Location-Scale Regression and the *lmls* Package

Hannes Riebl

Abstract

The linear model for location and scale (LMLS) is a multi-predictor regression model with explanatory variables for the mean (= the location) and the standard deviation (= the scale) of a normally distributed response variable. It is a special case of the generalized additive model for location, scale and shape (GAMLSS), as described by Rigby and Stasinopoulos (2005). This vignette discusses the *lmls* package for R, motivating the model class with a real-world application and illustrating the capabilities of the package: maximum likelihood and Markov chain Monte Carlo (MCMC) inference, a parametric bootstrap algorithm, and diagnostic plots for the LMLS model class.

The *lmls* package and this vignette were written for the "Advanced Statistical Programming" course at the University of Göttingen and published on CRAN to provide an accessible introduction to anybody who is curious about location-scale regression, and as a basis for the implementation of additional inference algorithms and model extensions.

Contents

1 Motivation and model

The *lmls* package comes with the *abdom* dataset (which it borrows from the *gamlss.data* package). The dataset consists of only two variables: the size of 610 fetuses (as measurements of their abdominal circumference taken from ultrasound scans) and their gestational age ranging from 12 to 42 weeks.

```r
library(lmls)

ggplot(abdom, aes(x, y)) +
  geom_point(color = "darkgray", size = 1) +
  xlab("Age [weeks]") +
  ylab("Size [mm]")
```

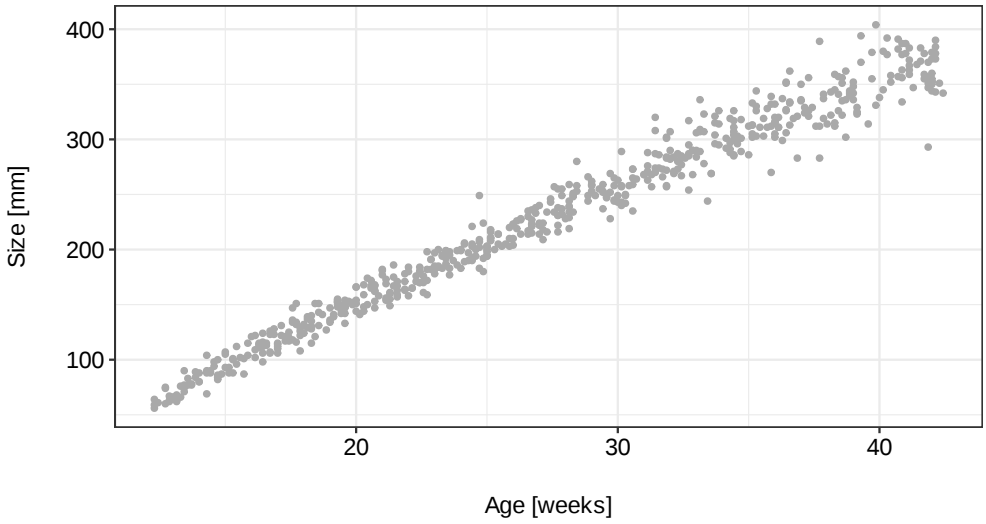

Figure 1: The *abdom* dataset containing the gestational age and abdominal circumference of 610 fetuses.

As expected, Figure 1 indicates that the size of the babies increases with their age. We can use a simple linear regression model to quantify this relationship:

```r
m1 <- lm(y ~ x, data = abdom)
summary(m1)
#>
#> Call:
#> lm(formula = y ~ x, data = abdom)
#>
#> Residuals:
#>     Min      1Q  Median      3Q     Max
#> -84.579  -8.105  -0.185   8.064  54.325
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept) -55.1795     2.0010  -27.58   <2e-16 ***
#> x            10.3382     0.0701  147.49   <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 14.63 on 608 degrees of freedom
#> Multiple R-squared:  0.9728, Adjusted R-squared:  0.9728
#> F-statistic: 2.175e+04 on 1 and 608 DF,  p-value: < 2.2e-16
```

The simple linear regression model with normally distributed errors is defined as

$$y_i = \beta_0 + x_i\beta_1 + \varepsilon_i, \text{ where } \varepsilon_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2).$$

Based on the scatterplot of the data in Figure 1, the homoscedasticity (= constant variance) assumption of the simple linear regression model seems implausible. In fact, the variance of the babies' size seems to increase with their age. We can confirm this by plotting the regression residuals against the babies' age:

```
abdom$resid <- resid(m1)

ggplot(abdom, aes(x, resid)) +
  geom_point(color = "darkgray", size = 1) +
  geom_hline(yintercept = 0, size = 0.5) +
  xlab("Age [weeks]") +
  ylab("Residuals")
```

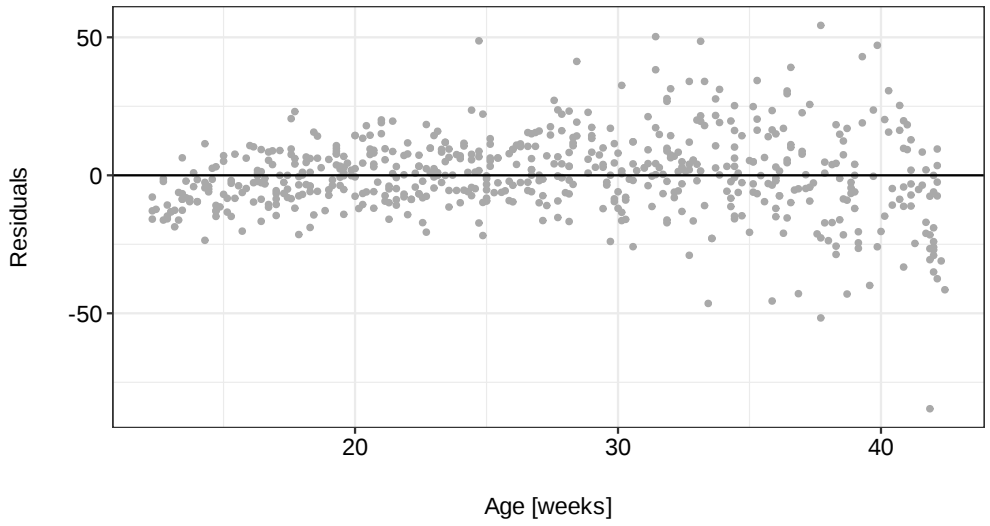

Figure 2: The residuals of the simple linear regression model for the *abdom* dataset show clear heteroscedasticity and some non-linearity.

It follows from the definition of the simple linear regression model that the response variable y_i is normally distributed with mean $\mu_i = \beta_0 + x_i\beta_1$ and standard deviation σ, yielding

$$y_i \overset{\text{ind.}}{\sim} \mathcal{N}(\beta_0 + x_i\beta_1, \sigma^2).$$

From this representation, we can extend the simple linear regression model and use the explanatory variable x_i to predict not only the mean μ_i but also the standard deviation σ_i of the response variable y_i. We translate this idea into the model

$$y_i \overset{\text{ind.}}{\sim} \mathcal{N}(\beta_0 + x_i\beta_1, (\exp(\gamma_0 + x_i\gamma_1))^2), \tag{1}$$

which is a minimal linear model for location and scale (LMLS). The regression coefficients γ_0 and γ_1 are the intercept and the slope parameter for the log-standard deviation, and the exponential function is the inverse link (or response) function. It ensures that the predictions for the standard deviation are always valid (= positive). This step is necessary, because the linear predictor $\gamma_0 + x_i\gamma_1$ can become zero or negative for some choices of γ_0 and γ_1.

Using the *lmls* package, we can estimate Model (1) for the *abdom* dataset with a maximum likelihood algorithm running the following line of code:

```r
m2 <- lmls(y ~ x, ~ x, data = abdom)

abdom$mu <- predict(m2, type = "response", predictor = "location")
abdom$sigma <- predict(m2, type = "response", predictor = "scale")
abdom$upper <- abdom$mu + 1.96 * abdom$sigma
abdom$lower <- abdom$mu - 1.96 * abdom$sigma

ggplot(abdom, aes(x, y)) +
  geom_point(color = "darkgray", size = 1) +
  geom_line(aes(y = mu), size = 0.7) +
  geom_line(aes(y = upper), size = 0.3) +
  geom_line(aes(y = lower), size = 0.3) +
  xlab("Age [weeks]") +
  ylab("Size [mm]")
```

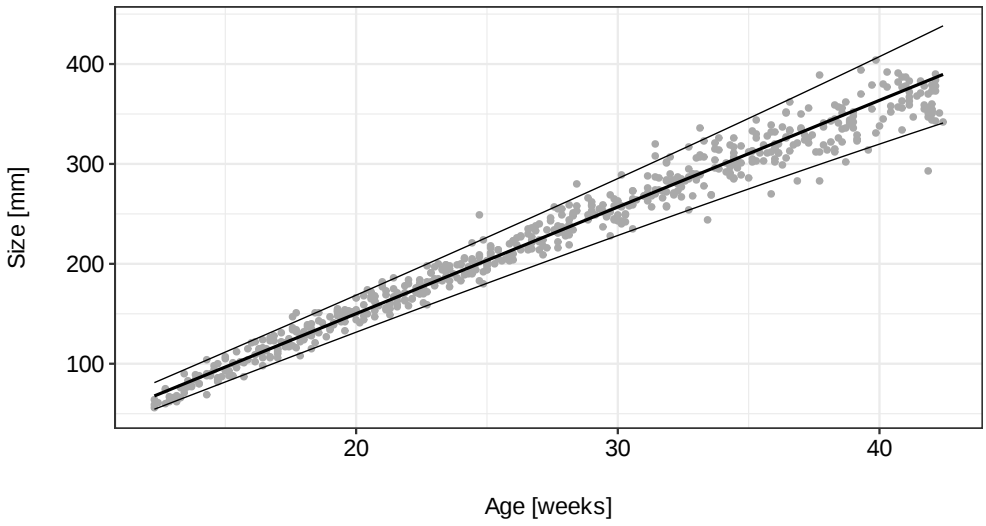

Figure 3: The LMLS for the *abdom* dataset with a linear effect of the babies' age on their average size and a linear effect on the log-standard deviation.

The general LMLS can include multiple explanatory variables, transformations of the explanatory variables, and it does *not* require the explanatory variables for the mean and the standard deviation to be identical. We define the general LMLS as

$$y_i \overset{\text{ind.}}{\sim} \mathcal{N}(\boldsymbol{x}_i'\boldsymbol{\beta}, (\exp(\boldsymbol{z}_i'\boldsymbol{\gamma}))^2),$$

where $\boldsymbol{x}_i$ and $\boldsymbol{\beta}$ are the covariate vector and the vector of regression coefficients for the mean, and $\boldsymbol{z}_i$ and $\boldsymbol{\gamma}$ are the covariate vector and the vector of regression coefficients for the standard deviation.

Polynomials are a straightforward way to include transformations of explanatory variables in a model. For example, we can improve Model (1) for the *abdom* dataset and add a quadratic effect of the babies' age on their average size with this command:

```r
m3 <- lmls(y ~ poly(x, 2), ~ x, data = abdom)
```

The AIC drops from 4861.184 to 4802.823 for this model compared to Model (1). Figure 4 illustrates how the quadratic effect improves the fit of the model.

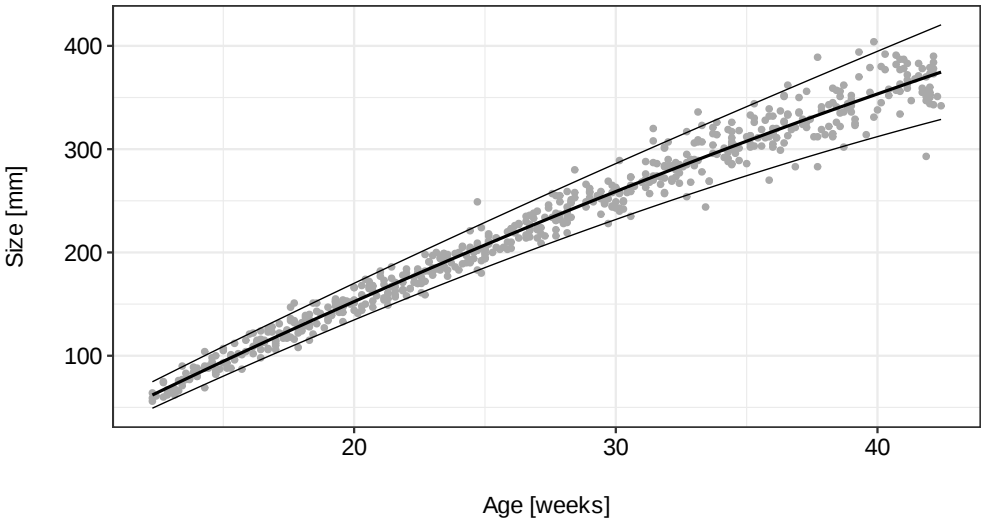

Figure 4: The LMLS for the *abdom* dataset with a quadratic effect of the babies' age on their average size and a linear effect on the log-standard deviation.

2 Statistical inference

The `lmls` package implements two inference algorithms: a frequentist maximum likelihood (ML) algorithm, which it uses by default, and a Bayesian Markov chain Monte Carlo (MCMC) algorithm. The derivatives that are required for these inference algorithms are derived in the Appendix in Section 4.

To simplify the notation in this and the next section, we first define the response vector $\boldsymbol{y} = [y_1, \ldots, y_n]'$, the design matrices $\mathbf{X} = [\boldsymbol{x}_1', \ldots, \boldsymbol{x}_n']'$ and $\mathbf{Z} = [\boldsymbol{z}_1', \ldots, \boldsymbol{z}_n']'$, and the standard deviation of the i-th observation $\sigma_i = \exp(\boldsymbol{z}_i'\boldsymbol{\gamma})$. Finally, let $\mathbf{W}$ be the weight matrix

$$\mathbf{W} = \begin{bmatrix} 1/\sigma_1^2 & 0 & \ldots & 0 \\ 0 & 1/\sigma_2^2 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \ldots & 1/\sigma_n^2 \end{bmatrix}. \tag{2}$$

2.1 Maximum likelihood

The ML algorithm combines weighted least squares (WLS) updates for $\hat{\boldsymbol{\beta}}$ and Fisher scoring updates for $\hat{\boldsymbol{\gamma}}$. We first discuss this update strategy and then take a look at the initialization strategy.

2.1.1 Update strategy

If we know the true $\boldsymbol{\gamma}$ and hence the true weight matrix $\mathbf{W}$, we can estimate $\boldsymbol{\beta}$ with WLS:

$$\hat{\boldsymbol{\beta}}^{(\mathrm{WLS})} = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}\boldsymbol{y}.$$

On the other hand, if we know the true $\boldsymbol{\beta}$, we can estimate $\boldsymbol{\gamma}$ with the Fisher scoring algorithm. Fisher scoring is an iterative algorithm for ML estimation, similar to Newton's method. The k-th update of the Fisher scoring algorithm is defined as

$$\hat{\boldsymbol{\gamma}}^{(k)} = \hat{\boldsymbol{\gamma}}^{(k-1)} + (\mathcal{I}(\hat{\boldsymbol{\gamma}}^{(k-1)}))^{-1} s(\hat{\boldsymbol{\gamma}}^{(k-1)}),$$

where $\mathcal{I}$ is the Fisher information and s is the score of $\boldsymbol{\gamma}$.

In most cases, we know neither the true $\boldsymbol{\beta}$ nor the true $\boldsymbol{\gamma}$, but we can estimate them with an iterative algorithm alternating between a Fisher scoring update for $\hat{\boldsymbol{\gamma}}$ and a WLS update for $\hat{\boldsymbol{\beta}}$. The other vector of regression coefficients is kept fixed at each step.

2.1.2 Initialization strategy

For the iterative algorithm to work well, we need to find good starting values. To initialize $\hat{\boldsymbol{\beta}}$, the ordinary least squares (OLS) estimator $\hat{\boldsymbol{\beta}}^{(\mathrm{OLS})} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{y}$ is a natural choice. Note that $\hat{\boldsymbol{\beta}}^{(\mathrm{OLS})}$ is unbiased for the LMLS, because

$$\mathbb{E}(\hat{\boldsymbol{\beta}}^{(\mathrm{OLS})}) = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbb{E}(\boldsymbol{y}) = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{X}\boldsymbol{\beta} = \boldsymbol{\beta}.$$

Under mild regularity conditions, $\hat{\boldsymbol{\beta}}^{(\mathrm{OLS})}$ is also consistent. However, it is not the best linear unbiased estimator (BLUE), because the homoscedasticity requirement of the Gauss-Markov theorem does not hold.

To initialize $\hat{\boldsymbol{\gamma}}$, we first estimate $\log \sigma_i$ with $\hat{s}_i = \log|y_i - \boldsymbol{x}_i'\hat{\boldsymbol{\beta}}^{(\mathrm{OLS})}| + 0.635$ and then regress $\hat{\boldsymbol{s}} = [\hat{s}_1, \ldots, \hat{s}_n]'$ on the design matrix $\mathbf{Z}$ with OLS to obtain $\hat{\boldsymbol{\gamma}}^{(0)} = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\hat{\boldsymbol{s}}$.

The motivation for $\hat{s}_i$ as an estimator for $\log \sigma_i$ is as follows: Consider

$$\log\left|\frac{y_i - \mu_i}{\sigma_i}\right| = \log|y_i - \mu_i| - \log \sigma_i \tag{3}$$

and

$$\log\left|\frac{y_i - \mu_i}{\sigma_i}\right| = \log\sqrt{\left(\frac{y_i - \mu_i}{\sigma_i}\right)^2} = \frac{1}{2} \cdot \log \underbrace{\left(\frac{y_i - \mu_i}{\sigma_i}\right)^2}_{\sim\chi^2(1)}. \tag{4}$$

Setting the RHS of Equation (3) equal to the RHS of Equation (4) and taking expectations yields

$$\mathbb{E}(\log|y_i - \mu_i|) - \log\sigma_i = \underbrace{1/2 \cdot (\psi(1/2) + \log 2)}_{\approx -0.635},$$

where ψ is the digamma function. The term $\psi(1/2) + \log 2$ follows from the fact that a $\chi^2(\nu)$ distribution is identical to a Gamma$(k = \nu/2, \theta = 2)$ distribution, and that for $X \sim \text{Gamma}(k, \theta)$, we have $\mathbb{E}(\log X) = \psi(k) + \log\theta$. Rearranging the equation to

$$\mathbb{E}\underbrace{(\log|y_i - \mu_i| + 0.635)}_{=s_i} = \log\sigma_i$$

shows that s_i is an unbiased estimator for $\log\sigma_i$. Unfortunately, we do not know the true μ_i in practice and have to use the unbiased estimator $\boldsymbol{x}_i'\hat{\boldsymbol{\beta}}^{(\text{OLS})}$ as an approximation instead.

2.1.3 Complete algorithm

1. Estimate $\hat{\boldsymbol{\beta}}^{(\text{OLS})} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{y}$.
2. Initialize $\hat{\boldsymbol{\gamma}}^{(0)} = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\hat{\boldsymbol{s}}$, where $\hat{\boldsymbol{s}} = [\hat{s}_i] = \log|y_i - \boldsymbol{x}_i'\hat{\boldsymbol{\beta}}^{(\text{OLS})}| + 0.635$.
3. Initialize $\hat{\boldsymbol{\beta}}^{(0)} = (\mathbf{X}'\hat{\mathbf{W}}^{(0)}\mathbf{X})^{-1}\mathbf{X}'\hat{\mathbf{W}}^{(0)}\boldsymbol{y}$, where $\hat{\mathbf{W}}^{(0)}$ is the weight matrix for $\hat{\boldsymbol{\gamma}}^{(0)}$.
4. Set $k = 1$.
5. Repeat the following steps until convergence:
    1. Update $\hat{\boldsymbol{\gamma}}^{(k)} = \hat{\boldsymbol{\gamma}}^{(k-1)} + (\mathcal{I}(\hat{\boldsymbol{\gamma}}^{(k-1)}))^{-1}s(\hat{\boldsymbol{\gamma}}^{(k-1)})$ keeping $\hat{\boldsymbol{\beta}}^{(k-1)}$ fixed.
    2. Update $\hat{\boldsymbol{\beta}}^{(k)} = (\mathbf{X}'\hat{\mathbf{W}}^{(k)}\mathbf{X})^{-1}\mathbf{X}'\hat{\mathbf{W}}^{(k)}\boldsymbol{y}$, where $\hat{\mathbf{W}}^{(k)}$ is the weight matrix for $\hat{\boldsymbol{\gamma}}^{(k)}$.
    3. Increase k by 1.

2.2 Markov chain Monte Carlo

To estimate an LMLS with MCMC, we can call the `mcmc()` function on an existing model object. The sampler requires a model object that contains the design matrices, so we need to make sure the `lmls()` function was called with the argument `light = FALSE`. Finally, we can use the `summary()` function with the argument `type = "mcmc"` to output some summary statistics of the posterior samples.

```
m3 <- lmls(y ~ poly(x, 2), ~ x, data = abdom, light = FALSE)
m3 <- mcmc(m3)

summary(m3, type = "mcmc")
#>
#> Call:
#> lmls(location = y ~ poly(x, 2), scale = ~x, data = abdom, light = FALSE)
#>
#> Deviance residuals:
#>      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
#> -3.188000 -0.700400 -0.063480 -0.004337  0.611500  4.060000
#>
#> Location coefficients (identity link):
#>                Mean    2.5%     50%   97.5%
#> (Intercept)  226.72  225.61  226.71  227.83
#> poly(x, 2)1 2160.34 2130.05 2160.87 2190.74
#> poly(x, 2)2  -99.46 -125.90  -99.24  -73.85
#>
#> Scale coefficients (log link):
```

```
#>                Mean    2.5%     50% 97.5%
#> (Intercept) 1.36740 1.16883 1.37487 1.569
#> x           0.04206 0.03513 0.04190 0.049
#>
#> Residual degrees of freedom: 605
#> Log-likelihood: -2399.48
#> AIC: 4808.95
#> BIC: 4831.02
```

The posterior samples for one regression coefficient, γ_1 in this case, can be extracted and plotted as follows:

```
samples <- as.data.frame(m3$mcmc$scale)
samples$iteration <- 1:nrow(samples)

p1 <- ggplot(samples, aes(iteration, x)) +
  geom_line() +
  xlab("Iteration") +
  ylab(expression(gamma[1]))

p2 <- ggplot(samples, aes(x, after_stat(density))) +
  geom_histogram(bins = 15) +
  xlab(expression(gamma[1])) +
  ylab("Density")

p1 + p2
```

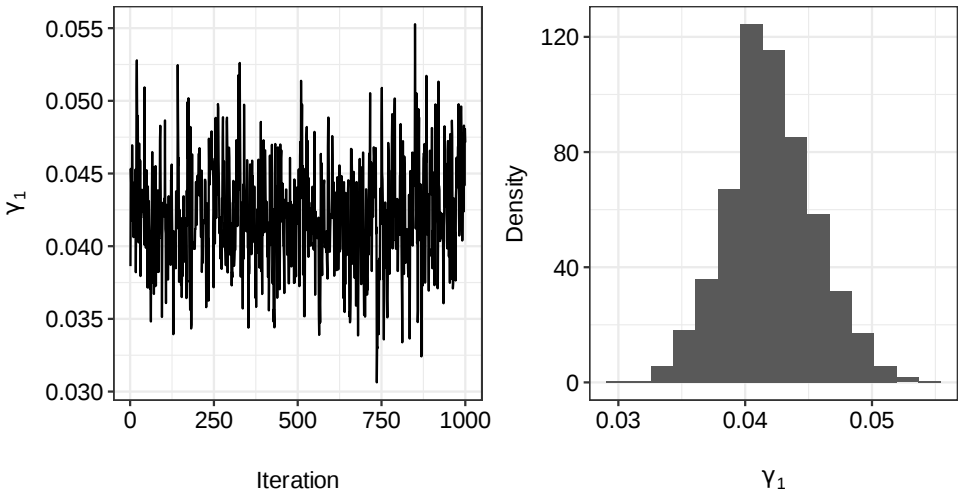

Figure 5: Trace plot and histogram of the posterior samples for γ_1.

2.2.1 MCMC algorithm

The MCMC algorithm assumes flat priors for $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ and works as follows:

1. Initialize $\boldsymbol{\beta}^{(0)}$ and $\boldsymbol{\gamma}^{(0)}$ with the ML estimates.

2. Set $k = 1$.
3. Repeat the following steps `num_samples` times:
    1. Sample $\boldsymbol{\beta}^{(k)}$ from the full conditional in a Gibbs update step (see Section 2.2.2).
    2. Sample $\boldsymbol{\gamma}^{(k)}$ with the Riemann manifold Metropolis-adjusted Langevin algorithm (MMALA) from Girolami and Calderhead (2011) using the Fisher-Rao metric tensor.
    3. Increase k by 1.

2.2.2 Full conditional of $\boldsymbol{\beta}$

The full conditional of $\boldsymbol{\beta}$ used in the Gibbs update step is given by

$$p(\boldsymbol{\beta} \mid \cdot) \propto \exp(-1/2 \cdot (\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta})'\mathbf{W}(\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta})) \cdot p(\boldsymbol{\beta}) \cdot p(\boldsymbol{\gamma}),$$

where the weight matrix $\mathbf{W}$ is defined as in Equation (2). The priors $p(\boldsymbol{\beta})$ and $p(\boldsymbol{\gamma})$ can be ignored, because we assume them to be flat. It can be shown with some tedious but simple linear algebra that

$$\begin{aligned} &(\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta})'\mathbf{W}(\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta}) \\ = &(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}^{(\mathrm{WLS})})'\mathbf{X}'\mathbf{W}\mathbf{X}(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}^{(\mathrm{WLS})}) + \boldsymbol{y}'(\mathbf{W} + \mathbf{W}\mathbf{X}(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W})\boldsymbol{y}, \end{aligned}$$

where $\hat{\boldsymbol{\beta}}^{(\mathrm{WLS})}$ is the WLS estimator for $\boldsymbol{\beta}$ using the weight matrix $\mathbf{W}$. As the second addend in the last equation is independent of $\boldsymbol{\beta}$, the full conditional reduces to

$$p(\boldsymbol{\beta} \mid \cdot) \propto \exp(-1/2 \cdot (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}^{(\mathrm{WLS})})'\mathbf{X}'\mathbf{W}\mathbf{X}(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}^{(\mathrm{WLS})})),$$

which implies the following multivariate normal distribution:

$$\boldsymbol{\beta} \mid \cdot \sim \mathcal{N}(\hat{\boldsymbol{\beta}}^{(\mathrm{WLS})}, (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}).$$

3 Comparison with other R packages

There are a number of R packages with similar capabilities as *lmls*. The popular *mgcv* package (Wood 2017), which is typically used to estimate generalized additive models (GAMs), added support for multi-predictor models including the LMLS with a normally distributed response variable in version 1.8. The *gamlss* package implements generalized additive models for location, scale and shape (GAMLSS, Rigby and Stasinopoulos 2005) in a very general way, and the *bamlss* package (Umlauf, Klein, and Zeileis 2018) is a Bayesian alternative to the *gamlss* package.

Compared to these packages, the scope of the *lmls* package is much narrower. Its functionality is limited to the LMLS with a normally distributed response variable. Other response distributions or the regularization of covariate effects are not supported. Instead, *lmls* aims to be...

- ... **user-friendly**. The few exported functions are intuitive and simple.
- ... **stable**. As *lmls* implements only one single, narrow model class, it can make use of specific and robust inference algorithms.
- ... **fast**. In fact, *lmls* seems to be 3.5 to 4 times faster than *mgcv* and *gamlss*. See below for a small benchmark on the *abdom* dataset.
- ... **lightweight**. *lmls* is written in pure R and depends only on the *generics* package.
- ... **comprehensible**. *lmls* was used as a teaching material for a university course. The code is modular and accessible, even to R beginners.
- ... **well-tested**. Most of the code is covered by unit and integration tests.

Finally, we compare the performance of the *lmls* package on the *abdom* dataset with *mgcv* and *gamlss* using the *microbenchmark* package. The results of the benchmark are shown in Figure 6.

```
library(gamlss)
library(mgcv)

bench <- microbenchmark::microbenchmark(
  lmls = lmls(y ~ poly(x, 2), ~ x, data = abdom),
  mgcv = gam(list(y ~ poly(x, 2), ~ x), family = gaulss(), data = abdom),
  gamlss = gamlss(y ~ poly(x, 2), ~ x, data = abdom)
)
```

4 Appendix: Score and Fisher information

The inference algorithms from Section 2 require the score (= the gradient of the log-likelihood) and the Fisher information (= the covariance of the score) with respect to $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$.

4.1 Location coefficients $\boldsymbol{\beta}$

The score of the location coefficients $\boldsymbol{\beta}$ is

$$s(\boldsymbol{\beta}) = \frac{\partial \ell}{\partial \boldsymbol{\beta}} = \mathbf{X}'\boldsymbol{q},$$

where $\boldsymbol{q}$ is an n-dimensional column vector with the elements $q_i = (y_i - \mu_i)/\sigma_i^2$. The corresponding Fisher information is given by

$$\mathcal{I}(\boldsymbol{\beta}) = \text{Cov}(s(\boldsymbol{\beta})) = \text{Cov}(\mathbf{X}'\boldsymbol{q}) = \mathbf{X}' \text{Cov}(\boldsymbol{q})\mathbf{X},$$

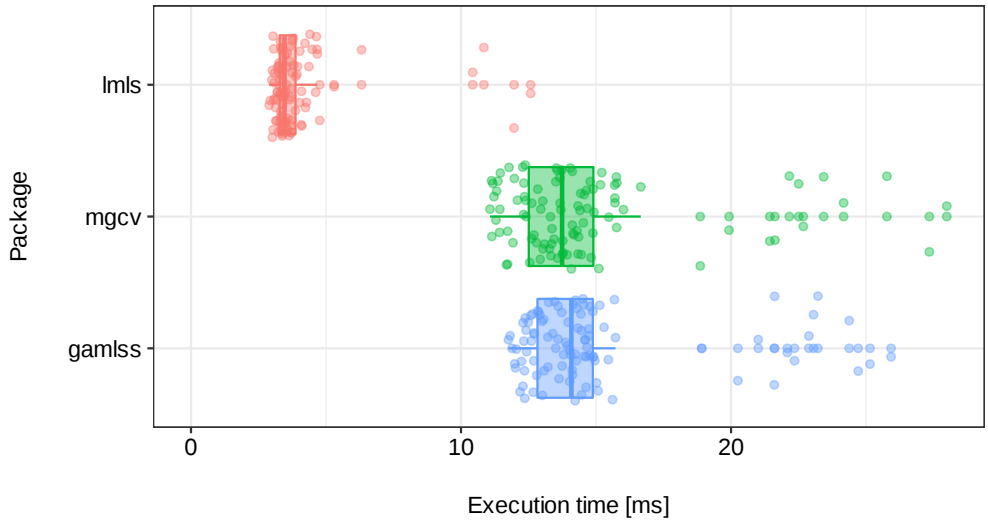

Figure 6: The *lmls* package is about 3.5 to 4 times faster than *mgcv* and *gamlss* on the *abdom* dataset.

where the diagonal elements of the covariance matrix $\mathrm{Cov}(\boldsymbol{q})$ are

$$\mathrm{Var}(q_i) = \mathrm{Var}\left(\frac{y_i - \mu_i}{\sigma_i^2}\right) = \frac{1}{\sigma_i^2} \cdot \mathrm{Var}\underbrace{\left(\frac{y_i - \mu_i}{\sigma_i}\right)}_{\sim\mathcal{N}(0,1)} = \frac{1}{\sigma_i^2},$$

and the off-diagonal elements are $\mathrm{Cov}(q_i, q_j) = 0$ for $i \neq j$, due to the independence assumption of the LMLS. In R, we can use the following efficient implementation of the Fisher information of $\boldsymbol{\beta}$:

```
crossprod(X / scale)
```

Here, `scale` is the vector $[\sigma_1, \ldots, \sigma_n]$. This code works, because R stores matrices in column-major order and recycles shorter vectors for operations like element-wise division.

4.2 Scale coefficients $\boldsymbol{\gamma}$

The score of the scale coefficients $\boldsymbol{\gamma}$ is

$$s(\boldsymbol{\gamma}) = \frac{\partial \ell}{\partial \boldsymbol{\gamma}} = \mathbf{Z}'\boldsymbol{r},$$

where $\boldsymbol{r}$ is an n-dimensional column vector with the elements $r_i = ((y_i - \mu_i)/\sigma_i)^2 - 1$. The corresponding Fisher information is given by

$$\mathcal{I}(\boldsymbol{\gamma}) = \mathrm{Cov}(s(\boldsymbol{\gamma})) = \mathrm{Cov}(\mathbf{Z}'\boldsymbol{r}) = \mathbf{Z}'\,\mathrm{Cov}(\boldsymbol{r})\mathbf{Z},$$

where the diagonal elements of the covariance matrix $\mathrm{Cov}(\boldsymbol{r})$ are

$$\mathrm{Var}(r_i) = \mathrm{Var}\left(\left(\frac{y_i - \mu_i}{\sigma_i}\right)^2 - 1\right) = \mathrm{Var}\underbrace{\left(\left(\frac{y_i - \mu_i}{\sigma_i}\right)^2\right)}_{\sim\chi^2(1)} = 2,$$

and the off-diagonal elements are $\mathrm{Cov}(r_i, r_j) = 0$ for $i \neq j$, due to the independence assumption of the LMLS. In R, we can use the following efficient implementation of the Fisher information of $\boldsymbol{\gamma}$:

```r
2 * crossprod(Z)
```

4.3 Mixed Fisher information of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$

The inference algorithms from Section 2 update the location coefficients $\boldsymbol{\beta}$ and the scale coefficients $\boldsymbol{\gamma}$ separately. Would a joint update maybe work better? Let us take a look at the Fisher information of the stacked regression coefficients

$$\mathcal{I}\left(\begin{bmatrix}\boldsymbol{\beta}\\ \boldsymbol{\gamma}\end{bmatrix}\right) = \begin{bmatrix}\mathcal{I}(\boldsymbol{\beta}) & \operatorname{Cov}(s(\boldsymbol{\beta}), s(\boldsymbol{\gamma}))\\ \operatorname{Cov}(s(\boldsymbol{\gamma}), s(\boldsymbol{\beta})) & \mathcal{I}(\boldsymbol{\gamma})\end{bmatrix}.$$

We are still missing the off-diagonal elements

$$\operatorname{Cov}(s(\boldsymbol{\beta}), s(\boldsymbol{\gamma})) = \operatorname{Cov}(s(\boldsymbol{\gamma}), s(\boldsymbol{\beta}))' = \operatorname{Cov}(\mathbf{X}'\boldsymbol{q}, \mathbf{Z}'\boldsymbol{r}) = \mathbf{X}' \operatorname{Cov}(\boldsymbol{q}, \boldsymbol{r})\mathbf{Z},$$

where the diagonal elements of the cross-covariance matrix $\operatorname{Cov}(\boldsymbol{q}, \boldsymbol{r})$ are

$$\begin{aligned}\operatorname{Cov}(q_i, r_i) &= \operatorname{Cov}\left(\frac{y_i - \mu_i}{\sigma_i^2}, \left(\frac{y_i - \mu_i}{\sigma_i}\right)^2 - 1\right)\\ &= \frac{1}{\sigma_i} \cdot \operatorname{Cov}\left(\frac{y_i - \mu_i}{\sigma_i}, \left(\frac{y_i - \mu_i}{\sigma_i}\right)^2\right)\\ &= 0.\end{aligned}$$

The third equality holds because $(y_i - \mu_i)/\sigma_i$ is a standard normal random variable and hence uncorrelated with its square. The off-diagonal elements of $\operatorname{Cov}(\boldsymbol{q}, \boldsymbol{r})$ are $\operatorname{Cov}(q_i, r_j) = 0$ for $i \neq j$, due to the independence assumption of the LMLS. It follows that $\operatorname{Cov}(s(\boldsymbol{\beta}), s(\boldsymbol{\gamma})) = \operatorname{Cov}(s(\boldsymbol{\gamma}), s(\boldsymbol{\beta})) = \mathbf{0}$, which means there is no additional information we can make use of for a joint update of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ (at least not in terms of the Fisher information).

References

Girolami, Mark, and Ben Calderhead. 2011. "Riemann Manifold Langevin and Hamiltonian Monte Carlo Methods." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 73 (2): 123–214. https://doi.org/10.1111/j.1467-9868.2010.00765.x.

Pedersen, Thomas Lin. 2020. *patchwork: The Composer of Plots*. https://CRAN.R-project.org/package=patchwork.

Rigby, R. A., and D. M. Stasinopoulos. 2005. "Generalized Additive Models for Location, Scale and Shape." *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 54 (3): 507–54. https://doi.org/10.1111/j.1467-9876.2005.00510.x.

Umlauf, Nikolaus, Nadja Klein, and Achim Zeileis. 2018. "BAMLSS: Bayesian Additive Models for Location, Scale, and Shape (and Beyond)." *Journal of Computational and Graphical Statistics* 27 (3): 612–27. https://doi.org/10.1080/10618600.2017.1407325.

Wickham, Hadley. 2016. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org.

Wood, S. N. 2017. *Generalized Additive Models: An Introduction with R*. 2nd ed. Chapman; Hall/CRC.