A compartative simulation study of the packages `mgcv` and `gamlss` in the case of location-scale modelling with zero-inflated Count Data

Alexander Piehler

Seminar on Mixed and Semiparametric Models, Institute for Statistics, LMU Munich

March 23, 2020

Contents

List of Figures

List of Tables

1 Introduction

In many applications of regression modelling, the explicit modelling of all parameters of a distribution through covariates is imperative. Such applications include for instance data with heterogeneous variance, quantile regression or the modeling of count data with zero-inflation, in which the amount of zero-inflation is dependent on one or more covariates. Rigby & Stasinopoulos (2005) propose a framework in which not only the mean of the dependent variable is modelled through covariates, but all the distribution parameters of the dependent variable are modeled through covariates. They call this framework Generalized Additve Models for Location, Scale and Shape (GAMLSS) and it builds, as the name implies, on the Generalized Addtive Model (GAM) framework as introduced by Hastie & Tibshirani (1990), in which covariates are incorporated into the linear predictor as smooth terms through basis expansion.

The incorporation of smooth functions enables the model to capture non-linear relationships arbitrarily well. To regulate under- and overfitting, the parameters associated with the basis expansion of the covariate are subject to a quadratic penalty, with a smoothing parameter controlling the appropriate degree of smoothing.

The estimation of smoothing parameters in GAM's and GAMLSS is an active area of research. Various methods have been proposed on how to estimate smoothing parameters in GAMLSS, differing in degree of penalization as well as computational reliability, stability and speed (see Rigby & Stasinopoulos 2014, Wood, Pya & Säfken 2016, Umlauf, Klein & Zeileis 2018). In this paper we compare two distinct methods, which are each implemented in the R-packages `gamlss` (Rigby & Stasinopoulos 2005) and `mgcv` (Wood et al. 2016) respectively and use different maximum likelihood estimation techniques to obtain estimates for the smoothing parameters. The comparison shall be carried out by simulating different data situations and assess in which one method outperforms the other. The simulation aims to replicate the results of Wood et al. (2016), in which the authors compared the performance of both packages when dealing with modelling the conditional mean of zero-inflated Poisson data with uncorrelated and correlated covariates. This simulation shall be extended in such a way, that not only the mean μ is modelled through the covariates but also the probability π of zero-inflation. Additionally, the data situation will be extended to include cases of low-mean zero-inflated Poisson data with correlated and uncorrelated covariates, as this case is known to lead to convergence issues when estimating smoothing parameters (Wood et al. 2016). Prior to presenting the methodology and results of the simulation study, a primer on Generalized Additive Models for Location, Scale and Shape shall be given, which is followed up by a brief overview of the existing methods for estimating smoothing parameters in GAMLSS via maximum likelihood. This paper concludes with closing remarks in which situations the use of the respective methods is appropriate.

2 Generalized Additive Models for Location, Scale and Shape

2.1 Motivation

In classical generalized linear and additive model frameworks, where the mean μ of the dependent variable y is modelled as a function of the explanatory variables, the variance of y, given by $\text{Var}(y) = \phi v(\mu)$, with ϕ being a fixed dispersion parameter and $v(\mu)$ a known variance function, is only modelled indirectly through the covariates by the means of μ. These assumptions, however, yield for many practical application unsatisfactory results.

This applies also for models that go beyond the exponential family such as mixture distributions. The zero-inflated Poisson distribution can be motivated as such a distribution, in such a way that the distribution is a mixture of responders $C_i = 1$ and non-responders $C_i = 0$, who always have a zero count (Tutz 2011). Formally it is assumed that $y \sim \text{ZIP}(\mu, \pi)$, where μ is the mean of the Poisson distribution and $(1 - \pi)$ is the probability of excess zeros or being a non-responder, such that

$$P(y_i = y) = P(y_i = y|C_i)\pi_i + P(y_i = y|C_i = 0)(1 - \pi_i).$$

Because one assumes a Poisson distribution in the subpopulations, $P(y_i = 0|C_i = 0) = 1$ and therefore the probability of a a zero count is

$$P(y_i = 0) = P(y_i = y|C_i)\pi_i + (1 - \pi_i) = \pi_i \exp(-\mu_i) + 1 - \pi_i$$

and the probability for counts greater than zero is

$$P(y_i = y) = P(y_i = y|C_i)\pi_i = \pi_i \frac{\mu_i^y \exp(-\mu_i)}{y!}.$$

In this formulation each observation y_i has its own rate μ_i and probability $(1 - \pi_i)$ of excess zeros. It is therefore likely that the regression model would be misspecified if π were to be treated solely as a nuisance parameter. This is particularly the case when the probability of excess zeros depends on the covariates. An example for zero-inflated count data can be found in Figure 1. Here, not only the mean μ is a function of x, but also the probability $(1 - \pi)$ of excess zeros, which increases as x increases.

In order to capture the dependency of the probability of zero-inflation appropriately, one would need to model the probability π explicitly through the covariate One way to explicitly model every parameter of a q-parametric distribution is using Generalized

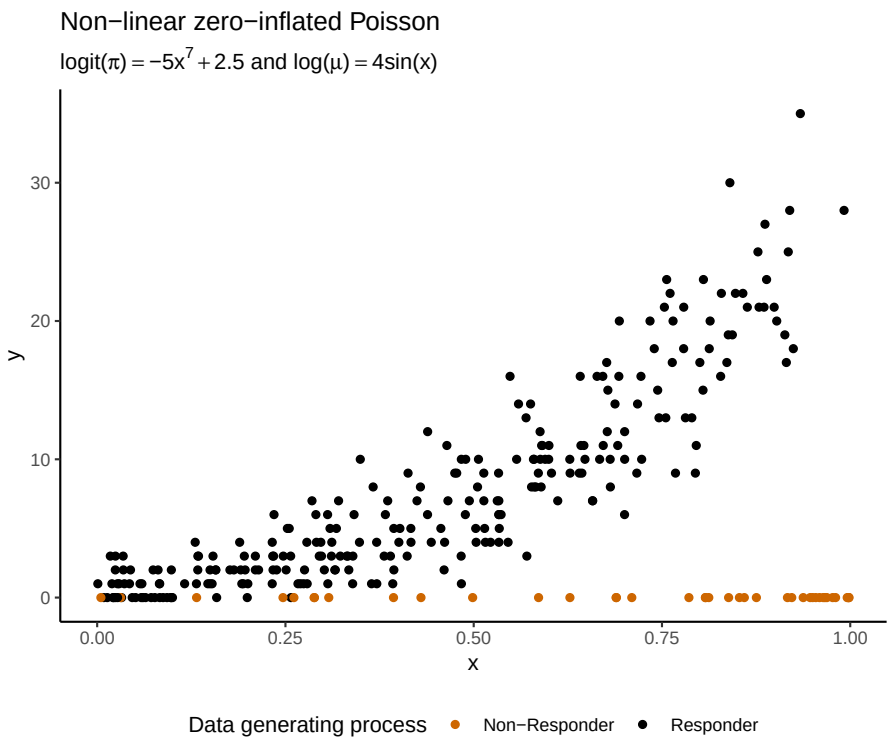

Figure 1: Zero-inflated data with the probability of excess zeros depending on a covariate

Additive Models for Location, Scale and Shape, which shall be defined in the next section.

2.2 Definition

Generalized Additive Models for Location, Scale and Shape (GAMLSS) take the form

$$
\begin{aligned}
\boldsymbol{Y} &\overset{ind}{\sim} \mathcal{D}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_q) \\
g_1(\boldsymbol{\theta}_1) = \boldsymbol{\eta}_1 &= \boldsymbol{X}_1\boldsymbol{\beta}_1 + \sum_{j=1}^{J_1} \boldsymbol{Z}_{j1}\boldsymbol{\gamma}_{j1} \\
&\vdots \\
g_k(\boldsymbol{\theta}_k) = \boldsymbol{\eta}_k &= \boldsymbol{X}_k\boldsymbol{\beta}_k + \sum_{j=1}^{J_k} \boldsymbol{Z}_{jk}\boldsymbol{\gamma}_{jk} \\
&\vdots \\
g_q(\boldsymbol{\theta}_q) = \boldsymbol{\eta}_q &= \boldsymbol{X}_q\boldsymbol{\beta}_q + \sum_{j=1}^{J_q} \boldsymbol{Z}_{jq}\boldsymbol{\gamma}_{jq},
\end{aligned}
$$

where $\mathcal{D}(\boldsymbol{\theta})$ is a q-parametric probability distribution with parameters $\boldsymbol{\theta}^\top = (\theta_1, \theta_2, \ldots, \theta_q)$ and $\boldsymbol{y}^\top = (y_1, \ldots, y_n)$ is the vector of response variable observations assumed to be independently drawn from the probability density function $f_{\mathcal{D}}(y|\boldsymbol{\theta})$. Furthermore, like in the

GLM and GAM case, for $k = 1, 2, \ldots, q$ there exists a monotonic link function $g_k(\cdot)$ and monotonic response function $g_k^{-1}(\cdot)$ such that the linear predictor η_k is mapped on the appropriate scale of the distribution parameter θ_k. Furthermore $\boldsymbol{X}_k$ is a known $n \times p_k$ design matrix with associated parameter vector $\boldsymbol{\beta}_k$ and $\boldsymbol{Z}_{jk}$ is a fixed $n \times p_{jk}$ basis matrix based on the covariate $\boldsymbol{x}_j$ with basis function $s_{jk}(\boldsymbol{x}_j)$ and associated parameter vector $\boldsymbol{\gamma}_{jk}$ of the dimension $p_{jk} \times 1$.
As it is with GAM's, each parameter vector $\boldsymbol{\gamma}_{jk}$ is subject to a quadratic penalty $\lambda_{jk}\boldsymbol{\gamma}_{jk}^\top \boldsymbol{S}_{jk}\boldsymbol{\gamma}_{jk}$ with λ_{jk} determining the degree of smoothing of the j-th basis function in the k-th linear predictor and $\boldsymbol{S}_{jk}$ being the respective known penalization matrix.
For many families of distributions four parameters are sufficient to model the distribution. These parameters are the location parameter μ, the scale parameter σ, and two shape parameters ν and τ. The later simulation shall be carried out with data generated from a zero-inflated Poisson distribution, which has μ as a location parameter and π as a scale parameter. Before that, however, a brief introduction of the inference in GAMLSS shall be given in next section. In particular the two ways of estimating the parameter vector $\boldsymbol{\lambda} = (\lambda_{11}, \ldots, \lambda_{J_1 1}, \ldots, \lambda_{1q}, \ldots, \lambda_{J_q q})$ in the packages `mgcv` and `gamlss`, when the same formulation of the penalized likelihood is used, shall be covered.

2.3 Inference in GAMLSS

While Bayesian methods exist to estimate the parameters of GAMLSS (see Umlauf et al. 2018), the two methods that shall be compared subsequently are both likelihood-based approaches. While both approaches of Wood (2017) and Rigby & Stasinopoulos (2014) seek to maximize the penalized likelihood, such that

$$\arg\max_{\boldsymbol{\beta},\boldsymbol{\gamma}} \ell_{pen}(\boldsymbol{\beta}, \boldsymbol{\gamma}) = \arg\max_{\boldsymbol{\beta},\boldsymbol{\gamma}} \left(\ell(\boldsymbol{\beta}, \boldsymbol{\gamma}) - \sum_{k=1}^{q} \sum_{j=1}^{J_k} \lambda_{jk} \boldsymbol{\gamma}_{jk}^\top \boldsymbol{S}_{jk} \boldsymbol{\gamma}_{jk} \right)$$

where

$$\ell(\boldsymbol{\beta}, \boldsymbol{\gamma}) = \sum_{i=1}^{n} \log f(y_i \mid \mu_i, \sigma_i, \nu_i, \tau_i),$$

the core differences in how the algorithms work is in how they estimate $\boldsymbol{\lambda}$. On a high level, the two methods can be distinguished as (1) local method (Stasinopoulos, Rigby, Heller, Voudouris & de Bastiani 2017) or the single-iteration case (Wood 2011) and global method (Stasinopoulos et al. 2017) or (2) the nested-iteration case (Wood 2011)[1]. Generally it is the case that in local method, in each penalized iteratively reweighted least

[1]Going forward the expressions "local method" and "nested iteration case" shall be used for cases (1) and (2) respectively

squares (PIRLS) step, which is used to update $\hat{\boldsymbol{\gamma}}_{jk}$ is also subsequently supplemented by λ_{jk} selection criterion $\mathcal{V}_{\hat{\boldsymbol{\gamma}}_{jk}}(\lambda_{jk})$, which depends as indicated on the prior estimate of $\hat{\boldsymbol{\gamma}}_{jk}$. The package `gamlss` then uses maximum likelihood (ML) to obtain the estimate $\hat{\boldsymbol{\lambda}}_{jk}$ (Rigby & Stasinopoulos 2014). In contrast in the nested iteration case, the smoothness selection criterion $\mathcal{V}(\boldsymbol{\lambda})$ depends on $\boldsymbol{\gamma}^\top = (\gamma_{11}^\top, \ldots, \gamma_{J_1 1}^\top, \ldots, \gamma_{1q}^\top, \ldots, \gamma_{J_q q}^\top)$ through $\hat{\boldsymbol{\gamma}}_{\boldsymbol{\lambda}}$. This means that an outer iteration updates $\hat{\boldsymbol{\lambda}}$ to optimize the global selection criterion $\mathcal{V}(\boldsymbol{\lambda})$ and an inner iteration finds the current for $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\gamma}}_\lambda$ through some optimization algorithm. Both methods shall be explained further in the following sections.

2.3.1 Estimation of $\boldsymbol{\lambda}$ in `gamlss`

While different methods for estimating $\boldsymbol{\lambda}$ have been implemented in the package `gamlss`, the one that - as mentioned beforehand - can computationally be compared to the implementation in the package `mgcv` is the local maximum likelihood (ML) estimation, as this implementation uses univariate P-Splines as smoothing functions.
The local ML method as proposed by Rigby & Stasinopoulos (2014) is applied within the so-called RS-algorithm, which in itself consists of an outer RS-algorithm and an inner RS-algorithm (Rigby & Stasinopoulos 2005). The outer RS-algorithm seeks to maximize the log-likelihood $\ell(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k, \ldots, \boldsymbol{\theta}_q)$ until convergence of twice the negative global deviance $-2\ell(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k, \ldots, \boldsymbol{\theta}_q)$. This is achieved by iteratively updating each estimate $\hat{\boldsymbol{\theta}}_k$ by means of the inner RS-Algorithm, while holding all other distribution parameters constant, which is shown in algorithm 1 for the case of a location, a scale and two scale parameters.

Algorithm 1: Outer RS Algorithm

Initialize $\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\sigma}}, \hat{\boldsymbol{\nu}}, \hat{\boldsymbol{\tau}} \leftarrow \boldsymbol{\mu}^{[0]}, \boldsymbol{\sigma}^{[0]}, \boldsymbol{\nu}^{[0]}, \boldsymbol{\tau}^{[0]}$;

$\hat{\boldsymbol{\gamma}} \leftarrow \boldsymbol{\gamma}_0$;

repeat

$\hat{\boldsymbol{\mu}}^{[i]} \leftarrow \arg\max_{\boldsymbol{\mu}} \ell(\hat{\boldsymbol{\mu}}^{[i-1]}, \hat{\boldsymbol{\sigma}}^{[i-1]}, \hat{\boldsymbol{\nu}}^{[i-1]}, \hat{\boldsymbol{\tau}}^{[i-1]})$ with inner RS algorithm;

$\hat{\boldsymbol{\sigma}}^{[i]} \leftarrow \arg\max_{\boldsymbol{\sigma}} \ell(\hat{\boldsymbol{\mu}}^{[i]}, \hat{\boldsymbol{\sigma}}^{[i-1]}, \hat{\boldsymbol{\nu}}^{[i-1]}, \hat{\boldsymbol{\tau}}^{[i-1]})$ with inner RS algorithm;

$\hat{\boldsymbol{\nu}}^{[i]} \leftarrow \arg\max_{\boldsymbol{\nu}} \ell(\hat{\boldsymbol{\mu}}^{[i]}, \hat{\boldsymbol{\sigma}}^{[i]}, \hat{\boldsymbol{\nu}}^{[i-1]}, \hat{\boldsymbol{\tau}}^{[i-1]})$ with inner RS algorithm;

$\hat{\boldsymbol{\tau}}^{[i]} \leftarrow \arg\max_{\boldsymbol{\tau}} \ell(\hat{\boldsymbol{\mu}}^{[i]}, \hat{\boldsymbol{\sigma}}^{[i]}, \hat{\boldsymbol{\nu}}^{[i]}, \hat{\boldsymbol{\tau}}^{[i-1]})$ with inner RS algorithm;

until $-2\ell(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\sigma}}, \hat{\boldsymbol{\nu}}, \hat{\boldsymbol{\tau}})$ *has converged*;

The inner RS Algorithm is a modified backfitting algorithm analogous to the one proposed by Hastie & Tibshirani (1990). The idea is to use partial pseudo residuals through the current estimates and a modified response variable $\boldsymbol{z}_k$, which is given by

$$\underset{(n\times 1)}{\boldsymbol{z}_k} = \underset{(n\times 1)}{\boldsymbol{\eta}_k} - \underbrace{\text{diag}\left\{\boldsymbol{\Sigma}(\theta_{ik})^{-1}\left(\frac{d\theta_{ik}}{d\eta_{ik}}\right)^2\right\}}_{\boldsymbol{W}_{(n\times n)}} \underset{(n\times 1)}{\frac{\partial \ell}{\partial \boldsymbol{\eta}_k}}$$

.

Here $\boldsymbol{\Sigma}(\boldsymbol{\theta}_k)$ is the expected Fisher information or the observed Fisher information w.r.t. $\boldsymbol{\theta}_k$, which is used to scale the squared derivative matrix of the parameter $\boldsymbol{\theta}_k$ w.r.t. the linear predictor $\boldsymbol{\eta}_k$. This yields the weight matrix $\boldsymbol{W}_k$, which is also used to obtain the partially reweighted least squares estimates in the modified backfitting algorithm.

The estimates for $\boldsymbol{\beta}_k$ or $\boldsymbol{\gamma}_{jk}$ are obtained by omitting the respective prediction terms of $\boldsymbol{X}_k\hat{\boldsymbol{\beta}}_k$ or $\boldsymbol{Z}_{jk}\hat{\boldsymbol{\gamma}}_{jk}$ when computing the partial residuals. The partial residuals then will be used as the response, when computing the weighted least squares estimate $\hat{\boldsymbol{\beta}}_k$ each $\hat{\boldsymbol{\gamma}}_{jk}$ using the weights $\boldsymbol{W}_k$, which can be seen in algorithm 2.

Once the estimates have been computed, the local ML estimation for obtaining estimates of each λ_{jk} is applied. The method makes use of the well-established relationship between penalized smoothing and a random effect model (see Wood 2017). In it is assumed that the partial residuals $\boldsymbol{\varepsilon}_{jk}$ and the parameters $\boldsymbol{\gamma}_{jk}$ are assumed to behave like a normally distributed variable, such that

$$\boldsymbol{\varepsilon}_{jk} = \boldsymbol{Z}_{jk}\boldsymbol{\gamma}_{jk} + \boldsymbol{e}_{jk}; \quad \boldsymbol{e}_{jk} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2_{jk}\boldsymbol{W}^{-1}_{jk}); \quad \boldsymbol{\gamma}_{jk} \sim \mathcal{N}(\boldsymbol{0}, \tau^2_{jk}\boldsymbol{S}^{-1}_{jk}).$$

Essentially one assumes that $\boldsymbol{\varepsilon}_{jk}$ has two sources of variation: $\boldsymbol{\gamma}_{jk}$, which are treated here as random effects, and the error term $\boldsymbol{e}_{jk}$.

Let furthermore the order of penalty for each $\boldsymbol{\gamma}_{jk}$, determined by the penalty matrix $\boldsymbol{S}_{jk}$, depend on only one λ_{jk}. Then an estimator for λ_{jk} is given by

$$\hat{\lambda}_{jk} = \frac{\sigma^2_{jk}}{\tau^2_{jk}},$$

which is derived from the posterior density $f(\boldsymbol{\gamma}_{jk}|\sigma^2_{jk}, \tau^2_{jk}, \boldsymbol{\varepsilon}_{jk})$. Estimators for σ^2_{jk} and τ^2_{jk} are in turn obtained by

$$\hat{\sigma}^2_{jk} = \frac{\|\boldsymbol{\varepsilon}_{jk} - \hat{\boldsymbol{\varepsilon}}_{jk}\|^2}{(n - tr(\boldsymbol{A}))} \quad \text{and} \quad \hat{\tau}^2_{jk} = \frac{\hat{\boldsymbol{\gamma}}^\top_{jk}\hat{\boldsymbol{\gamma}}_{jk}}{tr(\boldsymbol{A})},$$

where $tr(\boldsymbol{A}) = tr(\boldsymbol{Z}_{jk}(\boldsymbol{Z}_{jk}^\top \boldsymbol{W}_k \boldsymbol{Z}_{jk} + \hat{\lambda}_{jk}\boldsymbol{S}_{jk})^{-1}\boldsymbol{Z}_{jk}^\top \boldsymbol{W}_k)$ and can be considered the effective degree of freedom of the working linear mixed model (Wood 2017). As the estimates of $\boldsymbol{\gamma_{jk}}$ depend on the estimate for $\boldsymbol{\lambda_{jk}}$ this cycle is repeated until all parameters have converged (see algorithm 2).

Algorithm 2: Local ML estimation of λ within the inner RS algorithm

Input: $\hat{\boldsymbol{\theta}}_k^{[i-1]}, \hat{\boldsymbol{\gamma}}_k^{[i-1]}$
Output: $\hat{\boldsymbol{\theta}}_k^{[i]}, \hat{\boldsymbol{\beta}}_k, \hat{\boldsymbol{\gamma}}_k^{[i]}$
Initialize $\hat{\boldsymbol{\lambda}} \in [0, \infty]$
repeat
  The subscript k has been dropped to avoid clutter
  Evaluate current $\boldsymbol{z}$, $\boldsymbol{W}$ and $\frac{\partial \ell}{\partial \boldsymbol{\eta}}$
  repeat
    Obtain partial residuals: $\boldsymbol{\varepsilon} = \boldsymbol{z} - \sum_{j=1}^{J} \boldsymbol{Z}_j \hat{\boldsymbol{\gamma}}_j$
    Compute: $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{\varepsilon}$
    for $t = 1 \rightarrow J$ **do**
      Obtain partial residuals: $\boldsymbol{\varepsilon}_t = \boldsymbol{z} - \boldsymbol{X}\hat{\boldsymbol{\beta}} - \sum_{j=1, t\neq j}^{J} \boldsymbol{Z}_j \hat{\boldsymbol{\gamma}}_j$
      repeat
        Compute:
          $\hat{\boldsymbol{\gamma}}_t = (\boldsymbol{Z}_t^\top \boldsymbol{W} \boldsymbol{Z}_t + \hat{\lambda}_t \boldsymbol{S}_t)^{-1} \boldsymbol{Z}_t^\top \boldsymbol{W} \boldsymbol{\varepsilon}_t$
          predicted partial residuals: $\hat{\boldsymbol{\varepsilon}}_t = \boldsymbol{Z}_t \hat{\boldsymbol{\gamma}}_t$
          hat matrix: $\boldsymbol{A}_t = \boldsymbol{Z}_t (\boldsymbol{Z}_t^\top \boldsymbol{W} \boldsymbol{Z}_t + \hat{\lambda}_t \boldsymbol{S}_t)^{-1} \boldsymbol{Z}_t^\top \boldsymbol{W}$
          variance estimates: $\hat{\sigma}_t^2 = \|\boldsymbol{\varepsilon}_t - \hat{\boldsymbol{\varepsilon}}_t\|^2 / (n - tr(\boldsymbol{A}))$; $\hat{\tau}_t^2 = \hat{\boldsymbol{\gamma}}_t^\top \hat{\boldsymbol{\gamma}}_t / tr(\boldsymbol{A})$
          smoothing parameter: $\hat{\lambda}_t = \frac{\hat{\sigma}_t^2}{\hat{\tau}_t^2}$
      until $\hat{\lambda}_t$ *has converged*;
    end
  until $\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\gamma}}$ *have converged*;
until $-2\ell(\hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_k, \ldots, \hat{\boldsymbol{\theta}}_q)$ *has converged*;

It should be noted that the procedure of obtaining estimates of the smoothing parameters for the smoothing terms of one distribution parameter $\boldsymbol{\theta}_k$ in the inner RS-algorithm is equivalent to the penalized quasi-likelihood (PQL) method as proposed by Breslow & Clayton (1993). The PQL method has been studied extensively and it can be shown that for low-mean count data and overdispersion, the PQL method performs poorly (Wood 2011). Furthermore it is noteworthy that while Rigby & Stasinopoulos (2014) call it local "maximum likelihood" estimation of the smoothing terms, the variance estimates and the effective degrees of freedom are actually based on the restricted maximum likelihood of the working linear mixed model (Wood 2017, p. 150).

2.3.2 Estimation of $\boldsymbol{\lambda}$ in `mgcv`

Similarly to the local ML method described before, the nested iteration case takes an empirical Bayes approach by assuming that not only the parameters $\boldsymbol{\gamma}_{jk}$ have a zero-mean improper prior normal distribution $f_\lambda(\boldsymbol{\gamma})$ with the variance being some generalized inverse of the penalization Matrix $\boldsymbol{S}_{jk}$, but also the parameters associated with the linear terms $\boldsymbol{\beta}_{jk}$ such that $\boldsymbol{\gamma}_{jk} \sim \mathcal{N}(\mathbf{0}, \lambda_{jk}\mathbf{S}_{jk}^{-1})$ and $\boldsymbol{\beta}_{jk} \sim \mathcal{N}(\mathbf{0}, \lambda_{jk}\mathbf{S}_{jk}^{-1})$ (Wood et al. 2016). Henceforth the parameters belonging to the linear terms shall, for notational simplicity, not be distinguished in this section and be all considered $\boldsymbol{\gamma}$. If we take this approach, $\boldsymbol{\lambda}$ can be estimated to maximize the marginal log-likelihood.

$$\mathcal{V}(\boldsymbol{\lambda}) = \log \int f(\boldsymbol{y}|\boldsymbol{\gamma}) f_\lambda(\boldsymbol{\gamma}) d\boldsymbol{\gamma} \tag{1}$$

which is equivalent to REML estimation of the variance components in the mixed model case (Wood et al. 2016). As the integral in (1) is usually high-dimensional and therefore computationally intractable, one option is to use Laplace-approximation. Let $\boldsymbol{\mathcal{S}} = \text{diag}(\lambda_{11}\boldsymbol{S}_{11}, \ldots, \lambda_{J_k q}\boldsymbol{S}_{J_k q})$ be a block-diagonal containing all penalization matrices and $\boldsymbol{\mathcal{H}}$ the negative hessian with elements $-\frac{\partial \ell_{pen}(\hat{\boldsymbol{\gamma}})}{\partial \gamma_{ui} \partial \gamma_{vj}}$. The log Laplace-approximate marginal likelihood (LAML) can be derived by means of Taylor-expansion of $f(\boldsymbol{y}|\boldsymbol{\gamma})$ around the maximizer $\hat{\boldsymbol{\gamma}}$, which yields

$$\begin{aligned}
&\log \int f(\boldsymbol{y}|\boldsymbol{\gamma}) f_\lambda(\boldsymbol{\gamma}) d\boldsymbol{\gamma} \simeq \\
&\simeq \log \int \exp\left\{ \ell(\hat{\boldsymbol{\gamma}}) + \underbrace{\frac{\partial \ell(\hat{\boldsymbol{\gamma}})}{\partial \boldsymbol{\gamma}^\top}(\boldsymbol{\gamma} - \hat{\boldsymbol{\gamma}})}_{0} - \frac{1}{2}(\boldsymbol{\gamma} - \hat{\boldsymbol{\gamma}})^\top \boldsymbol{\mathcal{H}}(\boldsymbol{\gamma} - \hat{\boldsymbol{\gamma}}) + \log f_\lambda(\boldsymbol{\gamma}) \right\} d\boldsymbol{\gamma} \\
&= \log \int \exp\left\{ \ell(\hat{\boldsymbol{\gamma}}) - \frac{1}{2}(\boldsymbol{\gamma} - \hat{\boldsymbol{\gamma}})^\top \boldsymbol{\mathcal{H}}(\boldsymbol{\gamma} - \hat{\boldsymbol{\gamma}}) - \frac{1}{2}\hat{\boldsymbol{\gamma}}^\top \boldsymbol{\mathcal{S}} \hat{\boldsymbol{\gamma}} + \log |\boldsymbol{\mathcal{S}}|_+^{1/2} - \frac{1}{2}\log(2\pi)(p - M_p) \right\} d\boldsymbol{\gamma} \\
&= \log \int \exp\left\{ \ell_{pen}(\hat{\boldsymbol{\gamma}}) - \frac{1}{2}(\boldsymbol{\gamma} - \hat{\boldsymbol{\gamma}})^\top \boldsymbol{\mathcal{H}}(\boldsymbol{\gamma} - \hat{\boldsymbol{\gamma}}) + \log |\boldsymbol{\mathcal{S}}|_+^{1/2} - \frac{1}{2}\log(2\pi)(p - M_p) \right\} d\boldsymbol{\gamma} \\
&= \ell_{pen}(\hat{\boldsymbol{\gamma}}) + \log |\boldsymbol{\mathcal{S}}|_+^{1/2} + \frac{1}{2}\log(2\pi)(M_p - p) + \log \underbrace{\int \exp\left\{ -\frac{1}{2}(\boldsymbol{\gamma} - \hat{\boldsymbol{\gamma}})^\top \boldsymbol{\mathcal{H}}(\boldsymbol{\gamma} - \hat{\boldsymbol{\gamma}}) \right\} d\boldsymbol{\gamma}}_{\text{Invers normalization constant of } \mathcal{N}(\boldsymbol{\gamma}, \boldsymbol{\mathcal{H}}^{-1})} \\
&= \ell_{pen}(\hat{\boldsymbol{\gamma}}) + \log |\boldsymbol{\mathcal{S}}|_+^{1/2} - \log |\boldsymbol{\mathcal{H}}|^{1/2} + \frac{M_p}{2}\log(2\pi).
\end{aligned}$$

Here M_p is the number of zero eigenvalues of $\boldsymbol{\mathcal{S}}$, $|\boldsymbol{\mathcal{S}}|_+$ is the product of the positive eigenvalues of $\boldsymbol{\mathcal{S}}$, and $|\boldsymbol{\mathcal{H}}|$ is the determinant of $\boldsymbol{\mathcal{H}}$.

As mentioned beforehand, the optimization algorithm consists of an outer and an inner loop. Within the outer loop, the LAML criterion $\mathcal{V}(\boldsymbol{\lambda})$ is optimized with respect to $\log(\boldsymbol{\lambda})$, to acquire estimates $\hat{\boldsymbol{\lambda}}$. Before obtaining the new $\hat{\boldsymbol{\lambda}}$ this outer loop, the inner loop is performed to obtain estimates $\hat{\boldsymbol{\gamma}}$, via Newton's method by applying step length control. While there is certainly more depth to this algorithm, with e.g. methods being applied in the algorithm to ensure speed and stability in the inner and outer loop, they shall be not further mentioned here and can be found in Wood et al. (2016). One core advantage is that this method will converge under very mild regularity conditions, because it optimizes a single properly defined function of the smoothing parameters as opposed to a succession of different functions based on successive linearized models as it is the case with the PQL method (Wood 2017).

It can therefore be expected that due to the almost sure convergence of the nested iteration algorithm, this method will outperform the local ML method, especially in situations of high concurvity and low-mean overdispersed count data.

3 Simulation

3.1 Methodology

The objective of the simulation is to extend part of simulation conducted by Wood et al. (2016, p. 1557-1558). In their simulation, Wood et al. (2016) compared the implementation of smoothness-selection criteria implemented in the R-packages `mgcv` and `gamlss`, which use - as established earlier - the nested-iteration method and the local ML method respectively. The comparison is valid, as both models use the same penalized log-likelihood formulation. The simulation done in Wood et al. (2016) compared both implementations for various distributions, which have been implemented in `mgcv` so far. As mentioned before, in this paper we only extend this simulation by Wood et al. (2016) for the zero-inflated Poisson distribution, which was introduced in chapter 2.1. Going forward the simulation approach in this paper shall be laid out in greater detail and deviations from Wood et al. (2016) shall be mentioned where appropriate.
The data which is used as input to `mgcv::gam()` and `gamlss::gamlss()`, is generated by assuming an additive structure for the linear predictors η_μ and η_π, with the subscripts on each linear predictor signifying the belonging to the respective distribution parameter μ and π of the zero-inflated Poisson distribution. η_μ is assumed to have the structure $\eta_\mu/k = f_1(x_1) + f_2(x_2) + f_3(x_3)$, equivalent to Wood et al. (2016), where

$$
\begin{aligned}
f_1(x) &= 2\sin(\pi x) \\
f_2(x) &= \exp(2x) \\
f_3(x) &= 0.2x^{11}(10(1-x))^6 + 10(10x)^3(1-x)^{10}
\end{aligned}
$$

and k is a scaling factor controlling the noise level. To map η_μ to the appropriate scale of μ, the log-link was used. The functions as well as $\mu = \exp(\eta_\mu)$ in dependence of x, where x is assumed to range from 0 and 1, can be seen in figure 2, where all functions are scaled by 0.1. This was necessary - also in the simulation -, as otherwise this would have led to integer overflow, as the counts would have been extremely high. This is done differently to the procedure in Wood et al. (2016).

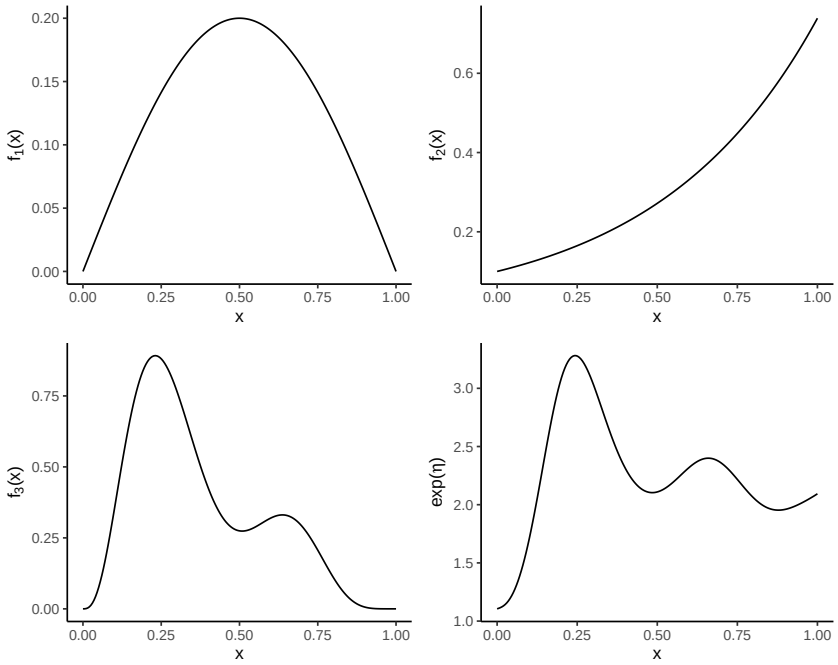

Figure 2: Components and true form of the linear predictor η_μ

Another thing that is done differently, which can be also seen as the core extension to the simulation done in Wood et al. (2016), is that the probability of zero-inflation π is modelled by covariates. The linear predictor η_π is assumed to have the true form $\eta_\pi/k = f_1(x_1) - f_2(x_2) + 1$. Currently `mgcv` only provides the option to use the complementary-log-log-link to map η_π to the appropriate range. Therefore the true underlying response function $g_\pi^{-1}()$ was assumed to be $g_\pi^{-1}(\eta_\pi) = 1 - \exp(-\exp(\eta_\pi))$. The non-linear functions of the linear predictor, the linear predictor itself and the probability

π can be found in figure 3.

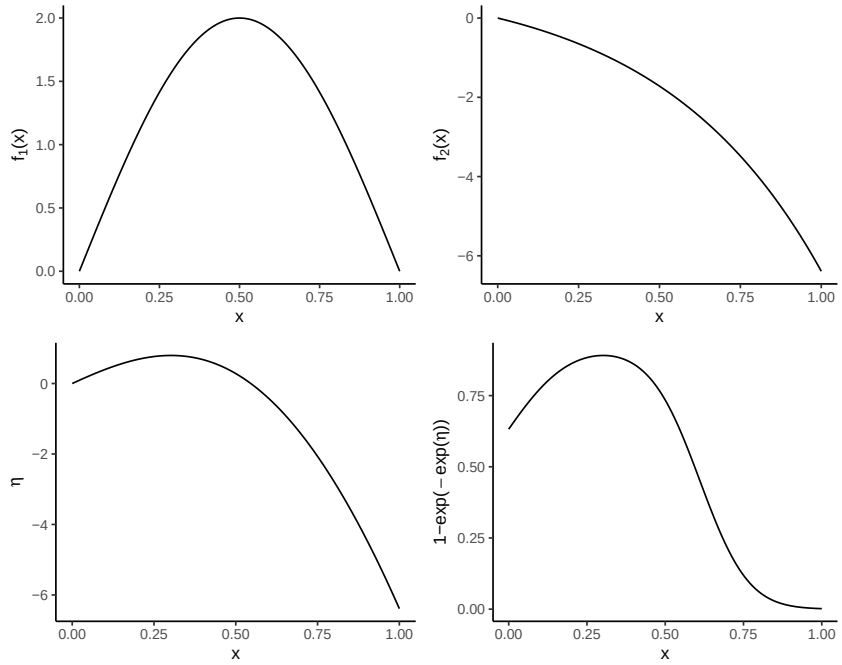

Figure 3: Components and true form of the linear predictor η_π

The comparisons of both implementations are compared for different data situations. The data situations are constructed by the cases, where the covariates are assumed to be correlated or not (2 situations), whether the target variable y has a low mean or not (2 situations), and how large the signal to noise ratio is (3 situations: low, medium, or high). We therefore have $2 \times 2 \times 3$-situations for which the comparisons are computed. Table 1 displays the settings for the scaling factor k for both linear predictors showing how high, medium, and low noise was achieved. The approximate r^2 refers to the amount of variance explained by the scaled linear predictor when compared to linear predictor without noise - the true variable. The uncorrelated covariates x_{1i}, x_{2i}, x_{3i}, and x_{4i}[2] are drawn from a uniform distribution ranging from 0 to 1. In the correlated case the random variables z_{1i}, z_{2i}, z_{3i}, and z_{4i} are drawn from a multivariate normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ with $\mathbf{\Sigma}$ having a unit diagonal and 0.9 for all other elements. Each z_{ji} is then transformed, such that $x_{ji} = \Phi^{-1}(z_{ji})$ where $\Phi(x)$ is the standard normal cumulative density function.

Once the covariates and target variables are generated, the model functions are then applied to that data. For the linear predictor η_μ a basis dimension of 10, 10, 15, and 8 is selected for the covariates x_1, x_2, x_3, and x_4 respectively. For the linear predictor η_π a basis

[2]The covariate x_4 is not included in any of the true linear predictors is used to induce extra noise to the model

	η_μ		η_π	
	scaling factor	approx. r^2	scaling factor	approx. r^2
high noise	3.00	0.54	6.00	0.58
medium noise	1.50	0.67	2.20	0.75
low noise	1.25	0.81	1.40	0.91

Table 1: Scaling factors for different noise levels and corresponding approximated r^2

dimension of 10, 10, and 8 is selected for the covariates x_1, x_2, and x_4 respectively. As for the function `mgcv::gam()`, P-splines are chosen explicitly and the family argument `mgcv::ziplss()` is used to model both μ and π explicitly. Beyond that the function is executed with default settings. As for the function `gamlss::gamlss()`, P-Splines are chosen to be fitted with ML, which constitutes the local ML estimation described in section 2.3.1. Furthermore the number of outer RS-iterations is increased to 30 and the family argument `ZIP(sigma.link = "cloglog")` is used, which corresponds to a log-link for μ and a complementary-log-log-link for π.
Once both models have been fitted on the same data, the mean squared error (MSE) $n^{-1}\sum_{i=1}^{n}(\hat{\eta}_i - \eta_i)^2$ for both linear predictors is computed. Furthermore, computation time for both models is taken note of as well as the information if the algorithm converged or not.
The simulation is performed for 12 different data situations. For each data situation 300 data sets with 400 observations were generated. The results of this simulation shall be laid out in the next section.

3.2 Results

The simulation[3] was carried out in R version 3.6.1 on a Windows operating system with a Intel(R) Core(TM) i5-6300U 2.40 GHz processor. The runtime of the simulation using multisession parallelization was 45.5 hours.
Firstly, we look at the differences. Figure 4 and figure 5 show the standardized differences in MSE between the `mgcv::gam()` and `gamlss::gamlss()` for normal count and low count respectively. Here "normal" means that the count was not coerced to have severe low count (e.g. a mean below 1). A positive difference in MSE signifies a better performance of `mgcv::gam()` and lower MSE for this algorithm. For differences in MSE a t-test was performed, to test the hypothesis if the nested-iteration case in the `mgcv` package yields a lower MSE than the local ML method in the `gamlss` package (alternative hypothesis) versus the assumption that there is now difference between these implementations (null hypothesis). If we take a closer look at the results for the normal counts in figure 4, we can see that the null hypothesis was rejected 8 out of 12 times at the 5% level. For both, correlated and uncorrelated data, there seems to be no difference

[3]The code can be found at https://github.com/alexpiehler/gamlss_vs_mgcv_simulation

in the situation where there is little induced noise to the linear predictor η_μ. The same applies to the linear predictor of η_π, when there is a lot of induced noise. We interpret this as the nested iteration case and the local ML method as performing equally well when they estimate the mean with low noise and performing equally poor when estimating the probability of zero-inflation when there is a lot of noise. For all other cases the nested iterations method has on average a significant lower MSE. While on average the nested iteration methods seems to yield better results, this superiority seems to be very much influenced by bad outliers in the fitting with the local method. For almost all cases the median standardized difference is below 0, which means that the local ML method actually outperformed the nested iteration method in most cases.

Interestingly, Wood et al. (2016) found that the local ML method performed better with uncorrelated zero-inflated Poisson data, where the level of zero-inflation was assumed to be a constant. When just looking at averages, this is not the case in this simulation. We explain these differences with having chosen a scaling factor, to prevent numerical problems in the algorithms due to integer overflow. Furthermore, another explanation could be that through the inclusion of the second parameter π into the model, a further layer of instability is added to the fitting process.

Looking further at the low count case in figure 5, it can be seen that also for 8 out 12 cases the MSE in the nested iteration case is significantly lower than the MSE in the local ML estimation case at the 5% level. In the uncorrelated case, the nested iteration algorithm outperforms in all cases except for the case when there is high noise in the linear predictor η_π. This, however, can be mostly attributed to the outliers in the fitting process of the local ML estimation, which skews the average in favor of the nested iteration case. When we once more look at the median of the differences, we can see in most cases the local ML method performed just as well or better than the nested iteration method. Interestingly, for the correlated low count case, in which the local ML estimation algorithm is assumed to perform very poorly, there is no meaningful difference in MSE for low noise when comparing it to the nested iteration case. This makes sense, as low noise also implies low overdispersion. For the estimation of η_π in the correlated low count case, we can see that both algorithms perform equally with high and medium noise.

These results yield mixed evidence on the assumption that the local ML estimation, which is essentially a PQL algorithm when the canonical link is used, performs worse than the nested iteration algorithm. When comparing the low-mean count situations with the normal-mean count situations, there are somewhat more positive valued outliers present in the low-mean count situation, which suggests a greater instability of the local ML method for these cases. However, it can be again observed that also here in most cases the local ML method performs equally well when compared to the nested iteration method.

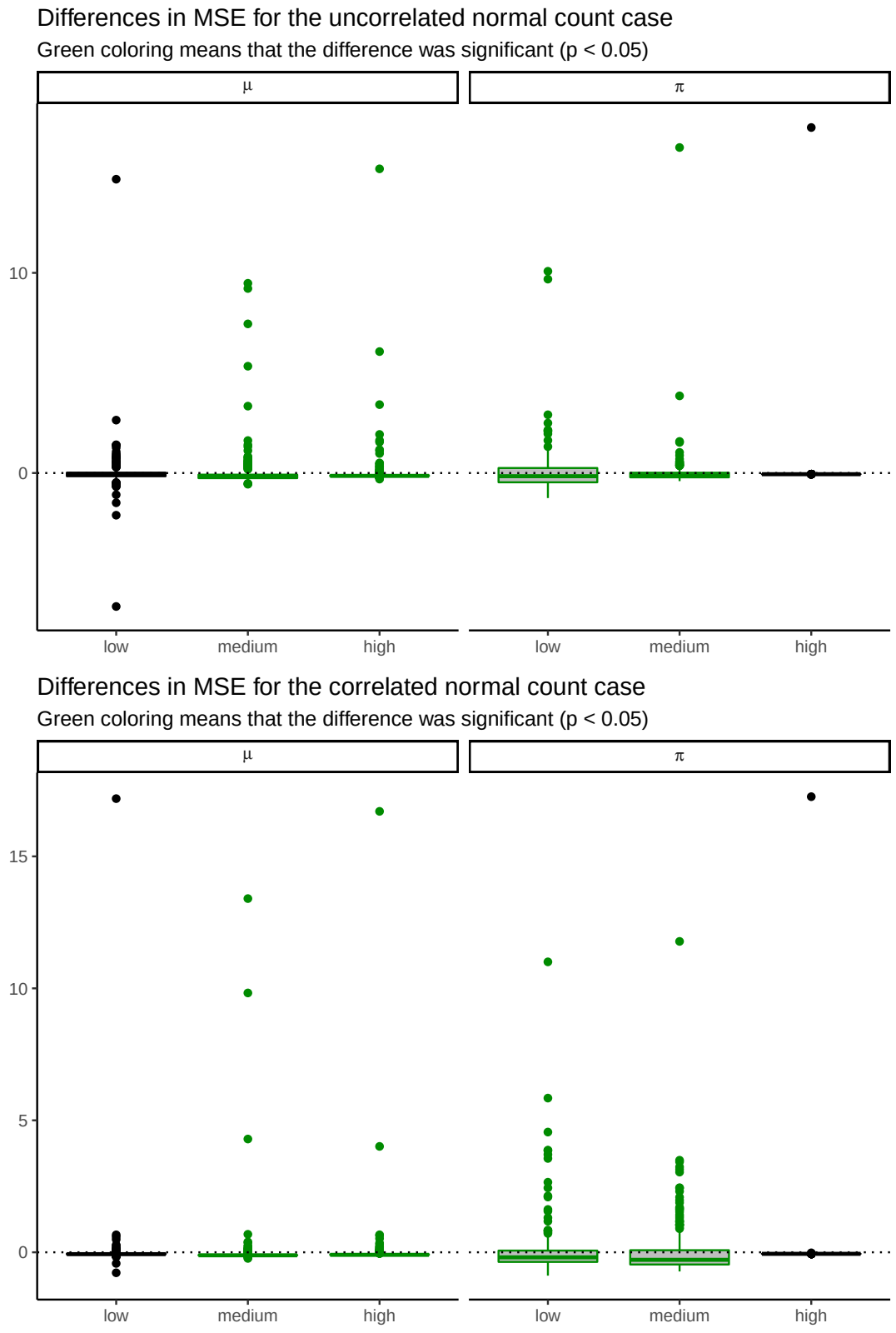

Figure 4: Differences in mean-squared-error for normal count data

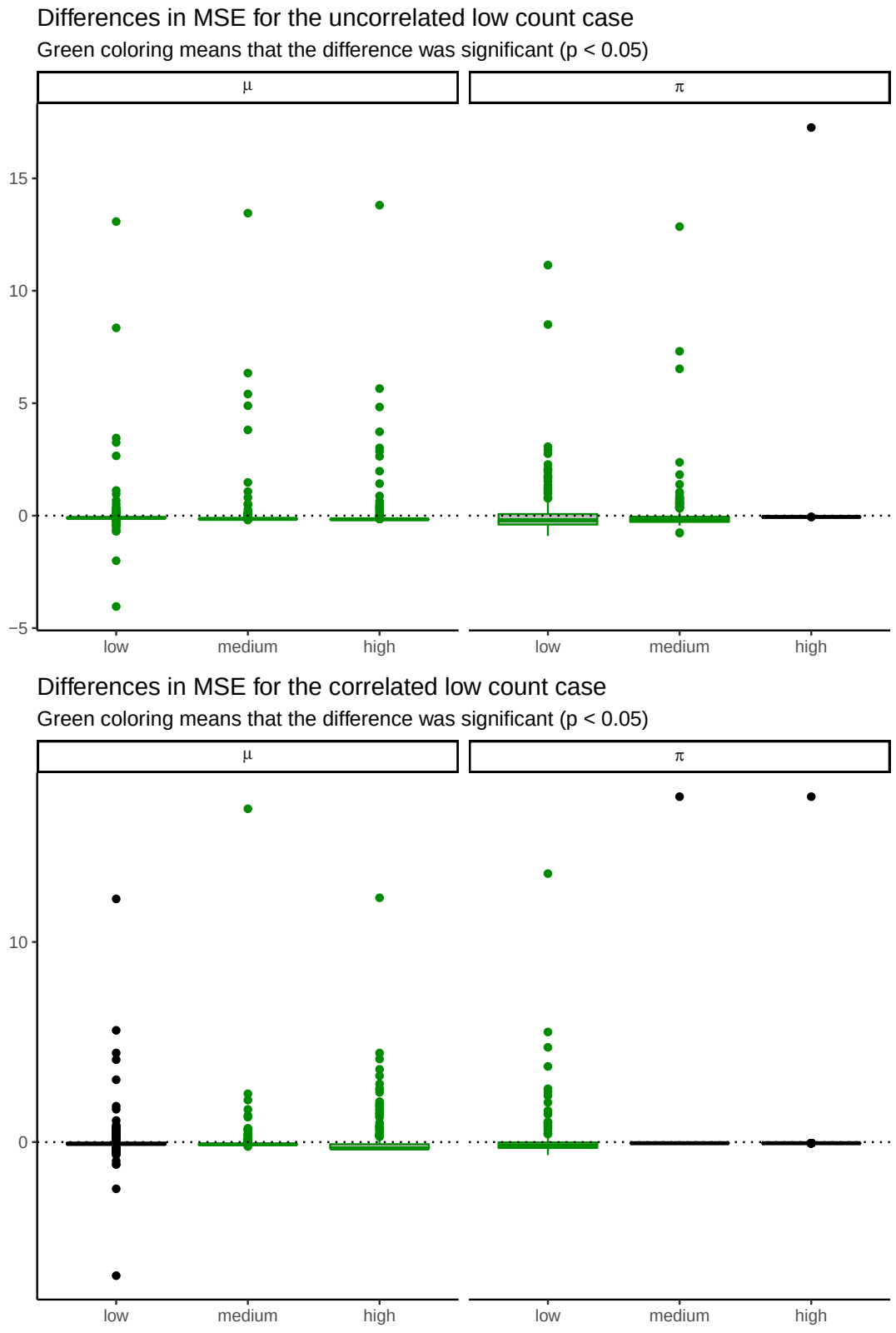

Figure 5: Differences in mean-squared-error for low count data

When looking at the convergence rates in table 2 one can see that the nested iteration algorithm in the `mgcv::gam()` function achieved convergence in almost all $12 \times 300 = 3600$ cases. Merely in the uncorrelated case with high noise, the algorithm failed once, which seems to be an outlier as it also cannot be reasoned theoretically. The results for local ML estimation algorithm in the function `gamlss::gamlss()` are somewhat different. In the uncorrelated case, the assumption that with increasing noise the algorithm will converge increasingly less can be sustained for the normal and the low count - with the low count converging as expected at a lesser rate than the normal count. For the correlated case, the results are paradoxical: in the low noise-low count case better convergence is achieved than in the correlated normal count case - even better convergence than uncorrelated low count case. For the correlated and high noise cases, the algorithm converges only 20% of the times in the normal count case and 26% of the times in the low count cases.

	`mgcv`			`gamlss`		
	low	medium	high	low	medium	high
uncorrelated	1.0000	1.0000	0.9970	0.9670	0.8500	0.6230
uncorrelated / low count	1.0000	1.0000	1.0000	0.9100	0.7870	0.6070
correlated	1.0000	1.0000	1.0000	0.9070	0.7900	0.2030
correlated / low count	1.0000	1.0000	1.0000	0.9400	0.7570	0.2630

Table 2: Convergence rates for the packages `mgcv` and `gamlss` for different data situations

While the convergence is indeed an important measure, it should be noted that the correlation coefficient for the outlier-removed mean difference in MSE between the two algorithms and the convergence rate of `gamlss::gamlss()` is about -0.08, which suggests that there is little association between convergence of the algorithms and the performance of both models. This can most likely be attributed to the rather strict convergence criterion in the `gamlss::gamlss()`, in which convergence is not obtained, yet the performance is still acceptable. There seems to be much rather an association between severe outliers in MSE and the convergence rates.

Lastly, we take a look at the computing times for each algorithm. For all cases the computational time needed to fit a model is less in `mgcv`, which can be seen in figure 6. When looking at all 3600 fitted models, the median duration for fitting a model with `mgcv` was 2.12 seconds and with `gamlss` 68.07 seconds. These results mirror the results of Wood et al. (2016).

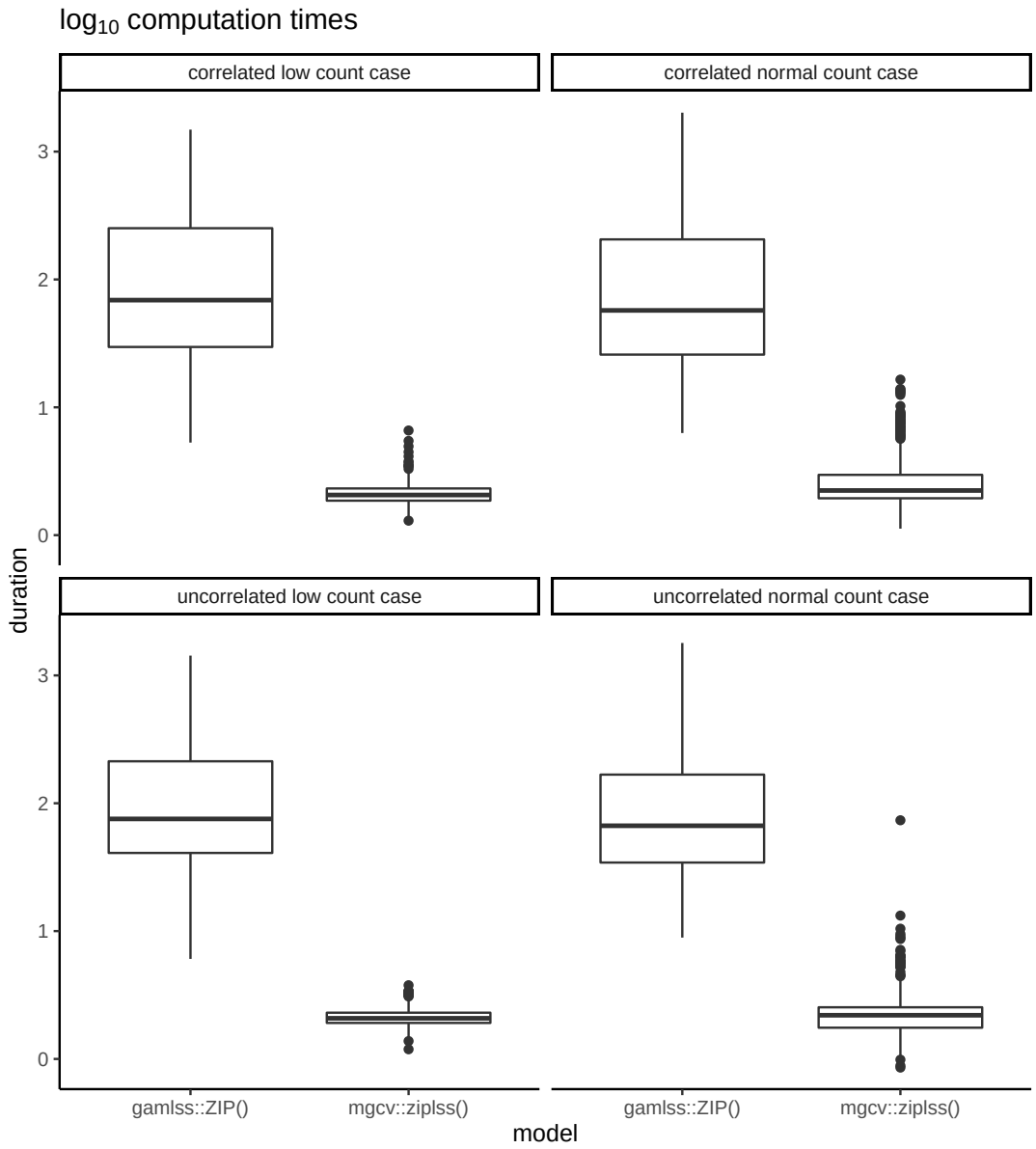

Figure 6: $\log_{10}$-transformed computation time (seconds) of algorithms

4 Conclusion

Generalized Additive Models for Location, Scale and Shape are a powerful tool in statistical modelling. They enable the practitioner to model a diverse set of models, in which the response variable stems from a multi-parametric distribution and the distribution parameters depend on some set of covariates. Just as in Generalized Additive Models the question arises how to select the appropriate smoothing parameters. Among many methods implemented `gamlss`, the fastest and most reliable one is the local ML method (see Rigby & Stasinopoulos 2014). This method estimates the smoothing parameter for each smooth individually. Another method for modelling some multi-parametric distributions in the context of GAMLSS is given by the package `mgcv`, which uses a fast and stable implementation of a nested algorithm, in which the smoothness parameters are obtained through restricted maximum likelihood estimation, where the integration of the coefficients is approximated by Laplace-approximation. Here the smoothing parameters are estimated jointly via Newton's method.

The results of the simulation corroborate some, but not all findings of (Wood et al. 2016). While the nested iteration case performs significantly better, this is mostly due to the instability of the local ML estimation producing occasional outlier results. If we disregard the instability and outliers, local ML estimation performs in most cases just as good if not better for all simulated data situations. Also in the case of low count response data, presence of medium to high noise merely increases the instability of the local ML method. It could also be shown that, contrary to the findings of (Wood et al. 2016), the nested iteration method performs on average better with uncorrelated data with zero-inflated Poisson data. This, however, might be attributable to the changed circumstances under which this simulation was conducted: the now explicit modelling of the probability of zero-inflation through covariates.

While the algorithm in `mgcv` almost always converges, the algorithm in `gamlss` still achieves comparable performance even with a very low convergence rate for a situation. Complete convergence does not seem the be a deciding factor in the case of this simulation. Convergence rates, however, do seem to be indicative of unstable results for a given data situation. Lastly, it can be observed that in regard speed the algorithm in `mgcv` outperforms the algorithm in `gamlss` even for location-scale modeling.

These findings point towards an overall advantage of the `mgcv` package over the `gamlss` package in location-scale modeling due to superior stability and robustness. So far, the package `mgcv` offers explicit modeling for the location and scale parameters for the Gaussian model family, the zero-inflated Poisson model family, the Multinomial model family, and the Generalized Extreme Value location-scale model family. The `gamlss` package offers support for over 80 distributions and also provides means to implement new distributions (Stasinopoulos et al. 2017). When encountering problems, which require the

use of one of the distribution families, which are implemented in the `mgcv` package, it is sensible to also use these. However, the `gamlss` offers greater flexibility with just as good if not better results. One should, however, expect greater instability in results, a longer runtime, and possible convergence issues. It is therefore advised to try different sensible starting values for the algorithm in `gamlss` and use model selection techniques to discern the best settings. This procedure could mitigate problems with instability.

References

Breslow, N. E. & Clayton, D. G. (1993), 'Approximate inference in generalized linear mixed models', *Journal of the American Statistical Association* **88**(421), 9.

Hastie, T. & Tibshirani, R. (1990), *Generalized Additive Models*, Chapman and Hall/CRC.

Rigby, R. A. & Stasinopoulos, D. M. (2005), 'Generalized additive models for location, scale and shape', *Journal of the Royal Statistical Society* **54**(3), 507–554.

Rigby, R. A. & Stasinopoulos, D. M. (2014), 'Automatic smoothing parameter selection in gamlss with an application to centile estimation', *Statistical methods in medical research* **23**(4), 318–332.

Stasinopoulos, M. D., Rigby, R. A., Heller, G. Z., Voudouris, V. & de Bastiani, F. (2017), *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman & Hall / CRC The R Series, CRC Press, London.

Tutz, G. (2011), *Regression for Categorical Data*, Cambridge University Press.

Umlauf, N., Klein, N. & Zeileis, A. (2018), 'BAMLSS: Bayesian additive models for location, scale, and shape (and beyond)', *Journal of Computational and Graphical Statistics* **27**(3), 612–627.

Wood, S. N. (2011), 'Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models', *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* **73**(1), 3–36.

Wood, S. N. (2017), *Generalized Additive Models: An Introduction with R*, Chapman & Hall / CRC Texts in Statistical Science, 2 edn, CRC Press, Portland.

Wood, S. N., Pya, N. & Säfken, B. (2016), 'Smoothing parameter and model selection for general smooth models', *Journal of the American Statistical Association* **111**(516), 1548–1563.