathbf{XX}} \boldsymbol{\beta}_s + \sigma_s^2} \tag{2.8}$$

, where $\hat{y}_i^s = E\left(y \mid X_i, \theta^s\right)$, and $\theta^s, s = 1, \dots, S$ are draws from the posterior parameter distribution.

The predictors in (2.6) and (2.8) could also be taken as random (Gelman *et al.*, 2017). The predictors are then called stochastic predictors. Using the sample covariance estimate provides us then just an estimate of the true covariance structure. With stochastic predictors there is an additional uncertainty in the R^2 formula that can have a large influence on the R^2 and especially the LMG values.

In practice (2.8) and (2.6) should lead to similar values in the standard linear model. In my opinion it is more reasonable to go the full Bayesian route and sample σ^2. This provides us the opportunity to include prior information about σ^2 directly into to R^2 calculations. The LMG calculations in the examples of this master thesis will therefore be based on (2.8). The benefit of (2.6) is that it also works for generalized linear models, where we often have no separate variance parameter.

The denominator of R^2 is no longer fixed in (2.6) and (2.8). We can therefore no longer interpret an increase in R^2 as an improved fit to a fixed target (Gelman *et al.*, 2017). This seems to be problematic for the LMG formula in the Bayesian framework. However, in the linear model it is possible to calculate the R^2 of all submodels from the parameters of the fullmodel and the covariance matrix of the predictors. We therefore compare all submodels of a posterior sample to the same fixed value. How it is possible to get the R^2 of the submodels from the full model is shown in the next section.

2.3 Conditional variance formula

For two predictors (2.5) simplifies to

$$\mathrm{Var}(y) = \beta_1^2 \,\mathrm{Var}(X_1) + 2\beta_1\beta_2 \,\mathrm{Cov}(X_1, X_2) + \beta_2^2 \,\mathrm{Var}(X_2) + \sigma^2, \tag{2.9}$$

When predictor X_1 is alone in the model the explained variance includes the variance of the predictor itself, the whole covariance term and in addition some of the contribution of the variance of X_2 in (2.9) . In mathematical notation that is

$$\operatorname{svar}(X_1 \mid \emptyset) = \beta_1^2 \operatorname{Var}(X_1) + 2\beta_1\beta_2 \operatorname{Cov}(X_1, X_2) + \beta_2^2 \operatorname{Var}(X_2)\rho_{12}^2$$

The contribution of the second regressor is then simply the difference to the total explained variance (Grömping, 2007).

In the general case with p regressors, the conditional variance formula (2.10) can be used to calculate the R^2 of all submodels. The conditional variance formula can for example be used to specify the conditional distribution of a multivariate normal distribution $\mathbf{Y}$.

The elements of the vector $\mathbf{Y}$ are reordered as

$$\mathbf{Y} = \begin{pmatrix} \mathbf{Y}_1 \\ \mathbf{Y}_2 \end{pmatrix}, \mathbf{Y}_1 \in \mathbb{R}^q, \mathbf{Y}_2 \in \mathbb{R}^{p-q}.$$

The joint distribution is a multivariate normal distribution with elements

$$\begin{pmatrix} \mathbf{Y}_1 \\ \mathbf{Y}_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}, \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{pmatrix} \right), \ \boldsymbol{\Sigma}_{21} = \boldsymbol{\Sigma}_{12}^T,$$

the conditional distribution is normally distributed again with mean

$$\mathrm{E}(\mathbf{Y}_1|\mathbf{y}_2) = \boldsymbol{\mu}_1 \ + \ \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\mathbf{Y}_2 \ - \ \boldsymbol{\mu}_2)$$

and the conditional variance is

$$\operatorname{Var}(\mathbf{Y}_1|\mathbf{y}_2) = \boldsymbol{\Sigma}_{11} \ - \ \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}. \tag{2.10}$$

The aim is to calculate R^2 of a submodel containining the predictors $\mathbf{X}_{q\dots p}$, and regression coefficients $\boldsymbol{\beta}^\top = (\beta_1, \dots, \beta_p)$ without the intercept. The regression coefficients are further separated in $\boldsymbol{\beta}_{1,\dots,q-1}^\top = (\beta_1, \dots, \beta_{q-1})$ and $\boldsymbol{\beta}_{q,\dots,p}^\top = (\beta_q, \dots, \beta_p)$.

As in the multivariate normal distribution example above, the covariance matrix of p predictors is written as

$$\operatorname{Cov}(\mathbf{X}) = \boldsymbol{\Sigma}_{\mathbf{XX}} = \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{pmatrix}^{p \times p},$$

$$\text{where} \qquad \begin{aligned} \boldsymbol{\Sigma}_{11} &= \operatorname{Cov}(\mathbf{X}_{1,\dots,q-1}, \mathbf{X}_{1,\dots,q-1}), \\ \boldsymbol{\Sigma}_{12} &= \operatorname{Cov}(\mathbf{X}_{1,\dots,q-1}, \mathbf{X}_{q,\dots,p}), \\ \boldsymbol{\Sigma}_{22} &= \operatorname{Cov}(\mathbf{X}_{q,\dots,p}, \mathbf{X}_{q\dots p}). \end{aligned}$$

The conditional variance of the predictors $\mathbf{X}_{1,\dots,q-1}$ given the predictors $\mathbf{X}_{q,\dots,p}$ is then

$$\operatorname{Cov}(\mathbf{X}_{1,\dots,q-1} \mid \boldsymbol{x}_{q,\dots,p}) = \boldsymbol{\Sigma}_{11} \ - \ \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}.$$

The total explained variance of the full model $\mathbf{X}_{1\dots p}$ omits simply the σ^2 parameter in (2.5) , which is

$$\operatorname{evar}(\mathbf{X}_{1,\dots,p}) = \boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{\mathbf{XX}} \boldsymbol{\beta}.$$

The explained variance of a submodel can be calculated by subtracting from the total explained variance the explained variance of the not-in-the-model-included-predictors that is not explained by in-the-model-included-predictors. The variance that is not explained by in-the-model-included-predictors is given by the variance of the not-in-the-model-included predictors conditional on the in-the-model-included-predictors. The explained variance of a submodel containing predictors $\mathbf{X}_{q,\dots,p}$ can therefore be written as

$$\operatorname{evar}(\mathbf{X}_{q\dots p}) = \operatorname{evar}(\mathbf{X}_{1,\dots,p}) - \boldsymbol{\beta}_{1,\dots,q-1}^{\top} \operatorname{Cov}(\mathbf{X}_{1,\dots,q-1} \mid \boldsymbol{x}_{q\dots p}) \boldsymbol{\beta}_{1,\dots,q-1}. \tag{2.11}$$

To gain the the R^2 value of the submodel we need to divide the explained variance by the total variance, which is

$$\operatorname{evar}(\mathbf{X}_{q,\dots,p}) / \operatorname{Var}(\mathbf{Y}),$$

where $\operatorname{Var}(\mathbf{Y})$ is definied as $\boldsymbol{\beta}^{\top} \boldsymbol{\Sigma}_{\mathbf{XX}} \boldsymbol{\beta} + \sigma^2$.

A posterior density distribution is obtained for the regression parameters in the Bayesian regression setting. The LMG formula requires calculation of the R^2 values for all $2^p - 1$ submodels. Samples from the joint posterior paramters of the full model are used to calculate the explained variance of the submodels. For each sample the conditional variance formula is used to obtain the R^2 of the $2^p - 1$ submodels. The property that all shares should be non-negative and the dependence of the parameter values from the submodels to each other is then respected for each sample.

Instead of using the conditional mean formula to get the R^2 of the submodels, it would be possible to fit a separate Bayesian model for each submodel. An R^2 distribution can easily be built for each submodel by using (2.6) or (2.8). However, the problem is how to calculate the LMG values out of these R^2 distributions. If we just sample independently from the R^2 distributions, the dependence of the paramter values of the submodels to each other is ignored. We would have many possibly true parameter values of a predictor in the same LMG comparison. It would then also be possible that the R^2 decreases when adding predictors. Another drawback is that it would be much more time consuming to fit a separate Bayesian model for each submodel. Using the conditional variance formula on the full model allows us to calculate LMG values in the Bayesian framework in a reasonable amount of time. Depending on the number of predictors and the number of posterior samples, the calculations still take some time in the Bayesian framework. For stochastic predictors the computation time is multiplied by the number of covariance samples.

2.4 Bayesian Regression

This Section provides a short introduction to Bayesian regression. It further shows that assuming stochastic or non-stochastic predictors results in the same posteriors for the regression parameters under some assumptions. It is summarized from the book Bayesian Analysis for the Social Sciences (Jackman, 2009).

In regression analysis we are interested in the dependence of $\boldsymbol{y}$ on $\mathbf{X}$. The conditional mean of a continuous response variable $\boldsymbol{y} = (y_1, \dots, y_n)^{\top}$ is related to a $n \times k$ predictor matrix $\mathbf{X}$ via a linear model,

$$\mathrm{E}(\boldsymbol{y} \mid \mathbf{X}, \boldsymbol{\beta}) = \mathbf{X}\boldsymbol{\beta},$$

where $\boldsymbol{\beta}$ is a $k \times 1$ vector of unknown regression coefficients.

Under some assumptions about the density, conditional independence and homoskedastic variances, the regression setting can be written as

$$\boldsymbol{y} \mid \mathbf{X}, \boldsymbol{\beta}, \sigma^2 \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2 \mathbf{I}_n).$$

Under the assumption of weak exogeneity and conditional independence, the joint density of the data can be written as

$$p(\boldsymbol{y}, \mathbf{X} \mid \boldsymbol{\theta}) = p(\boldsymbol{y} \mid \mathbf{X}, \boldsymbol{\theta}_{y|x})\, p(\mathbf{X} \mid \boldsymbol{\theta}_x),$$

where $\boldsymbol{\theta} = (\boldsymbol{\theta}_{y|x}, \boldsymbol{\theta}_x)^\top$. The weak exogeneity assumption implicates that the whole information about $\boldsymbol{y}_i$ is contained in x_i and $\boldsymbol{\theta}_{y|x}$. Knowledge of the parameters $\boldsymbol{\theta}_{x_i}$ provides no additional information about $\boldsymbol{y}_i$. The interest of regression is mostly in the posterior parameters $\boldsymbol{\theta}_{y|x}$. These posterior densities are proportional to the likelihood of the data multiplied by the prior density. The joint density $p(\boldsymbol{y}, \mathbf{X} \mid \boldsymbol{\theta})$ is used to learn about the posterior parameters, via Bayes Rule

$$p(\boldsymbol{\theta} \mid \boldsymbol{y}, \mathbf{X}) \propto p(\boldsymbol{y}, \mathbf{X} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta}).$$

The dependence of $\boldsymbol{y}$ on $\mathbf{X}$ is captured in the parameters $\boldsymbol{\theta}_{y|x} = (\beta, \sigma^2)$. Under the assumption of independent prior densities about $\boldsymbol{\theta}_{y|x}$ and $\boldsymbol{\theta}_x$ the posterior distribution of the parameters can be written as

$$p(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta}_x \mid \boldsymbol{y}, \mathbf{X}) = \frac{p(\boldsymbol{y} \mid \mathbf{X}, \boldsymbol{\beta}, \sigma^2)\, p(\boldsymbol{\beta}, \sigma^2)}{p(\boldsymbol{y} \mid \mathbf{X})} \times \frac{p(\mathbf{X} \mid \boldsymbol{\theta}_x)\, p(\boldsymbol{\theta}_x)}{p(\mathbf{X})}. \tag{2.12}$$

This factorization shows that under the above mentioned assumptions the posterior inference about the parameters $\boldsymbol{\theta}_{y|x} = (\beta, \sigma^2)$ is independent from the inference about $\boldsymbol{\theta}_x$ given data $\mathbf{X}$. This also means that the assumptions about $\mathbf{X}$ being non-stochastic or stochastic result in the same posterior density of $\boldsymbol{\theta}_{y|x}$. In the case of non-stochastic regressors $p(\mathbf{X})$ and $\boldsymbol{\theta}_x$ drop out of the calculations. For stochastic predictors it means that given $\mathbf{X}$ nothing more can be gained about $\theta_{y|x} = (\boldsymbol{\beta}, \sigma^2)$ from knowing $\boldsymbol{\theta}_x$.

The focus of regression is on $\boldsymbol{\theta}_{y|x} = (\boldsymbol{\beta}, \sigma^2)$, for which it does not matter whether we assume fixed or stochastic predictors under the above mentioned assumptions. For the LMG formula, the variance of the predictors is also incorporated. The LMG formula may be especially interesting for continuous predictors, which often are of stochastic nature. Grömping (2006) recommends in most cases to use the non fixed regressor option when calculating bootstrap confidence intervals. For stochastic predictors the information about $\boldsymbol{\theta}_x$ would therefore also be relevant. As seen in (2.12) inference about $\boldsymbol{\theta}_x$ is independent from inference about $\boldsymbol{\theta}_{y|x}$. If we have stochastic predictors and ignore dependence we just use an estimate of the covariance matrix and do not incorporate this uncertainty. Because the explained variance is calculated by $\boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{\mathbf{XX}} \boldsymbol{\beta}$, inference about $\boldsymbol{\theta}_x$ seems to be equally important as inference about $\boldsymbol{\theta}_{y|x}$ for stochastic predictors. If we know the distribution of the $p(\mathbf{X})$ the $\boldsymbol{\theta}_x$ could be estimated. However, the computation times are then much higher. We would need to do the whole LMG calculation for each posterior covariance sample of the predictors. Depending on the number of predictors this would quite take some time. In most cases the problem is that we do not know the distribution of the $\mathbf{X}$. As a practical solution we could then use nonparametric bootstrapping to include the uncertainty of the stochastic predictors in the LMG formula. We would then also need to do the whole LMG calculations for each bootstrap sample of the covariance matrix. There exist also different covariance estimator. The shrinkage method may be an interesting estimator with some nice properties (Schäfer and Strimmer, 2005).

Chapter 3

Examples

The following chapter presents the Bayesian LMG implementation on two examples. Simulated data is used for the first example. Empirical data is used for the second example.

3.1 Simulated Data

We assume a simple model for the first example:

$$
\begin{aligned}
&Y_i \sim \mathcal{N}(\beta_0 + x_1\beta_1 + x_2\beta_2 + x_3\beta_3 + x_4\beta_4,\, \sigma^2),\\
&\beta_1 = 0.5, , \beta_2 = 1,\ \beta_3 = 2,\ \beta_4 = 0,\ \sigma^2 = 1\\
&\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3, \mathbf{X}_4 \sim \mathcal{N}(0, 1)
\end{aligned}
$$

The values of the four predictors are sampled from a standard normal distribution. These values are then multiplied by the regression coefficients. A standard normal distributed error is added to obtain the dependent variable. Fifty observations were sampled.

The following Code was used to simulate the data :

```r
x1 <- rnorm(50, 0, 1); x2 <- rnorm(50, 0, 1)
x3 <- rnorm(50, 0, 1); x4 <- rnorm(50, 0, 1)
b1 <- 0.5; b2 <- 1; b3 <- 2; b4 <- 0.8
#b1 <- 1; b2 <- 1; b3 <- 1; b4 <- 1

y <- b1*x1 + x2*b2 + b3*x3 + b4*x4 + rnorm(50, 0, 1)

df <- data.frame(y = y, x1 = x1, x2 = x2, x3 = x3, x4 = x4)
```

The model is fitted using the `rstanarm` package (Stan Development Team, 2016) with the default priors for the regression and σ^2 parameters. These default priors are called weakly informative priors, because they take into account the order of magnitude of the variables by using variance of the observed data. Information about these priors can be found in Stan Development Team (2017). A burn-in period of 1000, a sample size of 20000 and a thinning of 20 were chosen, resulting in a posterior sample size of 1000. The first few posterior samples are shown in Table 3.1.

For each posterior sample of the parameters the R^2 value is calculated. The R^2 of the submodels is then calculated by the conditional variance formula for each posterior sample. The resulting R^2 values of the first few posterior samples is shown in Table 3.2. The thinning makes sense in this case to reduce the computational burden and still obtain an appropriate posterior of the R^2 values (Link and Eaton, 2012).

Table 3.1: Samples from the posterior distributions of the regression parameters

	x1	x2	x3	x4	sigma
sample 1	0.555	1.023	1.99	0.835	1.103
sample 2	0.283	1.153	2.19	0.679	1.143
sample 3	0.571	1.131	2.03	0.877	1.026
sample 4	0.455	1.231	2.00	0.925	1.266
sample 5	0.600	0.908	1.90	0.700	0.991
sample 6	0.730	1.018	2.03	0.401	1.266
sample 7	0.363	1.029	1.84	0.619	1.145
sample 8	0.296	0.805	1.77	0.599	1.290
sample 9	0.177	0.812	1.61	0.376	1.406
sample 10	0.694	1.018	1.72	1.028	1.192

Table 3.2: R^2 for all submodels for the first six posterior samples

	sample 1	sample 2	sample 3	sample 4	sample 5	sample 6
none	0.000	0.000	0.000	0.000	0.000	0.000
x1	0.055	0.023	0.057	0.041	0.067	0.081
x2	0.055	0.059	0.069	0.080	0.048	0.055
x3	0.639	0.687	0.640	0.580	0.667	0.637
x4	0.068	0.040	0.072	0.076	0.053	0.014
x1 x2	0.099	0.075	0.114	0.109	0.104	0.122
x1 x3	0.681	0.702	0.684	0.610	0.719	0.702
x1 x4	0.125	0.064	0.131	0.118	0.123	0.096
x2 x3	0.750	0.806	0.770	0.722	0.769	0.747
x2 x4	0.126	0.102	0.144	0.159	0.104	0.071
x3 x4	0.705	0.725	0.709	0.653	0.718	0.650
x1 x2 x3	0.778	0.812	0.798	0.738	0.805	0.794
x1 x2 x4	0.171	0.118	0.190	0.190	0.161	0.139
x1 x3 x4	0.748	0.741	0.755	0.685	0.772	0.716
x2 x3 x4	0.819	0.847	0.843	0.799	0.823	0.762
all	0.848	0.854	0.873	0.817	0.861	0.809

The `hier.part` package is used to calculate the LMG value for each posterior sample. The LMG posteriors are shown in Table3.3 . The independent component (I) represents the LMG value. The joint contribution (J) represents the difference from the independent component to the explained variance of the predictor-only model (T). Assuming stochastic or non-stochastic regressors has an influence on the uncertainty. First, non-stochastic regressors are assumed. The resulting LMG values and joint contributions with a 95% credible interval are shown in Table 3.3. An option to display the resulting LMG distribution is shown in Figure 3.1. Using the default weakly informative priors, the LMG distributions obtained from the Bayesian framework are very similar to the bootstrap confidence intervals assuming non-stochastic predictors of the LMG estimates obtained from the `relaimpo` package, as shown in Table 3.4.

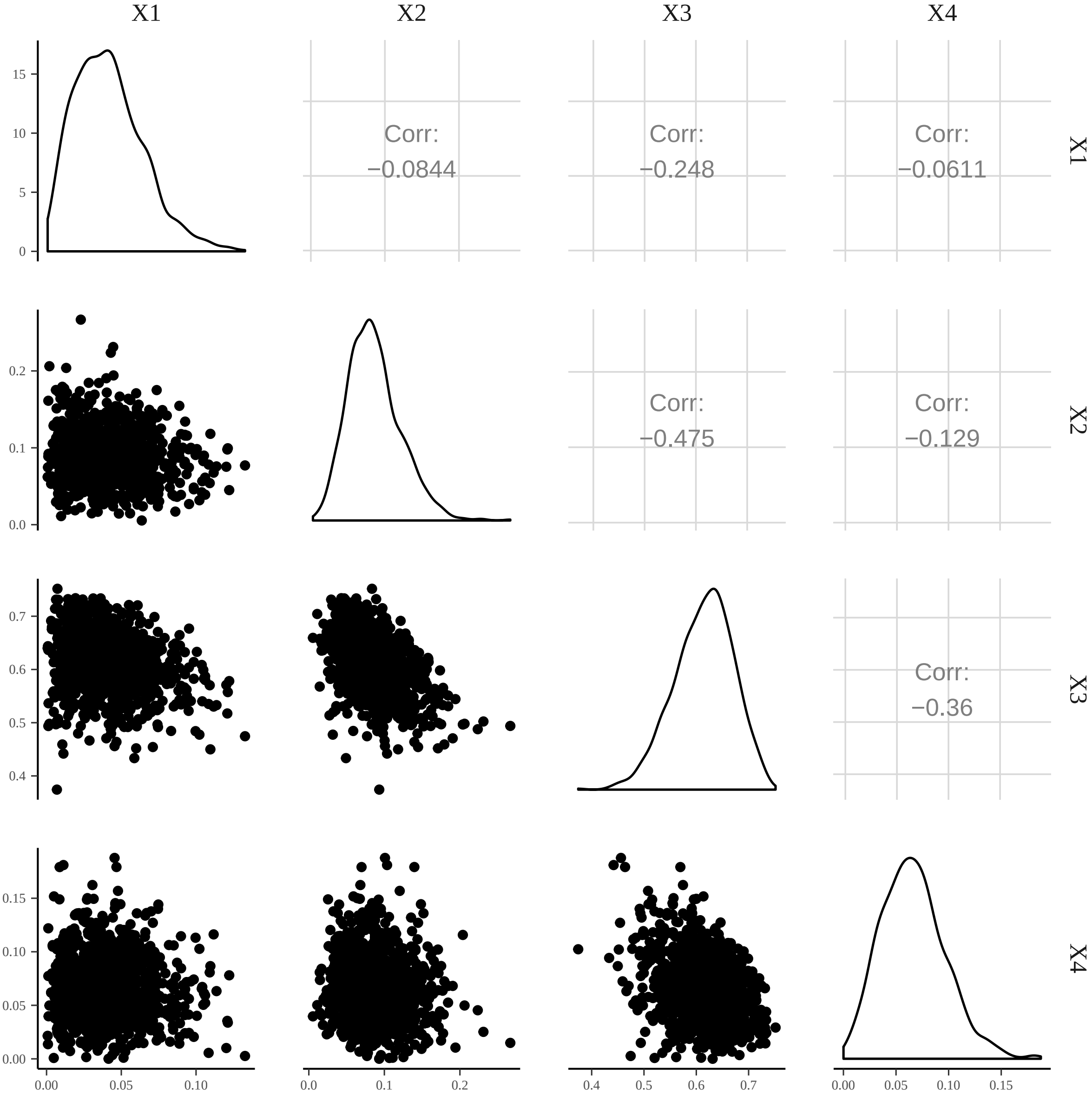

Figure 3.1: Posterior distribution of LMG values.

Table 3.3: Variance decomposition for non-stochastic predictors. I = LMG values, J = joint contribution, Total = total explained variance in one-predictor only model

Variable	**I**	**J**	**Total**
x1	0.039 (0.007, 0.094)	0.012 (0.004, 0.019)	0.051 (0.012, 0.113)
x2	0.083 (0.03, 0.164)	-0.022 (-0.032, -0.013)	0.06 (0.015, 0.133)
x3	0.618 (0.497, 0.714)	-0.02 (-0.031, -0.008)	0.598 (0.472, 0.697)
x4	0.064 (0.014, 0.134)	-0.001 (-0.002, -3.301×10^{-4})	0.063 (0.014, 0.133)

Table 3.4: Variance decomposition for non-stochastic predictors. I = LMG values, J = joint contribution, Total = total explained variance in one-predictor only model

	LMG value (95%-CI)	
Variable	Relaimpo	**Bayesian framework**
x1	0.092 (0.046, 0.157)	0.039 (0.007, 0.094)
x2	0.109 (0.062, 0.177)	0.083 (0.03, 0.164)
x3	0.636 (0.548, 0.723)	0.618 (0.497, 0.714)
x4	0.03 (0.007, 0.072)	0.064 (0.014, 0.134)

In this example we know that the predictor values were sampled from a normal distribution. It would therefore be more reasonable to assume stochastic predictors. Under the assumption of weak exogeinity and conditional independence, the posterior distributions of the regression parameters $\boldsymbol{\beta}$ are valid for non-stochastic and stochastic predictors. However, the uncertainty about the LMG values needs to include the uncertainty about the covariance matrix. If we know the distribution of the predictors we can incorporate this information and obtain the posterior distribution of the covariance matrix. The package `Jags` was used for inference about the covariance matrix in a Bayesian way. As an alternative non-parametric bootstrap was used for inference about the covariance matrix. Both approaches resulted in very similar LMG values. Using the default priors further resulted in very similar LMG values as the non-parametric bootstrap option of the relaimpo package. Table 3.5 shows the LMG values of these approaches. Using the bootstrap samples of the covariance matrix or samples from the posterior covariance matrix resulted in very similar LMG values. Bootstrap seems to be a valuable option for stochastic predictors when the distribution of the predictors is unknown. Even when the distribution is known, the difference seems to be tiny. A benefit of going the full Bayesian way is that we can also include prior knowledge about the covariance matrix. For stochastic predictors, compared to non-stochastic predictors, the uncertainty about the covariance matrix is reflected in the larger credible intervals . Even when we would knew the exact regression parameters, there is a lot of uncertainty in the LMG values caused by the uncertainty about the covariance matrix.

Table 3.5: LMG values assuming stochastic predictors with 95% CI.

	Relaimpo	**Bayesian framework**	
Variable		nonparameteric bootstrap	covariance inference
x1	0.092 (0.02, 0.221)	0.047 (0.005, 0.144)	0.078 (0.018, 0.17)
x2	0.109 (0.026, 0.241)	0.15 (0.055, 0.279)	0.156 (0.052, 0.273)
x3	0.636 (0.452, 0.757)	0.554 (0.395, 0.71)	0.561 (0.442, 0.66)
x4	0.03 (0.008, 0.11)	0.034 (0.008, 0.097)	0.033 (0.011, 0.103)

Table 3.6: Variable description

Variable Name	Description
paragrap	scores on paragraph comprehension test
general	scores on general information test
sentence	scores on sentence completion test
wordc	scores on word classification test
wordm	scores on word meaning test

3.2 Empirical Data

In the following section, the Bayesian LMG implementation is applied on an empirical dataset containing test scores of pupils (N = 301) from a study by Holzinger and Swineford (1939) available in the R package MBESS (Kelley, 2017). This dataset was used in Nimon *et al.* (2008) to present commonality analysis, which is another variance decomposition technique. Scores from a paragraph comprehension test (paragrap) was predicted by four verbal tests: general-information (general), sentence-comprehension (sentence) , word-classification (wordc), and word-meaning (wordm) (Table 3.6).

The aim of the regression analysis was to determine the strength of association between verbal ability and paragraph comprehension. An overview of the data is shown in Figure 3.3. The regression results of the paragraph comprehension scores on the verbal test scores are shown Table 3.7). A novice researcher may wrongly conclude, that there is little association between the *non-significant* predictors (general information and word classification) and paragraph comprehension. Given the other predictors are already included in the model, the predictors seem not to provide us much information about the expected paragraph comprehension ability. However, we can not conclude out of this regression table, that the association between any of these *non-significant* predictors and the dependent variable is unimportant. We see in Figure 3.3 that the correlations between the predictors is rather high. The LMG metric may therefore provide us some new information about the importance of each predictor.

The Bayesian regression model was fitted in `rstanarm`. The default priors were used for the regression coefficients and the σ^2 parameter. A burn-in period of 20000, a sample size of 20000 and a thinning of 20 resulted in a posterior sample size of 1000. The first few posterior samples are shown in Table 3.8. The resulting R^2 of of these posterior samples is shown in Table 3.9. The LMG values were calculated by using `hier.part`. The independent component (I), joint contribution (J) and total explained variance in a predictor-only model (T) are shown in Table 3.10. Sentence-comprehension and word-meaning seem to be the most important predictors by the LMG metric. However, none of the predictors seem to be unimportant. The joint contributions of each predictor is quite large.

For comparison purposes, the LMG metric was additionally calculated with the `relaimpo` package using parametric bootstrapping. The confidence intervals of `relaimpo` are almost identical to the credible intervals of the Bayesian framework (Table 3.11). Assuming stochastic or non-stochastic predictors would also result in almost identical uncertainty estimates with such a large sample size (Table3.12).

Table 3.7: Regression of paragraph comprehension on verbal tests.

	Coefficient	95%-confidence interval	*p*-value
Intercept	0.071	from -1.17 to 1.31	0.91
general	0.03	from -0.00 to 0.06	0.084
sentence	0.26	from 0.18 to 0.34	< 0.0001
wordc	0.047	from -0.01 to 0.11	0.14
wordm	0.14	from 0.08 to 0.19	< 0.0001

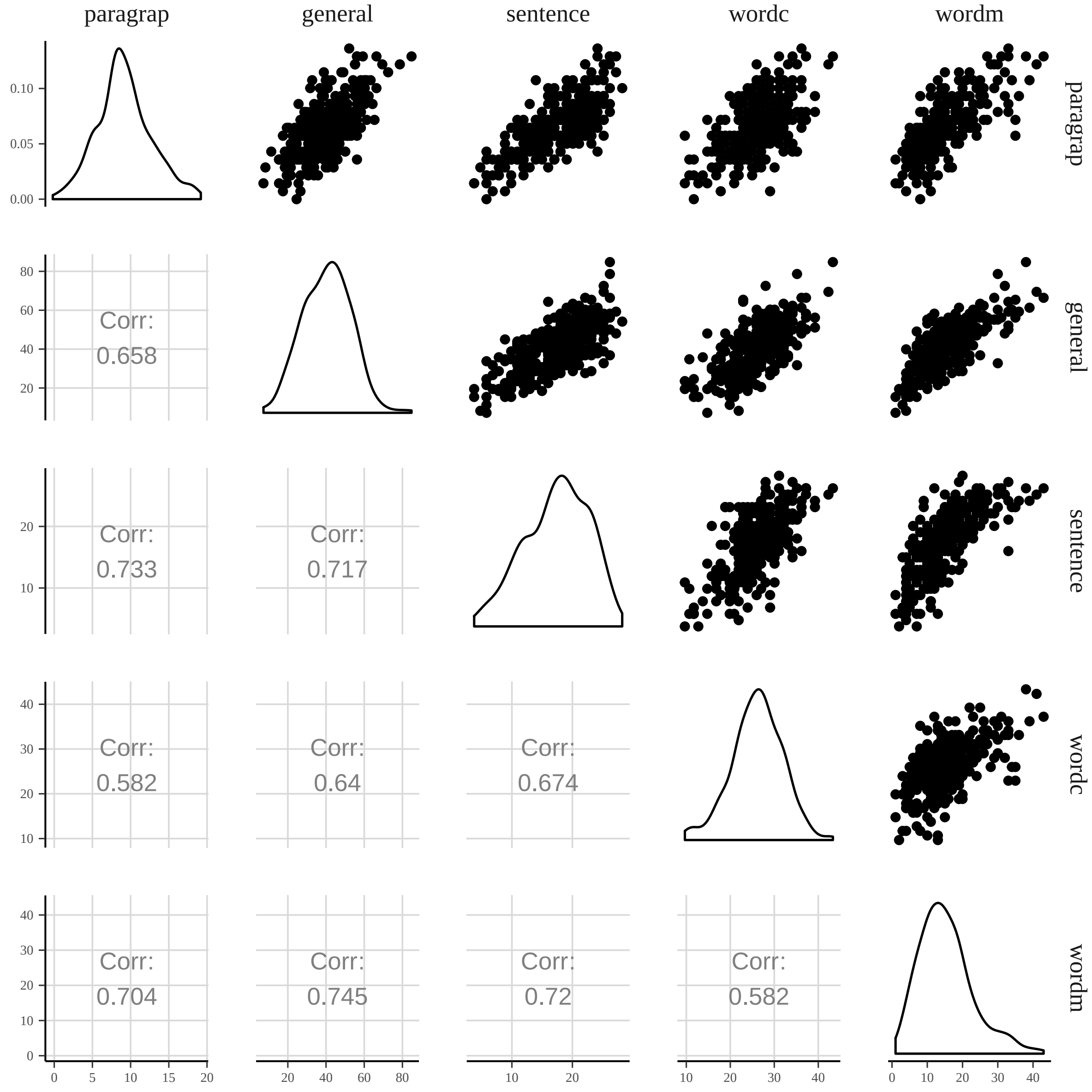

Figure 3.2: Test scores from Holzinger and Swineford's (1939) Study. N=301

Table 3.8: Samples from the posterior distributions of the regression parameters

	general	sentence	wordc	wordm	sigma
sample 1	0.041	0.270	0.014	0.113	2.18
sample 2	0.036	0.243	0.010	0.166	2.09
sample 3	0.043	0.295	-0.007	0.103	2.28
sample 4	0.024	0.240	0.051	0.192	2.19
sample 5	0.050	0.297	0.009	0.099	2.28
sample 6	0.027	0.259	0.040	0.157	2.09
sample 7	0.029	0.265	0.023	0.156	2.11
sample 8	0.017	0.259	0.026	0.168	2.18
sample 9	0.036	0.250	0.073	0.123	2.00
sample 10	0.004	0.269	0.059	0.163	2.10

Table 3.9: R^2 for all submodels for the first six posterior samples

	sample 1	sample 2	sample 3	sample 4	sample 5	sample 6
none	0.000	0.000	0.000	0.000	0.000	0.000
general	0.426	0.455	0.407	0.449	0.433	0.448
sentence	0.520	0.537	0.508	0.541	0.527	0.554
wordc	0.299	0.312	0.271	0.348	0.298	0.343
wordm	0.462	0.542	0.433	0.559	0.446	0.534
general sentence	0.558	0.583	0.541	0.583	0.566	0.592
general wordc	0.455	0.482	0.428	0.493	0.459	0.490
general wordm	0.511	0.578	0.482	0.588	0.504	0.569
sentence wordc	0.527	0.545	0.511	0.557	0.533	0.567
sentence wordm	0.574	0.627	0.551	0.640	0.571	0.633
wordc wordm	0.497	0.568	0.462	0.595	0.484	0.573
general sentence wordc	0.559	0.583	0.541	0.587	0.566	0.595
general sentence wordm	0.583	0.634	0.560	0.644	0.583	0.638
general wordc wordm	0.523	0.587	0.490	0.606	0.516	0.588
sentence wordc wordm	0.576	0.628	0.552	0.645	0.572	0.636
all	0.584	0.634	0.560	0.647	0.583	0.640

Table 3.10: Variance decomposition for non-stochastic predictors. I = LMG values, J = joint contribution, Total = total explained variance in one-predictor only model

Variable	**I**	**J**	**Total**
general	0.13 (0.105, 0.16)	0.299 (0.258, 0.333)	0.429 (0.364, 0.489)
sentence	0.203 (0.165, 0.245)	0.328 (0.293, 0.355)	0.532 (0.465, 0.588)
wordc	0.096 (0.073, 0.128)	0.24 (0.2, 0.275)	0.336 (0.273, 0.401)
wordm	0.176 (0.138, 0.214)	0.315 (0.278, 0.344)	0.492 (0.421, 0.55)

Table 3.11: Variance decomposition for non-stochastic predictors. I = LMG values, J = joint contribution, Total = total explained variance in one-predictor only model

	LMG value (95%-CI)	
Variable	Relaimpo	**Bayesian framework**
general	0.131 (0.106, 0.161)	0.13 (0.105, 0.16)
sentence	0.206 (0.168, 0.251)	0.203 (0.165, 0.245)
wordc	0.096 (0.073, 0.125)	0.096 (0.073, 0.128)
wordm	0.178 (0.141, 0.219)	0.176 (0.138, 0.214)

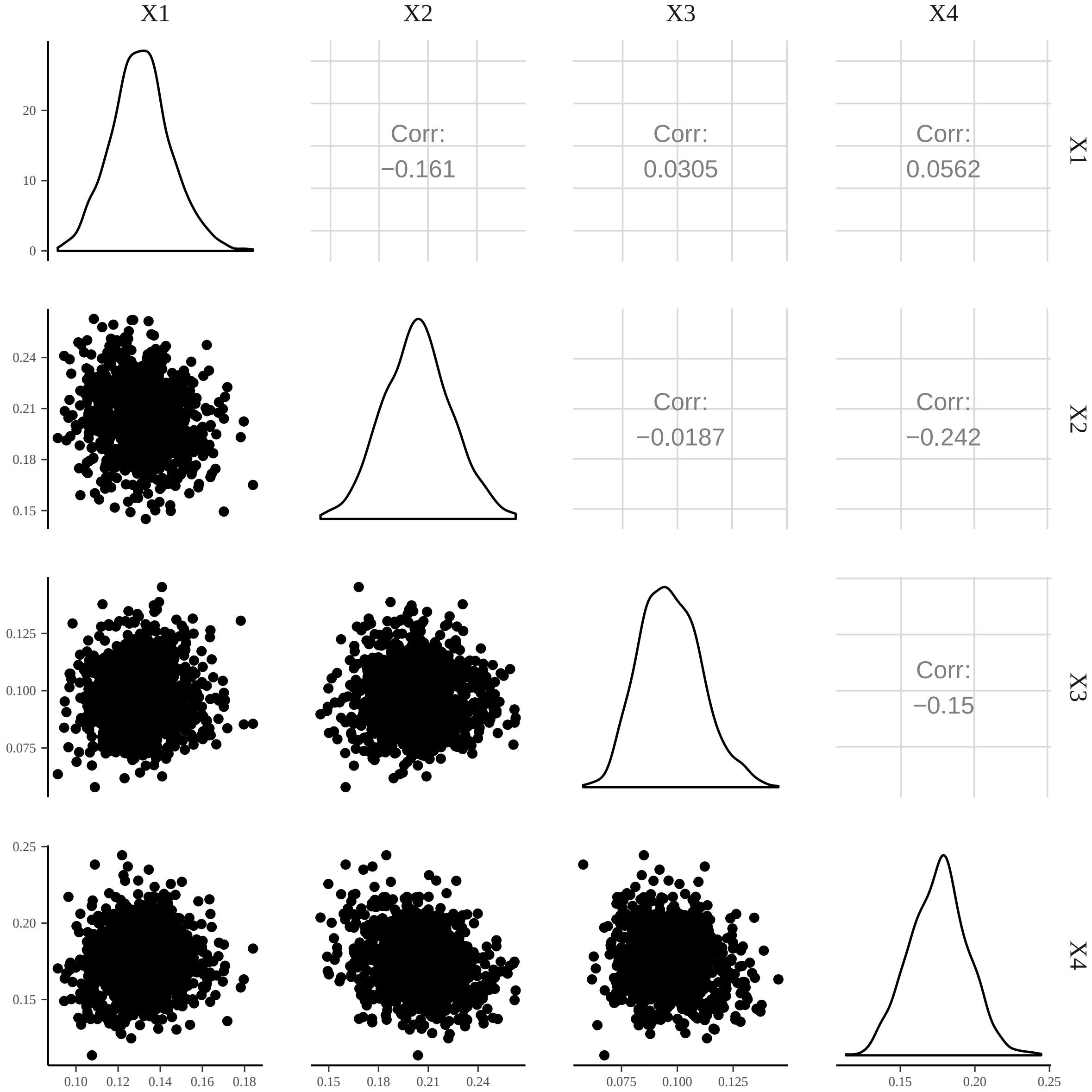

Figure 3.3: LMG posterior distribution of different verbal ability tests

Table 3.12: Variance decomposition for Stochastic predictors. I = LMG values, J = joint contribution, Total = total explained variance in one-predictor only model

	LMG value (95%-CI)	
Variable	Relaimpo	**Bayesian framework**
general	0.131 (0.106, 0.163)	0.127 (0.092, 0.16)
sentence	0.206 (0.171, 0.246)	0.204 (0.163, 0.249)
wordc	0.096 (0.075, 0.128)	0.097 (0.07, 0.128)
wordm	0.178 (0.144, 0.218)	0.175 (0.136, 0.214)

Chapter 4

Extension of the LMG formula for longitudinal data

In the following chapter some extenstions of the LMG formula beyond the simple linear regression model are shown. The focus is on repeated measurements models. These models extend the simple linear regression by allowing intra subject correlation between repeated measures.

The dependence within subjects can be modeled by including random effects (mixed model) or by assuming correlated errors within a subject (marginal model). Using a random intercept model or a compound symmetry matrix for the error will result in the same fixed effect estimate. A mixed model can be extended by including a random slope per subject, allowing for less restrictive longitudinal shapes. The marginal approach can get more freedom by different specified covariance matrices of the error terms. An unstructured covariance matrix, where no restriction are imposed, allows for the most freedom. However, depending on the number of repeated measuresments and the sample size the covariance matrix can get too large to make reasonable inference about it.

The extenstion of the LMG formula in the Bayesian framework to longitudinal models is restricted to models where the conditional variance formula can be applied to get the explained variance of the submodel from the regression parameters of the full model. The focus is therefore on the fixed predictors and not on the random effects. The conditional variance formula can be used in the marginal models, where only the fixed effects are modelled anyway. In the mixed model framework, the conditional variance formula is applicable to random intercepts models. For random-slope models there are atleast some difficulties involved, if it is possible at all, to get the explained variance of the submodel. This chapter shows the Bayesian LMG Implementation on a random intercept model and on a repeated measurement model with an unstructured covariance matrix.

4.1 random intercept model

The first example concerns a simple random intercept model with time varying predictors. Different R^2 metrics exist for linear mixed models (Nakagawa and Schielzeth, 2013). The variance of a random intercept model with regression parameter $\boldsymbol{\beta}$ can be written as

$$\mathrm{Var}(y) = \sigma_f^2 + \sigma_\alpha^2 + \sigma_\epsilon^2, \tag{4.1}$$

where $\sigma_f^2 = \mathrm{Var}(\mathbf{X}\boldsymbol{\beta}) = \boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{\mathbf{XX}} \boldsymbol{\beta}$, σ_α^2 is the random intercept and σ_ϵ^2 is the error term.

An R^2 that is guaranteed to be positive is defined in Nakagawa and Schielzeth (2013) as

$$\mathrm{R}^2_{\mathrm{LMM}} = \frac{\sigma^2_f}{\sigma^2_f + \sigma^2_\alpha + \sigma^2_\epsilon}, \tag{4.2}$$

It is theoretically possible that the $\mathrm{R}^2_{\mathrm{LMM}}$ decreases when adding predictors (Nakagawa and Schielzeth, 2013). By adding predictors σ^2_f should always increase and σ^2_ϵ decrease. However, the σ^2_α may also increase a little bit and the total R^2 may then be a little bit lower. The R^2 can not decrease by using the conditional variance formula on the full model to calculate the R^2 of the submodels because the total variance is fixed. The results will be the same as if we would fit a new model by maximum likelihood for each submodel and only compare the variance of the fixed effects to the explained variance of the full model. In the Bayesian framework we need the conditional variance formula to account for the interdepdence of the submodels to each other. The total variance of the full model can be calculated as $\mathrm{Var}(y) = \mathrm{Var}(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}b) + \sigma^2$ or by using samples of σ^2_α as in (4.1). The error term could again be sampled or calculated as in (2.6). In the following examples, (4.1) is used and σ^2_α and σ^2_ϵ are sampled from their posterior distribution.

In repeated measurement studies the focus is often in within-subject changes. The between subject variance estimated with the random intercept term is of minor importance. The more important question may be how much the fixed predictors explain compared to the within subject error, which is

$$\mathrm{R}^2_{\mathrm{repeated}} = \frac{\sigma^2_f}{\sigma^2_f + \sigma^2_\epsilon}, \tag{4.3}$$

The square root of this term is known under the name correlation within subjects in Bland and Altman (1995). Often, there are between-subject and within-subject predictors in a model. If we are interested in the within-subject effect, we can use a model including only the between-subject predictors as the null-model.

The following example shows a simple random intercept model with time-varying predictors. The main question is which within-subject predictors are the most important ones. The between-subject variance is of minor importance.

The data are simulated from the following regression setting with $m = 4$ timepoints and $n = 20$ number of subjects ,

$$Y_{i,j} \sim \mathcal{N}(\beta_0 + x_{1_{i,j}}\beta_1 + x_{2_{i,j}}\beta_2 + x_{3_{i,j}}\beta_3 + x_{4_{i,j}}\beta_4 + \alpha_i,\, \sigma^2), \qquad \begin{array}{l} i = 1, \ldots, n \\ j = 1, \ldots, m \end{array}$$

where $\beta_1 = 1\,,\ \beta_2 = 1\,,\ \beta_3 = 2\ \ \beta_4 = 2\,,\ \sigma^2 = 1\,,\ \alpha_i \sim \mathcal{N}(0, \sigma^2_\alpha)\,,\ \mathbf{X} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$.

The R code can be found in the appendix:

The random intercept effect is not of interest. The R^2 of the models is calculated according to the formula of repeated measure correlation (4.3). Most of the within subject variance is explained by the predictors (Table 4.1). The credible intervals are very narrow. For the information about the between subject varinace could look at the random intercept variance directly.

In the second part the random intercept is dirctly included in the total variance calculation of the R^2 values. There is a large between-subject variance in this dataset (Table 4.2). The LMG values including the between subject variance are much lower. The credible intervals are also much wider, because the uncertainty about the between-subject variance is included.

In my opinion we can get more useful infromation from separating the between and within variance components when it is easy possible. Note that we assumed non stochastic predictors. Otherwise, the credible intervals would be larger. In general, it seems more reasonable to assume stochastic time-varying predictors. The variance could then be estimated by non-parameteric bootstrap, resampling whole subjects (all repeated measurements of a subject).

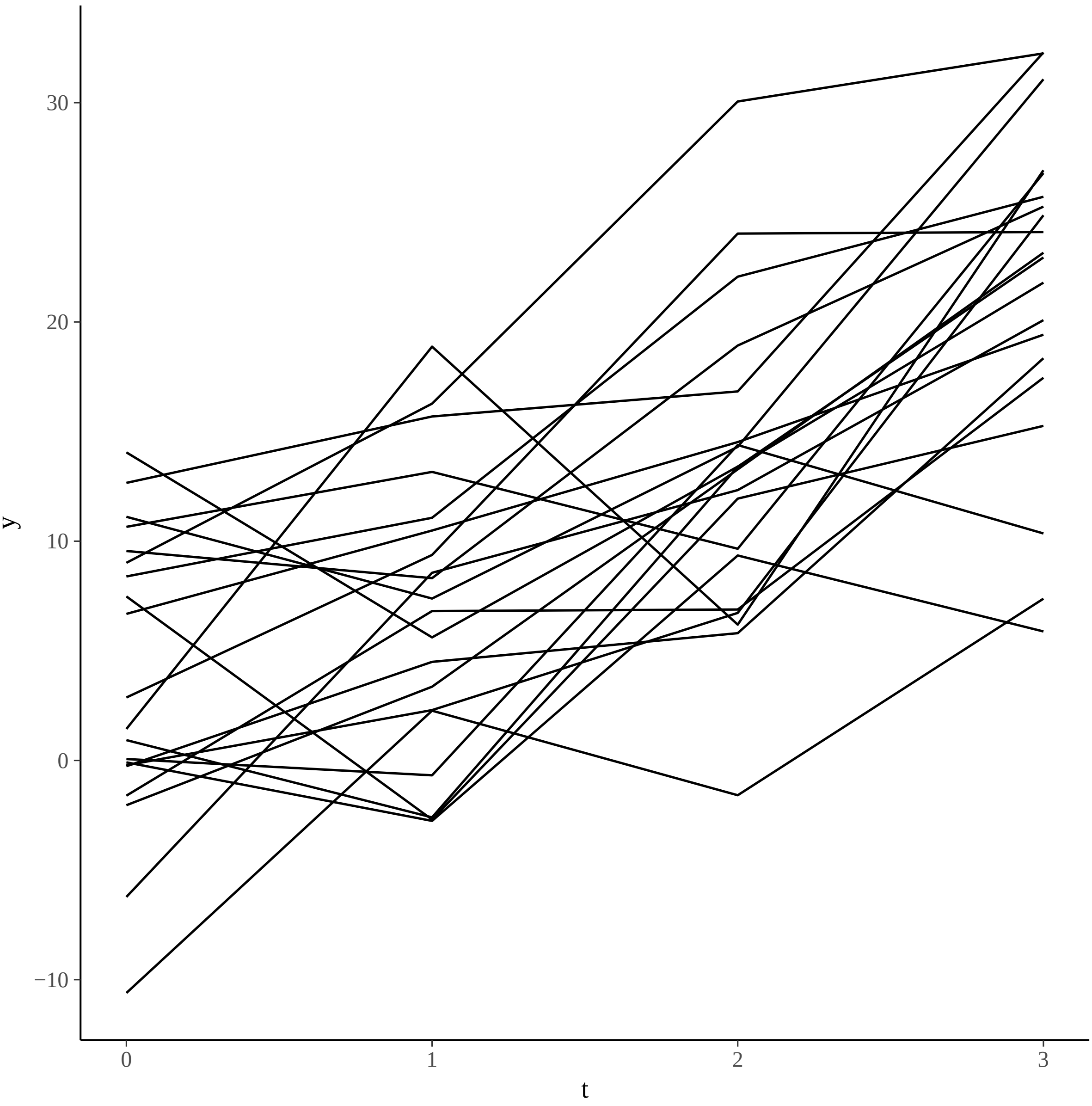

Figure 4.1: Individual trajectories of simulated random intercept model

Table 4.1: Variance decomposition for non-stochastic predictors. I = LMG values, J = joint contribution, Total = total explained variance in one-predictor only model

Variable	I	J	Total
x1	0.202 (0.2, 0.203)	0.525 (0.523, 0.526)	0.727 (0.723, 0.73)
x2	0.21 (0.208, 0.211)	0.538 (0.536, 0.539)	0.747 (0.744, 0.751)
x3	0.297 (0.296, 0.299)	0.658 (0.658, 0.658)	0.955 (0.954, 0.957)
x4	0.291 (0.29, 0.292)	0.651 (0.65, 0.651)	0.942 (0.94, 0.943)

4.2 marginal model

The next example concerns a repeated measurement model with time-varying predictors an unstructured covariance error covariance matrix. The data are generated from the following model:

Table 4.2: Variance decomposition for non-stochastic predictors. I = LMG values, J = joint contribution, Total = total explained variance in one-predictor only model

Variable	I	J	Total
x1	0.108 (0.046, 0.172)	0.238 (0.116, 0.35)	0.346 (0.163, 0.522)
x2	0.089 (0.046, 0.127)	0.229 (0.116, 0.329)	0.318 (0.162, 0.456)
x3	0.113 (0.06, 0.161)	0.272 (0.139, 0.391)	0.385 (0.199, 0.551)
x4	0.115 (0.061, 0.163)	0.27 (0.137, 0.387)	0.385 (0.199, 0.55)

$$Y_i \sim \mathcal{N}(\mathbf{X}_i\boldsymbol{\beta}, \boldsymbol{\Sigma}), \qquad i = 1, \dots, n \tag{4.4}$$

where $\boldsymbol{\Sigma}$ represents an unstructured error covariance matrix, $\mathbf{X}_i$ represents the predictor matrix of subject i.

In the variance calculation we need to take into account that we do not have just one σ^2 parameter, but a covariance matrix $\boldsymbol{\Sigma}$. The diagonal elements of $\boldsymbol{\Sigma}$ represent the variance of each timepoint. The sum of the diagonal elements of $\boldsymbol{\Sigma}$ represents the variance for a whole subject. When there are no missing timepoints in each subject, we can take the mean of diag($\boldsymbol{\Sigma}$) to make the formula compatiable with the $\boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{\mathbf{XX}}\boldsymbol{\beta}$ of (2.8), resulting in the total variance term

$$\text{Var}(\mathbf{Y}) = \boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{\mathbf{XX}}\boldsymbol{\beta} + \text{mean}(\text{diag}(\boldsymbol{\Sigma})), \tag{4.5}$$

The following R-code is used to generate the data:

The individual trajectories are shown in Figure 4.2. The resulting LMG values of the predictors are shown in Table 4.3.

Table 4.3: Variance decomposition for non-stochastic predictors. I = LMG values, J = joint contribution, Total = total explained variance in one-predictor only model

Variable	I	J	Total
x1	0.141 (0.113, 0.163)	0.364 (0.299, 0.406)	0.506 (0.413, 0.568)
x2	0.137 (0.11, 0.16)	0.353 (0.292, 0.395)	0.49 (0.405, 0.554)
x3	0.206 (0.168, 0.233)	0.443 (0.366, 0.489)	0.65 (0.537, 0.719)
x4	0.21 (0.172, 0.239)	0.444 (0.367, 0.489)	0.656 (0.543, 0.725)

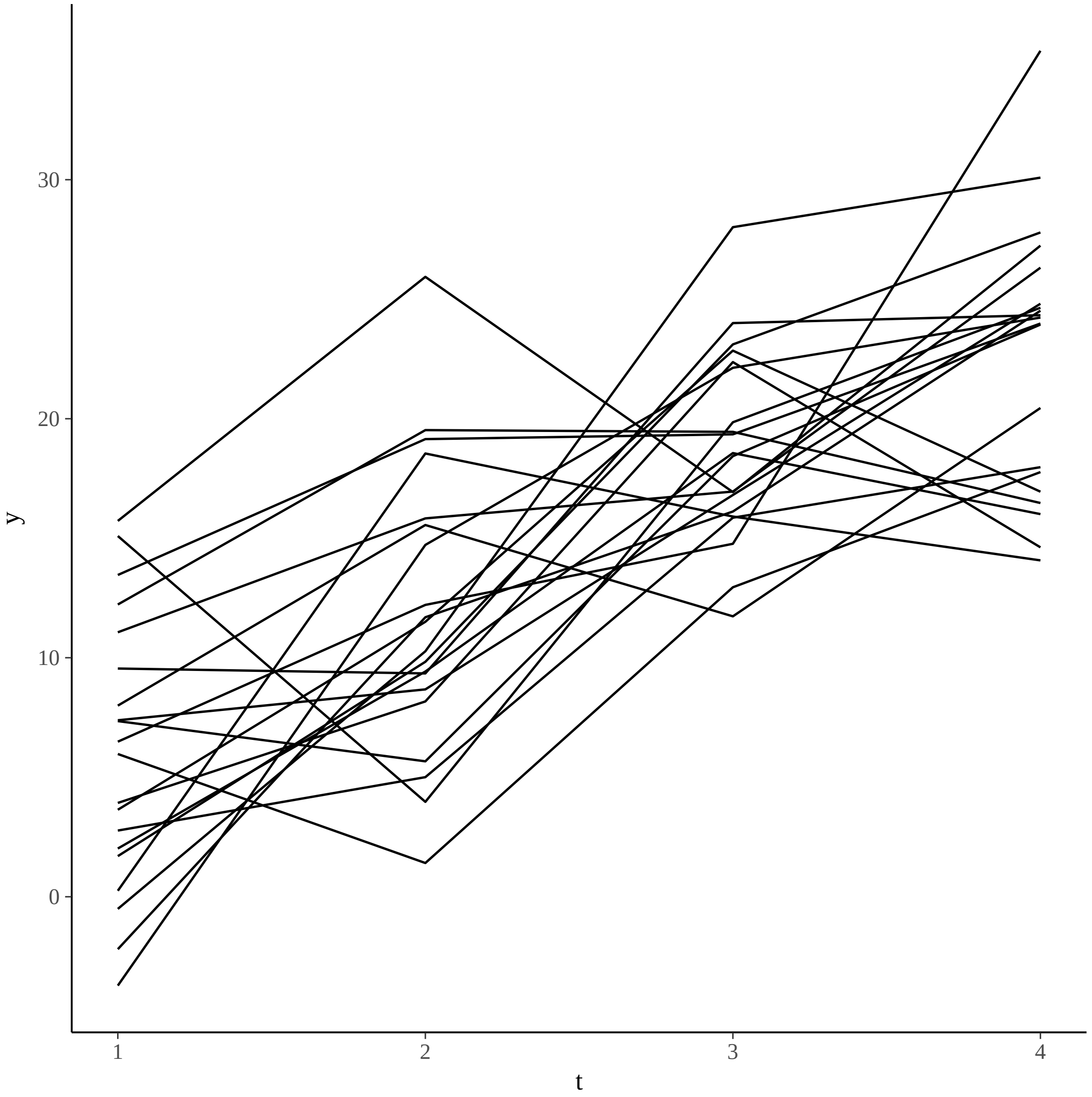

Figure 4.2: Individual trajectories of simulated data with unstructured error covariance matrix

Chapter 5

Other variable importance metrics in the Bayesian framework

Different variable importance metrics exist. The conditional variance formula allowed us to calculate the R^2 of the submodels from the posterior sample of the full model. The focus of this master thesis was on the LMG formula. For each posterior sample the LMG formula can be applied for the submodels of each posterior sample. A lot of the variable importance metrics are based on the R^2 of the full model compared to the submodels. Instead of the LMG formula we could as well have used another variable importance metric after we have calculated the R^2 of all the submodels. Commonality and dominance analysis seem to be interesting extensions of the LMG framework. Both provide besides the LMG information some other information about the variance decomposition of the predictors. The relaimpo package provides some more bootstrap options like pariwise differences that could also be easily transfered to the Bayesian framework.

5.1 Conclusion

The Bayesian framworks provides us with a nice option to include the uncertainty about parameters. Posterior distributions of the parameters allow us to calculate a distribution of R^2 values for each model. Using the conditional variance formula allowd us to calculate the R^2 of all the submodels from the posterior parameter distributions of the full model. Instead of fitting 2^{p-1} models, only the full model needs to be fitted. Doing it this way has some nice properties. The interdependence of the submodels to each other is respected. The R^2 of the submodels does not decrease when adding predictors. The non-negativity of the shares is therefore respected in the LMG formula. This property of using the conditional variance formula is also interesting when the LMG formula is applied to random intercept models fitted by maximum likelihood.

A disadventage about calculating the R^2 of all the submodels with the conditional variance formula seems to be the restriction to the linear model. Although this may be a topic of further research.

Assuming non-stochastic or stochastic predictors can have a big impact on the the uncertainty of the explained variance and the LMG values. Although the posterior regression parameter distribtuions is the same in both cases (under some assumptions described in chapter ...) the explainened variance of a model is directly dependent on the covarinace matrix. Inference about the covariance of the predictors $\mathbf{X}$ is therefore an important part when stochastic predictors are assumed. However, this does in general not seem to be an easy problem. Non-parametric bootstrap provides a practical solution.

Chapter 6

Appendix

he eror term in the by Gelman *et al.* (2017) proposed definition of the R^2 is defined as $\mathrm{Var}(\sum_{i=1}^{n} e_n^s)$. I think we could also use $\sum(y-\hat{y}^s)^2/(n-1)$ as an estimate for the error. For the maximum likelihood estimate $\mathrm{Var}(y_i - \hat{y}_i) = \sum(y_i - \hat{y}_i)^2/(n-1)$. This is because the mean of the residuals is 0. When samples of the posterior parameters are used, the mean of the residuals is not excatly zero. $\mathrm{Var}(y_i - \hat{y}_i) = \sum(y_i - \hat{y}_i)^2/(n-1)$ is than a little bit bigger than $\mathrm{Var}(y_i - \hat{y}_i)$. In practice the values should only differ by a very small amount. We do not expect the errors to have a systematic bias. However, the residuals are just a sample of the error. The mean of the residuals must not be excatly 0 when the samples of the posteriors are used for the regression coefficients.

Appendix A

Formula

```r
set_parent('Main.Rnw')
library(knitr)
opts_chunk$set(
    fig.path='figure/ch01_fig',
    self.contained=FALSE,
    cache=TRUE
)

set_parent('Main.Rnw')
library(knitr)
opts_chunk$set(
    fig.path='figure/ch02_fig',
    self.contained=FALSE,
    cache=!FALSE
)
set_parent('Main.Rnw')
library(knitr); library(hier.part); library(rstanarm);  library(GGally); library(ggplot2); libr


opts_chunk$set(
    fig.path='figure/ch03_fig',
    self.contained=FALSE,
    cache=TRUE
)

options(digits =3)
#function to calculate R2 data frame
rtwos<-function(X, post.sample){
  # X: Predictor data as data frame
  # post.sample: posterior sample as matrix (M[sample_i,]), last position should be sigma param

  X <- cov(X) #covariance matrix

  #Prepare data frame and rownames by using the combn() function

  lst <- list()
```

```
pcan <- dim(X)[2]
n <- (2^pcan)-1

for (i in 1:pcan){
  lst[[i]] <- combn(pcan,i)
}

var.names <- character(length = 0)

v<- rownames(X)

for(i in 1:length(lst))
{

  for (j in 1:ncol(lst[[i]])){
    cur<- lst[[i]][,j]
    name <- paste0(v[-cur])
    name <- paste(name, collapse = " ")
    var.names <- c(var.names, name)
  }
}

var.names<-c(rev(var.names), 'all')

var.names[1]<-'none'

size <- nrow(post.sample)  # how many samples

sig.posi <- ncol(post.sample)

df.Rtwos<-data.frame(matrix(0,n+1,size))

row.names(df.Rtwos) <- var.names

########### fill in R^2 matrix, use posterior samples and calculate submodels according t

v <- 1:dim(X)[2]

for (s in 1:size){

  sample.s <- post.sample[s,-sig.posi]

  Vtot <- sample.s%*%X%*%sample.s  #total explained variance

  count=n

  for(i in 1:(length(lst)-1))
  {

    for (j in 1:ncol(lst[[i]])){
      cur <- lst[[i]][,j]
```

```
        set <- v[-cur]
        matr <- X[cur, cur] - X[cur, set]%*%solve(X[set, set])%*%X[set, cur]  #conditional expe
        var.explain<- sample.s[cur]%*%matr%*%sample.s[cur]  #multiply by parameter sample
        df.Rtwos[count,s] <- Vtot - var.explain
        count=count-1
      }
    }

    df.Rtwos[n+1,s] <- Vtot

    df.Rtwos[,s] <- df.Rtwos[,s]/c(Vtot+post.sample[s,sig.posi]^2) #total variance + sigma^2 (p
  }

  return(df.Rtwos)
}



bootcov <- function(df, boot.n){
        len <- nrow(df)
        cov.m <- cov(df)
        l <- dim(cov.m)[1]
        M.boot <- array(NA, c(l,l,boot.n))
        M.boot[,,1] <- cov(df)
        for (i in 2 :boot.n){
                dfs <- df[sample(1:len, replace=T),]
                M.boot[,,i] <- cov(dfs)
        }

        return(M.boot)
}


#Function that includes uncertainty about stochastic predictors by bootstrapping using bootcov
#n is the number of bootstrap samples it should calculate
# takes boot.n times longer to calculate.
rtwos.boot<-function(df, post.sample, boot.n){
        # X: Predictor data as data frame
        # post.sample: posterior sample as matrix (M[sample_i,]), last position should be sigma

        X <- cov(df) #covariance matrix

        #Prepare data frame and rownames by using the combn() function

        lst <- list()

        pcan <- dim(X)[2]
        n <- (2^pcan)-1

        for (i in 1:pcan){
                lst[[i]] <- combn(pcan,i)
        }

        var.names <- character(length = 0)

        v<- rownames(X)

        for(i in 1:length(lst))
        {

                for (j in 1:ncol(lst[[i]])){
                        cur<- lst[[i]][,j]
                        name <- paste0(v[-cur])
                        name <- paste(name, collapse = " ")
                        var.names <- c(var.names, name)
                }
        }

        var.names<-c(rev(var.names), 'all')

        var.names[1]<-'none'

        size <- nrow(post.sample)  # how many samples

        sig.posi <- ncol(post.sample)

        df.Rtwos<-array(0,c(n+1,size, boot.n))

        dimnames(df.Rtwos)[[1]] <- var.names

        boot.M <- bootcov(df, boot.n)

        v <- 1:dim(X)[2]

        for (b in 1:boot.n){


                X <- boot.M[,,b]

        ########### fill in R^2 matrix, use posterior samples and calculate submodels accor

        for (s in 1:size){

                sample.s <- post.sample[s,-sig.posi]

                Vtot <- sample.s%*%X%*%sample.s  #total explained variance

                count=n

                for(i in 1:(length(lst)-1))
                {
                        for (j in 1:ncol(lst[[i]])){
                                cur <- lst[[i]][,j]
                                set <- v[-cur]
                                matr <- X[cur, cur] - X[cur, set]%*%solve(X[set, set])%*%X[set,
                                var.explain<- sample.s[cur]%*%matr%*%sample.s[cur]  #multiply t
                                df.Rtwos[count,s,b] <- Vtot - var.explain
                                count=count-1
                        }
                }

                df.Rtwos[n+1,s,b] <- Vtot

                df.Rtwos[,s,b] <- df.Rtwos[,s,b]/c(Vtot+post.sample[s,sig.posi]^2) #total varia
        }

        }
        return(df.Rtwos)
}


#Function to include samples of the predictor covariance matrix.

rtwos.covm<-function(df, post.sample, covm, boot.n){
        # X: Predictor data as data frame
        # post.sample: posterior sample as matrix (M[sample_i,]), last position should be sigma

        X <- cov(df) #covariance matrix

        #Prepare data frame and rownames by using the combn() function

        lst <- list()

        pcan <- dim(X)[2]
        n <- (2^pcan)-1

        for (i in 1:pcan){
                lst[[i]] <- combn(pcan,i)
        }

        var.names <- character(length = 0)

        v<- rownames(X)

        for(i in 1:length(lst))
        {

                for (j in 1:ncol(lst[[i]])){
                        cur<- lst[[i]][,j]
                        name <- paste0(v[-cur])
                        name <- paste(name, collapse = " ")
                        var.names <- c(var.names, name)
                }
        }

        var.names<-c(rev(var.names), 'all')

        var.names[1]<-'none'

        size <- nrow(post.sample)  # how many samples

        sig.posi <- ncol(post.sample)

        df.Rtwos<-array(0,c(n+1,size, boot.n))

        dimnames(df.Rtwos)[[1]] <- var.names

        v <- 1:dim(X)[2]


        for (b in 1:boot.n){


                X <- covm[,,b]

                ########### fill in R^2 matrix, use posterior samples and calculate submode

                for (s in 1:size){

                        sample.s <- post.sample[s,-sig.posi]

                        Vtot <- sample.s%*%X%*%sample.s  #total explained variance

                        count=n

                        for(i in 1:(length(lst)-1))
                        {

                                for (j in 1:ncol(lst[[i]])){
                                        cur <- lst[[i]][,j]
                                        set <- v[-cur]
                                        matr <- X[cur, cur] - X[cur, set]%*%solve(X[set, se
                                        var.explain<- sample.s[cur]%*%matr%*%sample.s[cur]
                                        df.Rtwos[count,s,b] <- Vtot - var.explain
                                        count=count-1
                                }
                        }

                        df.Rtwos[n+1,s,b] <- Vtot

                        df.Rtwos[,s,b] <- df.Rtwos[,s,b]/c(Vtot+post.sample[s,sig.posi]^2)
                }
        }
        return(df.Rtwos)
}


x1 <- rnorm(50, 0, 1); x2 <- rnorm(50, 0, 1)
x3 <- rnorm(50, 0, 1); x4 <- rnorm(50, 0, 1)
b1 <- 0.5; b2 <- 1; b3 <- 2; b4 <- 0.8
#b1 <- 1; b2 <- 1; b3 <- 1; b4 <- 1

y <- b1*x1 + x2*b2 + b3*x3 + b4*x4 + rnorm(50, 0, 1)

df <- data.frame(y = y, x1 = x1, x2 = x2, x3 = x3, x4 = x4)

post2 <- stan_glm(y ~ 1 + x1 + x2 + x3 + x4,
                  data = df,
                  chains = 1, cores =1, iter=40000, thin=20)



#posterior sample
post.sample <- as.matrix(post2)

dt <- data.frame(post.sample[1:10,2:6])
rownames(dt)<-c('sample 1', 'sample 2', 'sample 3', 'sample 4', 'sample 5', 'sample 6', 'sample
#example of the first 10 posterior samples
kable(dt, format = "latex", booktabs=TRUE, linesep = c("", "", "", ""), caption = 'Samples from
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))

#no need for the intercept, last parameter is sigma
post.sample <- post.sample[,-1]
#data frame with all submodels
df.rtwos <-rtwos(df[,2:5], post.sample)

colnames(df.rtwos)[1:6] <- c('sample 1', 'sample 2', 'sample 3', 'sample 4', 'sample 5', 'sampl
kable(df.rtwos[,1:6], format = 'latex', booktabs=TRUE, linesep = c("", "", "", ""), caption = '
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))

# prepare data frame for LMG values

LMG.Vals.I<-matrix(0, 4, dim(df.rtwos)[2])

LMG.Vals.J<-matrix(0, 4, dim(df.rtwos)[2])

LMG.Vals.T<-matrix(0, 4, dim(df.rtwos)[2])


for(i in 1:dim(df.rtwos)[2]){

  gofn<-df.rtwos[,i]
  obj.Gelman<-partition(gofn, pcan = 4, var.names = names(df[,2:5]))

  LMG.Vals.I[,i]=obj.Gelman$IJ[,1]
  LMG.Vals.J[,i]=obj.Gelman$IJ[,2]
        LMG.Vals.T[,i]=obj.Gelman$IJ[,3]
}

varnames <- row.names(obj.Gelman$IJ)


# posterior LMG distribution of each variable
quantile(LMG.Vals.I[1,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.I[2,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.I[3,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.I[4,], c(0.025, 0.5, 0.975))

quantile(LMG.Vals.J[1,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.J[2,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.J[3,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.J[4,], c(0.025, 0.5, 0.975))

quantile(LMG.Vals.T[1,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.T[2,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.T[3,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.T[4,], c(0.025, 0.5, 0.975))

# some example how it could be displayed
dat <- data.frame(t(LMG.Vals.I))

pairs.chart <- ggpairs(dat, lower = list(list(combo = "facetdensity")), upper = list(list(c
pairs.chart



# Comparison to relaimpo package

fit <- lm(y~., data=df)

######## compare to relimp package

run<-boot.relimp(fit, fixed=TRUE)

ci <- booteval.relimp(run, bty = "perc", level = 0.95,
                sort = FALSE, norank = TRUE, nodiff = TRUE,
                typesel = c("lmg"))@mark

ci <- as.matrix(ci)
ci <- matrix(as.numeric(ci), 4 , 3)
colnames(ci) <- c('LMG', 'l.b', 'u.b')
rownames(ci) <- c('X1', 'X2', 'X3', 'X4')
```

```
#Code assuming stochastic predictors
run.stochastic<-boot.relimp(fit, fixed=FALSE)

ci.s <- booteval.relimp(run.stochastic, bty = "perc", level = 0.95,
                sort = FALSE, norank = TRUE, nodiff = TRUE,
                typesel = c("lmg"))@mark

ci.s <- as.matrix(ci.s)
ci.s <- matrix(as.numeric(ci.s), 4 , 3)
colnames(ci.s) <- c('LMG', 'l.b', 'u.b')
rownames(ci.s) <- c('X1', 'X2', 'X3', 'X4')


pairs.chart



#-----------------------------------------------------------------------------------------


df.rtwos.boot <-rtwos.boot(df[,2:5], post.sample, 10)

n.boot = 10

LMG.Vals.I.boot<-array(0, c(4,dim(df.rtwos.boot)[2], n.boot))
LMG.Vals.J.boot<-array(0, c(4,dim(df.rtwos.boot)[2], n.boot))
LMG.Vals.T.boot<-array(0, c(4,dim(df.rtwos.boot)[2], n.boot))

for (b in 1:n.boot){

for(i in 1:dim(df.rtwos.boot)[2]){

        gofn<-df.rtwos.boot[,i,b]

        obj.Gelman<-partition(gofn, pcan = 4, var.names = names(df[,2:5]))

        LMG.Vals.I.boot[,i, b]=obj.Gelman$IJ[,1]
        LMG.Vals.J.boot[,i, b]=obj.Gelman$IJ[,2]
        LMG.Vals.T.boot[,i, b]=obj.Gelman$IJ[,3]

}

}

quantile(c(LMG.Vals.I.boot[1,1:1000,]), c(0.025, 0.5, 0.975))
quantile(c(LMG.Vals.I.boot[2,1:1000,]), c(0.025, 0.5, 0.975))
quantile(c(LMG.Vals.I.boot[3,1:1000,]), c(0.025, 0.5, 0.975))
quantile(c(LMG.Vals.I.boot[4,1:1000,]), c(0.025, 0.5, 0.975))

#very similar values as in the confidence intervals for stochastic predictors
t1 <- c(LMG.Vals.I.boot[1,,])
```

```
t2 <- c(LMG.Vals.I.boot[2,,])
t3 <- c(LMG.Vals.I.boot[3,,])
t4 <- c(LMG.Vals.I.boot[4,,])

dat.b<-data.frame(cbind(t1,t2,t3,t4))
cor(dat.b)



zy = df[,2:5]

#----------------------------------------------------------------------------
# In the following example we know that the $\X$ are coming from a normal distribution. The
#----------------------------------------------------------------------------

# Load some functions used below:
# Install the ellipse package if not already:
# Standardize the data:

#zy = apply(y,2,function(yVec){(yVec-mean(yVec))/sd(yVec)})
# Assemble data for sending to JAGS:
dataList = list(
        zy = zy ,
        Ntotal =  nrow(zy) ,
        Nvar = ncol(zy) ,
        # Include original data info for transforming to original scale:
        # For wishart (dwish) prior on inverse covariance matrix:
        zRscal = ncol(zy) ,  # for dwish prior
        zRmat = diag(x=1,nrow=ncol(zy))  # Rmat = diag(apply(y,2,var))
)

# Define the model:
modelString = "
model {
for ( i in 1:Ntotal ) {
zy[i,1:Nvar] ~ dmnorm( zMu[1:Nvar] , zInvCovMat[1:Nvar,1:Nvar] )
}
for ( varIdx in 1:Nvar ) { zMu[varIdx] ~ dnorm( 0 , 1/2^2 ) }
zInvCovMat ~ dwish( zRmat[1:Nvar,1:Nvar] , zRscal )
# Convert invCovMat to sd and correlation:
zCovMat <- inverse( zInvCovMat )

}
" # close quote for modelString
writeLines( modelString , con="Jags-MultivariateNormal-model.txt" )

# Run the chains:
nChain = 3
nAdapt = 500
nBurnIn = 500
nThin = 10
```

```
nStepToSave = 20000
require(rjags)
jagsModel = jags.model( file="Jags-MultivariateNormal-model.txt" ,

update( jagsModel , n.iter=nBurnIn )
codaSamples = coda.samples( jagsModel ,





# Convergence diagnostics:
parameterNames = varnames(codaSamples) # get all parameter names


# Examine the posterior distribution:
mcmcMat = as.matrix(codaSamples)
chainLength = nrow(mcmcMat)

covMat <- array(NA, c(4,4,chainLength))

for (i in 1:chainLength){
covMat[1:4,1:4,i]<-matrix(mcmcMat[i,], 4, 4)
}

covMat <- covMat[1:4,1:4,sample(1:20000, replace=F)]


df.rtwos <-rtwos.covm(df[,2:5], post.sample, covMat, 10)


n.boot = 10

LMG.Vals.I.covm<-array(0, c(4,dim(df.rtwos)[2], n.boot))
LMG.Vals.J.covm<-array(0, c(4,dim(df.rtwos.boot)[2], n.boot))
LMG.Vals.T.covm<-array(0, c(4,dim(df.rtwos.boot)[2], n.boot))


for (b in 1:n.boot){

        for(i in 1:dim(df.rtwos)[2]){

                gofn<-df.rtwos[,i,b]

                obj.Gelman<-partition(gofn, pcan = 4, var.names = names(df[,2:5]))

                LMG.Vals.I.covm[,i, b]=obj.Gelman$IJ[,1]
                LMG.Vals.J.covm[,i, b]=obj.Gelman$IJ[,2]
                LMG.Vals.T.covm[,i, b]=obj.Gelman$IJ[,3]

        }
```

```r
}

quantile(c(LMG.Vals.I.covm[1,1:1000,]), c(0.025, 0.5, 0.975))
quantile(c(LMG.Vals.I.covm[2,1:1000,]), c(0.025, 0.5, 0.975))
quantile(c(LMG.Vals.I.covm[3,1:1000,]), c(0.025, 0.5, 0.975))
quantile(c(LMG.Vals.I.covm[4,1:1000,]), c(0.025, 0.5, 0.975))





#How much variance is effectively in the bootstrap matrix when we know the regression param

#fake post sample

x1 <- rnorm(50, 0, 1); x2 <- rnorm(50, 0, 1)
x3 <- rnorm(50, 0, 1); x4 <- rnorm(50, 0, 1)
#b1 <- 0.5; b2 <- 1; b3 <- 2; b4 <- 0
b1 <- 1; b2 <- 1; b3 <- 1; b4 <- 1

y <- b1*x1 + x2*b2 + b3*x3 + b4*x4 + rnorm(50, 0, 1)

df <- data.frame(y = y, x1 = x1, x2 = x2, x3 = x3, x4 = x4)

post.sample <- matrix(1, 1000,5)

df.rtwos.boot <-rtwos.boot(df[,2:5], post.sample, 10)

n.boot = 10

LMG.Vals.I.boot<-array(0, c(4,dim(df.rtwos.boot)[2], n.boot))

for (b in 1:n.boot){

for(i in 1:dim(df.rtwos.boot)[2]){

        gofn<-df.rtwos.boot[,i,b]

        obj.Gelman<-partition(gofn, pcan = 4, var.names = names(df[,2:5]))

        LMG.Vals.I.boot[,i, b]=obj.Gelman$IJ[,1]
}

}

quantile(c(LMG.Vals.I.boot[1,1:1000,]), c(0.025, 0.5, 0.975))
quantile(c(LMG.Vals.I.boot[2,1:1000,]), c(0.025, 0.5, 0.975))
quantile(c(LMG.Vals.I.boot[3,1:1000,]), c(0.025, 0.5, 0.975))
quantile(c(LMG.Vals.I.boot[4,1:1000,]), c(0.025, 0.5, 0.975))
```

```
cov(df[,2:5])

#Comparison of sample covariance and shrink covariance matrix

cov(df[,2:5])
cov.shrink(df[,2:5])

data('HS.data', package='MBESS')
hs.data <- HS.data[,c('paragrap','general', 'sentence' , 'wordc' , 'wordm')]
pairs.chart <- ggpairs(hs.data, lower = list(continuous = "cor"), upper = list(continuous = "po
pairs.chart



bayes.hs <- stan_glm(paragrap ~ . ,
                  data = hs.data,
                  chains = 1, cores = 1, iter=40000, thin=20)

# Comparison to relaimpo package



fit <- lm(paragrap~., data= hs.data)


######## compare to relimp package

run<-boot.relimp(fit, fixed=TRUE)

ci <- booteval.relimp(run, bty = "perc", level = 0.95,
                sort = FALSE, norank = TRUE, nodiff = TRUE,
                typesel = c("lmg"))@mark

ci <- as.matrix(ci)
ci <- matrix(as.numeric(ci), 4 , 3)
colnames(ci) <- c('LMG', 'l.b', 'u.b')
rownames(ci) <- c('X1', 'X2', 'X3', 'X4')

#Code assuming stochastic predictors
run.stochastic<-boot.relimp(fit, fixed=TRUE)

ci.s <- booteval.relimp(run.stochastic, bty = "perc", level = 0.95,
                sort = FALSE, norank = TRUE, nodiff = TRUE,
                typesel = c("lmg"))@mark

ci.s <- as.matrix(ci.s)
ci.s <- matrix(as.numeric(ci.s), 4 , 3)
colnames(ci.s) <- c('LMG', 'l.b', 'u.b')
rownames(ci.s) <- c('X1', 'X2', 'X3', 'X4')
res.table <- tableRegression(fit, caption='Regression of paragraph comprehension on verbal test
#posterior sample
post.sample <- as.matrix(bayes.hs)

dt <- data.frame(post.sample[1:10,2:6])
rownames(dt)<-c('sample 1', 'sample 2', 'sample 3', 'sample 4', 'sample 5', 'sample 6', 'sa
#example of the first 10 posterior samples
kable(dt, format = "latex", booktabs=TRUE, linesep = c("", "", "", ""), caption = 'Samples
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))

#no need for the intercept, last parameter is sigma
post.sample <- post.sample[,-1]
#data frame with all submodels
df.rtwos <-rtwos(hs.data[,2:5], post.sample)

colnames(df.rtwos)[1:6] <- c('sample 1', 'sample 2', 'sample 3', 'sample 4', 'sample 5', 's
kable(df.rtwos[,1:6], format = 'latex', booktabs=TRUE, linesep = c("", "", "", ""), caption
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))

LMG.Vals.I<-matrix(0, 4, dim(df.rtwos)[2])

LMG.Vals.J<-matrix(0, 4, dim(df.rtwos)[2])

LMG.Vals.T<-matrix(0, 4, dim(df.rtwos)[2])


for(i in 1:dim(df.rtwos)[2]){

  gofn<-df.rtwos[,i]

  obj.Gelman<-partition(gofn, pcan = 4, var.names = names(hs.data[,2:5]))

  LMG.Vals.I[,i]=obj.Gelman$IJ[,1]
  LMG.Vals.J[,i]=obj.Gelman$IJ[,2]
        LMG.Vals.T[,i]=obj.Gelman$IJ[,3]
}

varnames <- row.names(obj.Gelman$IJ)


# posterior LMG distribution of each variable
quantile(LMG.Vals.I[1,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.I[2,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.I[3,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.I[4,], c(0.025, 0.5, 0.975))

quantile(LMG.Vals.J[1,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.J[2,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.J[3,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.J[4,], c(0.025, 0.5, 0.975))

quantile(LMG.Vals.T[1,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.T[2,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.T[3,], c(0.025, 0.5, 0.975))
quantile(LMG.Vals.T[4,], c(0.025, 0.5, 0.975))




#Visualization
dat <- data.frame(t(LMG.Vals.I))

pairs.chart <- ggpairs(dat, lower = list(list(combo = "density")), upper = list(list(continuous
pairs.chart




#---------------------------------------------------------------------------------------


df.rtwos.boot <-rtwos.boot(hs.data[,2:5], post.sample, 10)

n.boot = 10

LMG.Vals.I.boot<-array(0, c(4,dim(df.rtwos.boot)[2], n.boot))
LMG.Vals.J.boot<-array(0, c(4,dim(df.rtwos.boot)[2], n.boot))
LMG.Vals.T.boot<-array(0, c(4,dim(df.rtwos.boot)[2], n.boot))

for (b in 1:n.boot){

for(i in 1:dim(df.rtwos.boot)[2]){

        gofn<-df.rtwos.boot[,i,b]

        obj.Gelman<-partition(gofn, pcan = 4, var.names = names(df[,2:5]))

        LMG.Vals.I.boot[,i, b]=obj.Gelman$IJ[,1]
        LMG.Vals.J.boot[,i, b]=obj.Gelman$IJ[,2]
        LMG.Vals.T.boot[,i, b]=obj.Gelman$IJ[,3]

}

}

quantile(c(LMG.Vals.I.boot[1,1:1000,]), c(0.025, 0.5, 0.975))
quantile(c(LMG.Vals.I.boot[2,1:1000,]), c(0.025, 0.5, 0.975))
quantile(c(LMG.Vals.I.boot[3,1:1000,]), c(0.025, 0.5, 0.975))
quantile(c(LMG.Vals.I.boot[4,1:1000,]), c(0.025, 0.5, 0.975))

#very similar values as in the confidence intervals for stochastic predictors
t1 <- c(LMG.Vals.I.boot[1,,])
t2 <- c(LMG.Vals.I.boot[2,,])
```

```
t3 <- c(LMG.Vals.I.boot[3,,])
t4 <- c(LMG.Vals.I.boot[4,,])

dat.b<-data.frame(cbind(t1,t2,t3,t4))
cor(dat.b)


set_parent('Main.Rnw')
library(knitr); library(ggplot2); library(rjags);  library(MASS); library(hier.part)

opts_chunk$set(
    fig.path='figure/ch04_fig',
    self.contained=FALSE,
    cache=FALSE
)

options(digits =3)


#function to calculate R2 data frame from random intercept model
# rtwos for repeated measurement
# how much variance is explained by the within subject predictors (repeated measuremnt corr

# var(fixed) / (var(fixed)+residual)
rtwos.ri.r<-function(X, post.sample){
        # X: Predictor data as data frame
        # post.sample: posterior sample as matrix (M[sample_i,]), second last position shou

        X <- cov(X) #covariance matrix

        #Prepare data frame and rownames by using the combn() function

        lst <- list()

        pcan <- dim(X)[2]
        n <- (2^pcan)-1

        for (i in 1:pcan){
                lst[[i]] <- combn(pcan,i)
        }

        var.names <- character(length = 0)

  v<- rownames(X)

        for(i in 1:length(lst))
        {

                for (j in 1:ncol(lst[[i]])){
                        cur<- lst[[i]][,j]
                        name <- paste0(v[-cur])
                        name <- paste(name, collapse = " ")
                        var.names <- c(var.names, name)
                }
        }

        var.names<-c(rev(var.names), 'all')

        var.names[1]<-'none'

        size <- nrow(post.sample)  # how many samples

        sig.posi <- ncol(post.sample)-1 # second last position sigma

        df.Rtwos<-data.frame(matrix(0,n+1,size))

        row.names(df.Rtwos) <- var.names



        ########### fill in R^2 matrix, use posterior samples and calculate submodels according

        v <- 1:dim(X)[2]

        for (s in 1:size){

                sample.s <- post.sample[s,1:ncol(X)]

                Vtot <- sample.s%*%X%*%sample.s  #total explained variance

                count=n

                for(i in 1:(length(lst)-1))
                {

                        for (j in 1:ncol(lst[[i]])){
                                cur <- lst[[i]][,j]
                                set <- v[-cur]
                                matr <- X[cur, cur] - X[cur, set]%*%solve(X[set, set])%*%X[set,
                                var.explain<- sample.s[cur]%*%matr%*%sample.s[cur]  #multiply t
                                df.Rtwos[count,s] <- Vtot - var.explain
                                count=count-1
                        }
                }

                df.Rtwos[n+1,s] <- Vtot

                df.Rtwos[,s] <- df.Rtwos[,s]/c(Vtot+post.sample[s,sig.posi]^2) #total variance
        }

        return(df.Rtwos)
}
```

```
rtwos.ri.a<-function(X, post.sample){
        # X: Predictor data as data frame
        # post.sample: posterior sample as matrix (M[sample_i,]), second last position shou

        X <- cov(X) #covariance matrix

        #Prepare data frame and rownames by using the combn() function

        lst <- list()

        pcan <- dim(X)[2]
        n <- (2^pcan)-1

        for (i in 1:pcan){
                lst[[i]] <- combn(pcan,i)
        }

        var.names <- character(length = 0)

        v<- rownames(X)


        for(i in 1:length(lst))
        {

                for (j in 1:ncol(lst[[i]])){
                        cur<- lst[[i]][,j]
                        name <- paste0(v[-cur])
                        name <- paste(name, collapse = " ")
                        var.names <- c(var.names, name)
                }
        }

        var.names<-c(rev(var.names), 'all')

        var.names[1]<-'none'

        size <- nrow(post.sample)  # how many samples

        sig.posi <- (ncol(post.sample)-1) # second last position sigma, last position rando

        subj.posi <- ncol(post.sample)
        df.Rtwos<-data.frame(matrix(0,n+1,size))

        row.names(df.Rtwos) <- var.names

        ########### fill in R^2 matrix, use posterior samples and calculate submodels accor
```

```
          v <- 1:dim(X)[2]

        for (s in 1:size){

                sample.s <- post.sample[s,1:ncol(X)]

                Vtot <- sample.s%*%X%*%sample.s  #total explained variance

                count=n

                for(i in 1:(length(lst)-1))
                {

                        for (j in 1:ncol(lst[[i]])){
                                cur <- lst[[i]][,j]
                                set <- v[-cur]
                                matr <- X[cur, cur] - X[cur, set]%*%solve(X[set, set])%*%X[set,
                                var.explain<- sample.s[cur]%*%matr%*%sample.s[cur]  #multiply b
                                df.Rtwos[count,s] <- Vtot - var.explain
                                count=count-1
                        }
                }

                df.Rtwos[n+1,s] <- Vtot

                df.Rtwos[,s] <- df.Rtwos[,s]/c(Vtot+post.sample[s,sig.posi]^2 +post.sample[s,su
        }

        return(df.Rtwos)
}



rtwos.marg<-function(X, post.sample, m){

        # X: Predictor data as data frame
        # post.sample: posterior sample as matrix (M[sample_i,]), the last positions are filled

        X <- cov(X) #covariance matrix

        #Prepare data frame and rownames by using the combn() function

        lst <- list()

        pcan <- dim(X)[2]
        n <- (2^pcan)-1

        for (i in 1:pcan){
                lst[[i]] <- combn(pcan,i)
        }
```

```
var.names <- character(length = 0)

v<- rownames(X)

for(i in 1:length(lst))
{

        for (j in 1:ncol(lst[[i]])){
                cur<- lst[[i]][,j]
                name <- paste0(v[-cur])
                name <- paste(name, collapse = " ")
                var.names <- c(var.names, name)
        }
}

var.names<-c(rev(var.names), 'all')

var.names[1]<-'none'

size <- nrow(post.sample)  # how many samples

sig.posi <- ((ncol(post.sample)-(m-1)):ncol(post.sample)) # second last position si

df.Rtwos<-data.frame(matrix(0,n+1,size))

row.names(df.Rtwos) <- var.names

########### fill in R^2 matrix, use posterior samples and calculate submodels accor
v <- 1:dim(X)[2]


for (s in 1:size){

        sample.s <- post.sample[s,1:ncol(X)]

        Vtot <- sample.s%*%X%*%sample.s  #total explained variance

        count=n

        for(i in 1:(length(lst)-1))
        {

                for (j in 1:ncol(lst[[i]])){
                        cur <- lst[[i]][,j]
                        set <- v[-cur]
                        matr <- X[cur, cur] - X[cur, set]%*%solve(X[set, set])%*%X[
                        var.explain<- sample.s[cur]%*%matr%*%sample.s[cur]  #multip
                        df.Rtwos[count,s] <- Vtot - var.explain
                        count=count-1
                }
        }
```

```
                df.Rtwos[n+1,s] <- Vtot

                df.Rtwos[,s] <- df.Rtwos[,s]/c(Vtot + sum(post.sample[s,sig.posi] )) #total var
        }

        return(df.Rtwos)
}





sub<- 1:20
subi <- rnorm(20, 0, 4)
subi<-rep(subi, 4)
t <- c(0, 1, 2,3)
t <- c(rep(0, 20), rep(1,20), rep(2, 20), rep(3,20))

mu <- rep(0,4)
sig <- matrix(0.4, 4, 4)
diag(sig) <- 1
sig[3,4] <- 0.9
sig[4,3] <- 0.9
sig[1,2] <- 0.3
sig[2,1] <- 0.3


rawvars <- mvrnorm(n=80, mu=mu, Sigma=sig)

x1 <- t+rawvars[,1]
x2 <- t+rawvars[,2]
x3 <- t+rawvars[,3]
x4 <- t+rawvars[,4]

b1 <- b2  <-1
b3 <- b4 <- 2

y<- x1*b1 +x2*b2 +x3*b3+  x4*b4 + subi+ rnorm(80, 0, 0.1)

df <- data.frame(y=y, x1 = x1, x2=x2, x3 = x3, x4 = x4, sub = rep(sub,4))


p <- ggplot(data = df, aes(x = t, y = y, group = sub))
p + geom_line()


fit <- stan_glmer(y ~ x1+x2+x3+x4 + (1|sub) ,
                                                                        data = df,
                                                                        chains = 4, cores = 4)
```

```
post.sample <- as.matrix(fit)
post.sample.r <- post.sample[,c(2:5,(ncol(post.sample)-1):ncol(post.sample))]

df.rtwos <- rtwos.ri.r(df[,2:5], post.sample.r)

LMG.Vals.I<-matrix(0, 4, dim(df.rtwos)[2])

LMG.Vals.J<-matrix(0, 4, dim(df.rtwos)[2])

LMG.Vals.T<-matrix(0, 4, dim(df.rtwos)[2])


for(i in 1:dim(df.rtwos)[2]){

  gofn<-df.rtwos[,i]

  obj.Gelman<-partition(gofn, pcan = 4, var.names = names(df[,2:5]))

  LMG.Vals.I[,i]=obj.Gelman$IJ[,1]
  LMG.Vals.J[,i]=obj.Gelman$IJ[,2]
        LMG.Vals.T[,i]=obj.Gelman$IJ[,3]
}

varnames <- row.names(obj.Gelman$IJ)



# explained compared to total variance

df.rtwos <- rtwos.ri.a(df[,2:5], post.sample)


LMG.Vals.I<-matrix(0, 4, dim(df.rtwos)[2])

LMG.Vals.J<-matrix(0, 4, dim(df.rtwos)[2])

LMG.Vals.T<-matrix(0, 4, dim(df.rtwos)[2])


for(i in 1:dim(df.rtwos)[2]){

  gofn<-df.rtwos[,i]

  obj.Gelman<-partition(gofn, pcan = 4, var.names = names(df[,2:5]))

  LMG.Vals.I[,i]=obj.Gelman$IJ[,1]
  LMG.Vals.J[,i]=obj.Gelman$IJ[,2]
        LMG.Vals.T[,i]=obj.Gelman$IJ[,3]
}

sub<- 1:20
```

```
subi <- rnorm(20, 0, 1)
subi<-rep(subi, 4)

mu <- rep(0,4)
sig <- matrix(0.4, 4, 4)
diag(sig) <- 1
sig[3,4] <- 0.9
sig[4,3] <- 0.9
sig[1,2] <- 0.3
sig[2,1] <- 0.3

rawvars <- mvrnorm(n=80, mu=mu, Sigma=sig)
cov(rawvars)
t <- c(rep(1, 20),rep(2,20), rep(3, 20), rep(4, 20))
x1 <- t+rawvars[,1]
x2 <- t+rawvars[,2]
x3 <- t+rawvars[,3]
x4 <- t+rawvars[,4]

Sig<- matrix(3, 4,4)
diag(Sig) <- 10
u <- rep(0, 4)
Sig[1,1] <- 5
Sig[2,2] <- 7
Sig[3,4] <- 8
Sig[4,3] <- 8

Sig[1,2] <- 4
Sig[2,1] <-4


error <- mvrnorm(20, u, Sig)

y<- x1*b1 +x2*b2 + x3*b3 +x4*b4 +c(error)


t <- c(rep(1, 20), rep(2, 20), rep(3, 20), rep(4, 20))
df <- data.frame(y=y, x1 = x1, x2=x2 , x3 = x3, x4 = x4, sub = rep(sub,4), t =t)







# Bayesian framework

Y <- matrix(df[,'y'], 20, 4, byrow=F)
x1 <- matrix(df[,'x1'], 20, 4, byrow=F)
x2 <- matrix(df[,'x2'], 20, 4, byrow=F)
x3 <- matrix(df[,'x3'], 20, 4, byrow=F)
x4 <- matrix(df[,'x4'], 20, 4, byrow=F)
```

```
N = 20 #subjects
M = 4 # repeated measures
p <- ggplot(data = df, aes(x = t, y = y, group = sub))
p + geom_line()



#--------------------------------------------

modelString <- "model{

# Likelihood
for(i in 1:N){
Y[i,1:M] ~ dmnorm(mu[i,1:M],Omega[1:M,1:M])
for(j in 1:M){
mu[i,j] <- beta0 + beta1*x1[i,j]+ beta2*x2[i,j]+ beta3*x3[i,j] + beta4*x4[i,j]
}}

# Priors

Omega[1:M, 1:M] ~dwish(zRmat[1:M,1:M] , zRscal)
Sigma[1:M, 1:M] <- inverse(Omega)

beta0      ~ dnorm(0,0.001)
beta1      ~ dnorm(0,0.001)
beta2      ~ dnorm(0,0.001)
beta3      ~ dnorm(0,0.001)
beta4      ~ dnorm(0,0.001)

}"


writeLines( modelString , con="Jags-MultivariateNormal-model.txt" )

model <- jags.model(
        textConnection(modelString), data = list(Y=Y,N=N,M=M,x1 = x1, x2 = x2, x3 = x3, x4

samp <- coda.samples(model, variable.names=c("beta1",'beta2', 'beta3', 'beta4' ,"Sigma"), n

summary(samp)

#LMG calculations

samp <- coda.samples(model, variable.names=c("beta1",'beta2', 'beta3', 'beta4' ,"Sigma[1,1]
                                                                                 n.iter=200

post.sample <- samp[[1]][,c(5:8, 1:4)]

df.rtwos <- rtwos.marg(df[,2:5], post.sample, 4) # 4 repeated measures
```

```
LMG.Vals.I<-matrix(0, 4, dim(df.rtwos)[2])

LMG.Vals.J<-matrix(0, 4, dim(df.rtwos)[2])

LMG.Vals.T<-matrix(0, 4, dim(df.rtwos)[2])


for(i in 1:dim(df.rtwos)[2]){

  gofn<-df.rtwos[,i]

  obj.Gelman<-partition(gofn, pcan = 4, var.names = names(df[,2:5]))

  LMG.Vals.I[,i]=obj.Gelman$IJ[,1]
  LMG.Vals.J[,i]=obj.Gelman$IJ[,2]
        LMG.Vals.T[,i]=obj.Gelman$IJ[,3]
}



set_parent('Main.Rnw')
library(knitr)
opts_chunk$set(
    fig.path='figure/ch05_fig',
    self.contained=FALSE,
    cache=TRUE
)
set_parent('Main.Rnw')
library(knitr)
opts_chunk$set(
    fig.path='figure/ch06_fig',
    self.contained=FALSE,
    cache=TRUE
)
```

Bibliography

Alexander, D. L., Tropsha, A., and Winkler, D. A. (2015). Beware of R2: Simple, Unambiguous Assessment of the Prediction Accuracy of QSAR and QSPR Models. *Journal of Chemical Information and Modeling*, **55**, 1316–1322. 4

Bland, J. M. and Altman, D. G. (1995). Calculating correlation coefficients with repeated observations: Part 1–Correlation within subjects. *BMJ (Clinical research ed.)*, **310**, 446. 20

Chevan, A. and Sutherland, M. (1991). Hierarchical partitioning. *American Statistician*, **45**, 90–96. 4

Gelman, A., Goodrich, B., Gabry, J., and Ali, I. (2017). R-squared for Bayesian regression models *. Technical report. 5, 27

Grömping, U. (2006). Relative Importance for Linear Regression in R : The Package relaimpo. *Journal of Statistical Software*, **17**, 1–27. 4, 8

Grömping, U. (2007). Estimators of relative importance in linear regression based on variance decomposition. *American Statistician*, **61**, 139–147. 1, 6

Grömping, U. (2015). Variable importance in regression models. *Wiley Interdisciplinary Reviews: Computational Statistics*, **7**, 137–152. 1, 3, 4

Holzinger, K. J. and Swineford, F. (1939). A study in factor analysis: The stability of a bi-factor solution. *Supplementary Educational Monographs*, **48**, 1–91. 14

Jackman, S. (2009). *Bayesian Analysis for the Social Sciences*. Wiley. 7

Kelley, K. (2017). Mbess (version 4.0.0 and higher) [computer software and manual]. R package version 4.0.0 and higher. 14

Kvalseth, T. O. (1985). Cautionary Note about R 2. *The American Statistician*, **39**, 279. 4

Link, W. A. and Eaton, M. J. (2012). On thinning of chains in MCMC. *Methods in Ecology and Evolution*, **3**, 112–115. 9

Nakagawa, S. and Schielzeth, H. (2013). A general and simple method for obtaining R2 from generalized linear mixed-effects models. *Methods in Ecology and Evolution*, **4**, 133–142. 19, 20

Nimon, K., Lewis, M., Kane, R., and Haynes, R. M. (2008). An R package to compute commonality coefficients in the multiple regression case: An introduction to the package and a practical example. *Behavior Research Methods*, **40**, 457–466. 14

Nimon, K. F. and Oswald, F. L. (2013). Understanding the Results of Multiple Linear Regression: Beyond Standardized Regression Coefficients. *Organizational Research Methods*, **00**, 1–25. 1

Schäfer, J. and Strimmer, K. (2005). A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, **4**, 1–30. 8

Stan Development Team (2016). rstanarm: Bayesian applied regression modeling via Stan. R package version 2.13.1. 9

Stan Development Team (2017). *Stan Modeling Language: User's Guide and Reference Manual.* 9

Walsh, C. and Nally, R. M. (2015). Title Hierarchical Partitioning. Technical report. 4