Stat Comput (2018) 28:673–687
https://doi.org/10.1007/s11222-017-9754-6

Gradient boosting for distributional regression: faster tuning and improved variable selection via noncyclical updates

Janek Thomas[1] · **Andreas Mayr**[2,3] · **Bernd Bischl**[1] · **Matthias Schmid**[3] · **Adam Smith**[4] · **Benjamin Hofner**[5]

Received: 3 November 2016 / Accepted: 5 May 2017 / Published online: 11 May 2017

Abstract We present a new algorithm for boosting generalized additive models for location, scale and shape (GAMLSS) that allows to incorporate stability selection, an increasingly popular way to obtain stable sets of covariates while controlling the per-family error rate. The model is fitted repeatedly to subsampled data, and variables with high selection frequencies are extracted. To apply stability selection to boosted GAMLSS, we develop a new "noncyclical" fitting algorithm that incorporates an additional selection step of the best-fitting distribution parameter in each iteration. This new algorithm has the additional advantage that optimizing the tuning parameters of boosting is reduced from a multi-dimensional to a one-dimensional problem with vastly decreased complexity. The performance of the novel algorithm is evaluated in an extensive simulation study. We apply this new algorithm to a study to estimate abundance of common eider in Massachusetts, USA, featuring excess zeros, overdispersion, nonlinearity and spatiotemporal structures. Eider abundance is estimated via boosted GAMLSS, allowing both mean and overdispersion to be regressed on covariates. Stability selection is used to obtain a sparse set of stable predictors.

Keywords Boosting · Additive models · GAMLSS · GamboostLSS · Stability selection

1 Introduction

In view of the growing size and complexity of modern databases, statistical modeling is increasingly faced with heteroscedasticity issues and a large number of available modeling options. In ecology, for example, it is often observed that outcome variables do not only show differences in *mean* conditions but also tend to be highly *variable* across different geographical features or states of a combination of covariates (e.g., Osorio and Galiano 2012). In addition, ecological databases typically contain large numbers of correlated predictor variables that need to be carefully chosen for possible incorporation in a statistical regression model (Aho et al. 2014; Dormann et al. 2013; Murtaugh 2009.

A convenient approach to address both heteroscedasticity and variable selection in statistical regression models is the combination of GAMLSS modeling with gradient boosting algorithms. GAMLSS, which refer to "generalized additive models for location, scale and shape" (Rigby and Stasinopoulos 2005), are a modeling technique that relates not only the mean but all parameters of the outcome distribution to the available covariates. Consequently, GAMLSS simultaneously fit different submodels for the location, scale and shape parameters of the conditional distribution. Gradient boosting, on the other hand, has become a popular tool for data-driven variable selection in generalized additive models

Electronic supplementary material The online version of this article (doi:10.1007/s11222-017-9754-6) contains supplementary material, which is available to authorized users.

✉ Janek Thomas
janek.thomas@stat.uni-muenchen.de

[1] Department of Statistics, Ludwig-Maximilians-Universität München, Ludwigstrasse 33, 80539 Munich, Germany

[2] Department of Medical Informatics, Biometry and Epidemiology, FAU Erlangen-Nürnberg, Erlangen, Germany

[3] Department of Medical Biometry, Informatics and Epidemiology, University Hospital Bonn, Bonn, Germany

[4] U.S. Fish & Wildlife Service, National Wildlife Refuge System, Southeast Inventory & Monitoring Branch, Lewistown, MT, USA

[5] Section Biostatistics, Paul-Ehrlich-Institute, Langen, Germany

(Bühlmann and Hothorn 2007). The most important feature of gradient boosting is the ability of the algorithm to perform variable selection in each iteration, so that model fitting and variable selection are accomplished in a single algorithmic procedure. To combine GAMLSS with gradient boosting, we have developed the gamboostLSS algorithm (Mayr et al. 2012) and have implemented this procedure in the R add-on package **gamboostLSS** (Hofner et al. 2016, 2017).

A remaining problem of gradient boosting is the tendency of boosting algorithms to select a relatively high number of false-positive variables and to include too many noninformative covariates in a statistical regression model. This issue, which has been raised in several previous articles (Bühlmann and Hothorn 2010; Bühlmann and Yu 2006; Huang et al. 2012), is particularly relevant for model building in the GAMLSS framework, as the inclusion of noninformative false positives in the submodels for the scale and shape parameters may result in overfitting with a highly inflated variance. As a consequence, it is crucial to include only those covariates in these submodels that show a relevant effect on the outcome parameter of interest. From an algorithmic point of view, this problem is aggravated by the conventional fitting procedure of gamboostLSS: Although the fitting procedure proposed in Mayr et al. (2012) incorporates different iteration numbers for each of the involved submodels, the algorithm starts with mandatory updates of each submodel at the beginning of the procedure. Consequently, due to the tendency of gradient boosting to include relatively high numbers of noninformative covariates, false-positive variables may enter a GAMLSS submodel at a very early stage, even before the iteration number of the submodel is finally reached.

To address these issues and to enforce sparsity in GAML SS, we propose a novel procedure that incorporates *stability selection* (Meinshausen and Bühlmann 2010) in gamboostLSS. Stability selection is a generic method that investigates the importance of covariates in a statistical model by repeatedly subsampling the data. Sparsity is enforced by including only the most "stable" covariates, in the final statistical model. Importantly, under appropriate regularity conditions, stability selection can be tuned such that the expected number of false-positive covariates is controlled in a mathematically rigorous way. As will be demonstrated in Sect. 3 of this paper, the same property holds in the gamboostLSS framework.

To combine gamboostLSS with stability selection, we present an improved "*noncyclical*" fitting procedure for gamboostLSS that addresses the problem of possible false-positive inclusions at early stages of the algorithm. In contrast to the original "*cyclical*" gamboostLSS algorithm presented in Mayr et al. (2012), the new version of gamboostLSS not only performs variable selection in each iteration but additionally an iteration-wise selection of the best submodel (location, scale, or shape) that leads to the largest improvement in model fit. As a consequence, sparsity is not only enforced by the inclusion of the most "stable" covariates in the GAMLSS submodels but also by a data-driven choice of iteration-wise submodel updates. It is this procedure that theoretically justifies and thus enables the use of stability selection in gamboostLSS.

A further advantage of "*noncyclical*" fitting is that the maximum number of boosting iterations for each submodel does not have to be specified individually for each submodel (as in the originally proposed "*cyclical*" variant), instead only the overall number of iterations must be chosen optimally. Tuning the complete model reduces from a multidimensional to a one-dimensional optimization problem, regardless of the number of submodels, therefore drastically reducing the amount of needed runtime for model selection.

A similar approach for noncyclical fitting of multiparameter models was recently suggested by Messner et al. (2017) for the specific application of ensemble post-processing for weather forecasts. Our proposed method generalizes this approach, allowing gamboostLSS to be combined with stability selection in a generic way that applies to a large number of outcome distributions.

The rest of this paper is organized as follows: In Sect. 2, we describe the gradient boosting, GAMLSS and stability selection techniques and show how to combine the three approaches in a single algorithm. In addition, we provide details on the new gamboostLSS fitting procedure. Results of extensive simulation studies are presented in Sect. 3. They demonstrate that combining gamboostLSS with stability selection results in prediction models that are both easy to interpret and show a favorable behavior with regard to variable selection. They also show that the new gamboostLSS fitting procedure results in a large decrease in runtime while showing similar empirical convergence rates as the traditional gamboostLSS procedure. We present an application of the proposed algorithm to a spatiotemporal data set on sea duck abundance in Nantucket Sound, USA, in Sect. 4. Section 5 summarizes the main findings and provides details on the implementation of the proposed methodology in the R package **gamboostLSS** (Hofner et al. 2017).

2 Methods

2.1 Gradient boosting

Gradient boosting is a supervised learning technique that combines an ensemble of *base-learners* to estimate complex statistical dependencies. Base-learners should be *weak* in the sense that they only possess moderate prediction accuracy, usually assumed to be at least slightly better than a random predictor, but on the other hand easy and fast to fit. Base-learners can be, for example, simple linear regression models, regression splines with low degrees of freedom, or stumps (i.e., trees with only one split; Bühlmann and Hothorn 2007). One base-learner by itself will usually not be enough to fit a well-performing statistical model to the data, but a boosted combination of a large number can compete with other state-of-the-art algorithms on many tasks, e.g., classification (Li 2012) or image recognition (Opelt et al. 2004).

Let $D = \{(x^{(i)}, y^{(i)})\}_{i=1,\dots,n}$ be a learning data set sampled i.i.d. from a distribution over the joint space $\mathcal{X} \times \mathcal{Y}$, with a p-dimensional input space $\mathcal{X} = (\mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_p)$ and a usually one-dimensional output space $\mathcal{Y}$. The response variable is estimated through an additive model where $\mathbb{E}(y|x) = g^{-1}(\eta(x))$, with link function g and additive predictor $\eta : \mathcal{X} \to \mathbb{R}$,

$$\eta(x) = \beta_0 + \sum_{j=1}^{J} f_j(x|\beta_j), \tag{1}$$

with a constant intercept coefficient β_0 and additive effects $f_j(x|\beta_j)$ derived from the pre-defined set of base-learners. These are usually (semi-)parametric effects, e.g., splines, with parameter vector β_j. Note that some effects may later be estimated as 0, i.e., $f_j(x|\beta_j) = 0$. In many cases, each base-learner is defined on exactly one element $\mathcal{X}_j$ of $\mathcal{X}$ and thus Eq. 1 simplifies to

$$\eta(x) = \beta_0 + \sum_{j=1}^{p} f_j(x_j|\beta_j). \tag{2}$$

To estimate the parameters $\beta_1, \dots, \beta_J$ of the additive predictor, the boosting algorithm minimizes the *empirical risk* R which is the loss $\rho : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}$ summed over all training data:

$$R = \sum_{i=1}^{n} \rho(y^{(i)}, \eta(x^{(i)})). \tag{3}$$

The loss function measures the discrepancy between the true outcome $y^{(i)}$ and the additive predictor $\eta(x^{(i)})$. Examples are the absolute loss $|y^{(i)} - \eta(x^{(i)})|$, leading to a regression model for the median, the quadratic loss $(y^{(i)} - \eta(x^{(i)}))^2$, leading to the conventional (mean) regression model or the binomial loss $-y^{(i)}\eta(x^{(i)}) + \log(1+\exp(\eta(x^{(i)})))$ often used in classification of binary outcomes $y^{(i)} \in \{0, 1\}$. Very often the loss is derived from the negative log likelihood of the distribution of $\mathcal{Y}$, depending on the desired model (Friedman et al. 2000).

While there exist different types of gradient boosting algorithms (Mayr et al. 2014a, b), in this article we will focus on component-wise gradient boosting (Bühlmann and Yu 2003; Bühlmann and Hothorn 2007). The main idea is to fit simple regression-type base-learners $h(\cdot)$ one by one to the negative gradient vector of the loss $u = (u^{(1)}, \dots, u^{(n)})$ instead of to the true outcomes $y = (y^{(1)}, \dots, y^{(n)})$. Base-learners are chosen in such a way that they approximate the effect $\hat{f}(x|\beta_j) = \sum_m h_j(\cdot)$. The negative gradient vector in iteration m, evaluated at the estimated additive predictor $\hat{\eta}^{[m-1]}(x^{(i)})$, is defined as

$$\boldsymbol{u} = \left(- \frac{\partial}{\partial \eta} \rho(y, \eta) \Big|_{\eta = \hat{\eta}^{[m-1]}(x^{(i)}),\ y = y^{(i)}} \right)_{i=1,\dots,n}.$$

In every boosting iteration, each base-learner is fitted separately to the negative gradient vector by least-squares or penalized least-squares regression. The best-fitting base-learner is selected based on the residual sum of squares with respect to u

$$j^* = \underset{j \in 1\dots J}{\operatorname{argmin}} \sum_{i=1}^{n} (u^{(i)} - \hat{h}_j(x^{(i)}))^2. \tag{4}$$

Only the best-performing base-learner $\hat{h}_{j^*}(x)$ will be used to update the current additive predictor,

$$\hat{\eta}^{[m]} = \hat{\eta}^{[m-1]} + \mathrm{sl} \cdot \hat{h}_{j^*}(x) \tag{5}$$

where $0 < \mathrm{sl} \ll 1$ denotes the step length (learning rate; usually $\mathrm{sl} = 0.1$). The choice of sl is not of critical importance as long as it is sufficiently small (Schmid and Hothorn 2008).

The main tuning parameter for gradient boosting algorithms is the number of iterations m that are performed before the algorithm is stopped (denoted as m_{stop}). The selection of m_{stop} has a crucial influence on the prediction performance of the model. If m_{stop} is set too small, the model cannot fully incorporate the influence of the effects on the response and will consequently have a poor performance. On the other hand, too many iterations will result in *overfitting*, which hampers the interpretation and generalizability of the model.

2.2 GAMLSS

In classical generalized additive models (GAM, Hastie and Tibshirani 1990), it is assumed that the conditional distribution of $\mathcal{Y}$ depends only on one parameter, usually the conditional mean. If the distribution has multiple parameters, all but one are considered to be constant nuisance parameters. This assumption will not always hold and should be critically examined, e.g., the assumption of constant variance is not adequate for heteroscedastic data. Potential dependency of the scale (and shape) parameter(s) of a distribution on predictors can be modeled in a similar way to the conditional mean (i.e., location parameter). This extended model class is called *generalized additive models for location, scale and shape* (GAMLSS, Rigby and Stasinopoulos 2005).

The framework hence fits different prediction functions to multiple distribution parameters $\theta = (\theta_1, \ldots, \theta_k)$, $k = 1, \ldots, 4$. Given a conditional density $p(y|\theta)$, one estimates additive predictors (cf. Eq. 1) for each of the parameters θ_k

$$\eta_{\theta_k} = \beta_{0\theta_k} + \sum_{j=1}^{J_k} f_{j\theta_k}(x|\beta_{j\theta_k}), \qquad k = 1, \ldots, 4. \tag{6}$$

Typically, these models are estimated via penalized likelihood. For details on the fitting algorithm, see Rigby et al. (2008).

Even though these models can be applied to a large number of different situations, and the available fitting algorithms are extremely powerful, they still inherit some shortcomings from the penalized likelihood approach:

1. It is not possible to estimate models with more covariates than observations.
2. Maximum likelihood estimation does not feature an embedded variable selection procedure. For GAMLSS models, the standard AIC has been expanded to the *generalized* AIC (GAIC) in Rigby and Stasinopoulos (2005) to be applied to multi-dimensional prediction functions. Variable selection via information criteria has several shortcomings, for example the inclusion of too many non-informative variables (Anderson and Burnham 2002).
3. Whether to model predictors in a linear or nonlinear fashion is not trivial. Unnecessary complexity increases the danger of overfitting as well as computation time. Again, a generalized criterion like GAIC must be used to choose between linear and nonlinear terms.

2.3 Boosted GAMLSS

To deal with these issues, gradient boosting can be used to fit the model instead of the standard maximum likelihood algorithm. Based on an approach proposed in Schmid et al. (2010) to fit zero-inflated count models, in Mayr et al. (2012) the author developed a general algorithm to fit multi-dimensional prediction functions with component-wise gradient boosting (see Algorithm 1).

The basic idea is to cycle through the distribution parameters θ in the fitting process. Partial derivatives with respect to each of the additive predictors are used as response vectors. In each iteration of the algorithm, the best-fitting base-learner is determined for *each* distribution parameter, while all other parameters stay fixed. For a four parametric distribution, the update in boosting iteration $m+1$ may be sketched as follows:

$$\frac{\partial}{\partial \eta_{\theta_1}} \rho(y, \hat{\theta}_1^{[m]}, \hat{\theta}_2^{[m]}, \hat{\theta}_3^{[m]}, \hat{\theta}_4^{[m]}) \overset{\text{update}}{\longrightarrow} \eta_{\theta_1}^{[m+1]}$$

$$\frac{\partial}{\partial \eta_{\theta_2}} \rho(y, \hat{\theta}_1^{[m+1]}, \hat{\theta}_2^{[m]}, \hat{\theta}_3^{[m]}, \hat{\theta}_4^{[m]}) \overset{\text{update}}{\longrightarrow} \eta_{\theta_2}^{[m+1]}$$

$$\frac{\partial}{\partial \eta_{\theta_3}} \rho(y, \hat{\theta}_1^{[m+1]}, \hat{\theta}_2^{[m+1]}, \hat{\theta}_3^{[m]}, \hat{\theta}_4^{[m]}) \overset{\text{update}}{\longrightarrow} \eta_{\theta_3}^{[m+1]}$$

$$\frac{\partial}{\partial \eta_{\theta_4}} \rho(y, \hat{\theta}_1^{[m+1]}, \hat{\theta}_2^{[m+1]}, \hat{\theta}_3^{[m+1]}, \hat{\theta}_4^{[m]}) \overset{\text{update}}{\longrightarrow} \eta_{\theta_4}^{[m+1]}.$$

Unfortunately, separate stopping values for each distribution parameter have to be specified, as the prediction functions will most likely require different levels of complexity and hence a different number of boosting iterations. In case of multi-dimensional boosting, the different $m_{\text{stop},k}$ values are not independent of each other and have to be jointly optimized. The usually applied *grid search* scales exponentially with the number of distribution parameters and can quickly become computationally demanding or even infeasible.

Algorithm 1 "Cyclical" component-wise gradient boosting in multiple dimensions (Mayr et al. 2012)

```
Initialize
1. Initialize the additive predictors $\hat{\eta}^{[0]} = (\hat{\eta}_{\theta_1}^{[0]}, \hat{\eta}_{\theta_2}^{[0]}, \hat{\eta}_{\theta_3}^{[0]}, \hat{\eta}_{\theta_4}^{[0]})$ with offset values.
2. For each distribution parameter $\theta_k, k = 1, \ldots, 4$, specify a set of base-learners, i.e., for parameter $\theta_k$ define $h_{k1}(x^{(i)}), \ldots, h_{kJ_k}(x^{(i)})$ where $J_k$ is the cardinality of the set of base-learners specified for $\theta_k$.

Boosting in multiple dimensions
For $m = 1$ to $\max(m_{\text{stop},1}, \ldots, m_{\text{stop},4})$:
3. For $k = 1$ to 4:
    (a) If $m > m_{\text{stop},k}$ set $\hat{\eta}_{\theta_k}^{[m]} := \hat{\eta}_{\theta_k}^{[m-1]}$ and skip this iteration.
        Else compute negative partial derivative $-\frac{\partial}{\partial \eta_{\theta_k}} \rho(y, \eta)$ an plug in the current estimates $\hat{\eta}^{[m-1]}(\cdot)$:
            $u_k = \left( \frac{\partial}{\partial \eta_{\theta_k}} \rho(y, \eta) \Big|_{\eta = \hat{\eta}^{[m-1]}(x^{(i)}), y = y^{(i)}} \right)_{i=1,\ldots,n}$
    (b) Fit each of the base-learners $u_k$ contained in the set of base-learners specified for the distribution parameter $\theta_k$ in step (2) to the negative gradient vector.
    (c) Select the component $j^*$ that best fits the negative partial derivative vector according to the residual sum of squares, i.e., select the base-learner $h_{kj^*}$ defined by
            $j^* = \underset{j \in 1,\ldots,J_k}{\operatorname{argmin}} \sum_{i=1}^{n} (u_k^{(i)} - \hat{h}_{kj}(x^{(i)}))^2.$
    (d) Update the additive predictor $\eta_{\theta_k}$
            $\hat{\eta}_{\theta_k}^{[m]} = \hat{\eta}_{\theta_k}^{[m-1]} + \text{sl} \cdot \hat{h}_{kj^*}(x),$
        where sl is the step length (typically $\text{sl} = 0.1$), and update the current estimates for step 4(a):
            $\hat{\eta}_{\theta_k}^{[m-1]} = \hat{\eta}_{\theta_k}^{[m]}.$
```

2.4 Stability selection

Selecting an optimal subset of explanatory variables is a crucial step in almost every supervised data analysis problem. Especially in situations with a large number of covariates, it is often almost impossible to get meaningful results without *automatic*, or at least *semiautomatic*, selection of the most relevant predictors. Selection of covariate subsets based on modified R^2 criteria (e.g., the AIC) can be unstable, see, for example, Flack and Chang (1987), and tend to select too many covariates (see, e.g., Mayr et al. 2012).

Component-wise boosting algorithms are one solution to select predictors in high dimensions and/or $p > n$ problems. As only the best-fitting base-learner is selected to update the model in each boosting step, as discussed above, variable selection can be obtained by stopping the algorithm early enough. Usually, this is done via cross-validation methods, selecting the stopping iteration that optimizes the empirical risk on test data (*predictive* risk). Hence, boosting with *early stopping* via cross-validation offers a way to perform variable selection while fitting the model. Nonetheless, boosted models stopped early via cross-validation still have a tendency to include too many noise variables, particularly in rather low-dimensional settings with few possible predictors and many observations ($n > p$; Bühlmann et al. 2014).

2.4.1 Stability selection for boosted GAM models

To circumvent the problems mentioned above, the *stability selection* approach was developed (Meinshausen and Bühlmann 2010; Shah and Samworth 2013). This generic algorithm can be applied to boosting and all other variable selection methods. The main idea of *stability selection* is to run the selection algorithm on multiple subsamples of the original data. Highly relevant base-learners should be selected in (almost) all subsamples.

Stability selection in combination with boosting was investigated in Hofner et al. (2015) and is outlined in Algorithm 2. In the first step, B random subsets of size $\lfloor n/2 \rfloor$ are drawn, and a boosting model is fitted to each one. The model fit is interrupted as soon as q different base-learners have entered the model. For each base-learner, the selection frequency $\hat{\pi}_j$ is the fraction of subsets in which the base-learner j was selected (7). An effect is included in the model if the selection frequency exceeds the user-specified threshold π_{thr} (8).

This approach leads to upper bounds for the *per-family error rate* (PFER) $\mathbb{E}(V)$, where V is the number of noninformative base-learners wrongly included in the model (i.e., false positives; Meinshausen and Bühlmann 2010):

$$\mathbb{E}(V) \leq \frac{q^2}{(2\pi_{\text{thr}} - 1)p}. \tag{9}$$

Algorithm 2 Stability selection for model-based boosting

```
1. For $b = 1, \ldots, B$:
    (a) Draw a subset of size $\lfloor n/2 \rfloor$ from the data
    (b) Fit a boosting model until the number of selected base-learners
        is equal to $q$ or the number of iterations reaches a pre-specified
        number ($m_{\text{stop}}$).
2. Compute the relative selection frequencies per base-learner:
        $\hat{\pi}_j := \frac{1}{B} \sum_{b=1}^{B} \mathbb{I}_{\{j \in \hat{S}_b\}},$    (7)
    where $\hat{S}_b$ denotes the set of selected base-learners in iteration $b$.
3. Select base-learners with a selection frequency of at least $\pi_{\text{thr}}$,
    which yields a set of stable covariates
        $\hat{S}_{\text{stable}} := \{j : \hat{\pi}_j \geq \pi_{\text{thr}}\}.$    (8)
```

Under certain assumptions, refined, less conservative error bounds can be derived (Shah and Samworth 2013).

One of the main difficulties of stability selection in practice is the choice of the parameters q, π_{thr} and PFER. Even though only two of the three parameters need to be specified (the last one can then be derived under equality in (9)), their choice is not trivial and not always intuitive for the user.

Meinshausen and Bühlmann (2010) state that the threshold should be $\pi_{\text{thr}} \in (0.6, 0.9)$ and has little influence on the result. The number of base-learners q has to be sufficiently large, i.e., q should be at least as big as the number of informative variables in the data (or better to say the number of corresponding base-learners). This is obviously a problem in practical applications, in which the number of informative variables (or base-learners) is usually unknown. One nice property is that if q is fixed, π_{thr} and the PFER can be varied without the need to refit the model. A general advice would thus be to choose q relatively large or to make sure that q is large enough for a given combination of π_{thr} and PFER. Simulation studies like Hofner et al. (2015), Mayr et al. (2016) have shown that the PFER is quite conservative and the true number of false positives will most likely be much smaller than the specified value.

In practical applications, two different approaches to select the parameters are typically used. Both assume that the number of covariates to include, q, is chosen intuitively by the user: The first idea is to look at the calculated inclusion frequencies $\hat{\pi}_j$ and look for a *breakpoint* in the decreasing order of the values. The threshold can be then chosen so that all covariates with inclusion frequencies larger than the breakpoint are included, and the resulting PFER is only used as an additional information. The second possibility is to fix the PFER as a conservative upper bound for the expected number of false-positive base-learners. Hofner et al. (2015) provide some rationales for the selection of the PFER by relating it to common error types, the per-comparison error (i.e., the type I error without multiplicity correction) and the family-wise error rate (i.e., with conservative multiplicity correction).

2.4.2 Stability selection for boosted GAMLSS models

The question of variable selection in (boosted) GAMLSS models is even more critical than in regular (GAM) models, as the question of including a base-learner implies not only if the base-learner should be used in the model at all, but also for which distribution parameter(s) it should be used. Essentially, the number of possible base-learners doubles in a distribution with two parameters, triples in one with three parameters and so on. This is particularly challenging in situations with a large amount of base-learners and in $p > n$ situations.

The method of fitting boosted GAMLSS models in a cyclical way leads to a severe problem when used in combination with stability selection. In each iteration of the algorithm, *all* distribution parameters will receive an additional base-learner as long as their m_{stop} limit is not exceeded. This means that base-learners are added to the model that might have a rather small importance compared to base-learners for other distribution parameters. This becomes especially relevant if the number of informative base-learners varies substantially between distribution parameters.

Regarding the maximum number of base-learners q to be considered in the model, base-learners are counted separately for each distribution parameter, so a base-learner that is selected for the location *and* scale parameter counts as two different base-learners. Arguably, one might circumvent this problem by choosing a higher value for q, but still less stable base-learners could be selected instead of stable ones for other distribution parameters. One aspect of the problem is that the possible model improvement between different distribution parameters is not considered. A careful selection of m_{stop} per distribution parameter might resolve the problem, but the process would still be unstable because the margin of base-learner selection in later stages of the algorithm is quite small. Furthermore, this is not in line with the general approach of stability selection where the standard tuning parameters do not play an important role.

2.5 Noncyclical fitting for boosted GAMLSS

The main idea to solve the previously stated problems of the cyclical fitting approach is to update only one distribution parameter in each iteration, i.e., the distribution parameter with a base-learner that leads to the highest loss reduction over all distribution parameters and base-learners.

Usually, base-learners are selected by comparing their residual sum of squares with respect to the negative gradient vector (*inner loss*). This is done in step (4c) of Algorithm 1 where the different base-learners are compared. However, the residual sum of squares cannot be used to compare the fit of base-learners over different distribution parameters, as the gradients are not comparable.

Inner loss One solution is to compare the empirical risks (i.e., the negative log likelihood of the modeled distribution) after the update with the best-fitting base-learners that have been selected via the residual sum of squares for each distribution parameter: first, for each distribution parameter the best-performing base-learner is selected via the residual sum of squares of the base-learner fit with respect to the gradient vector. Then, the potential improvement in the empirical loss $\Delta\rho$ is compared for all selected base-learners (i.e., over all distribution parameters). Finally, only the best-fitting base-learner (w.r.t. the inner loss) which leads to the highest improvement (w.r.t. the outer loss) is updated. The base-learner selection for each distribution parameter is still done with the *inner loss* (i.e., the residual sum of squares), and this algorithm will be called analogously.

Outer loss Choosing base-learners and parameters with respect to two different optimization criteria may not always lead to the best possible update. A better solution could be to use a criterion which can compare all base-learners for all distribution parameters. As stated, the inner loss cannot be used for such a comparison. However, the empirical loss (i.e., the negative log likelihood of the modeled distribution) can be used to compare both, the base-learners within a distribution parameter and over the different distribution parameters. Now, the negative gradients are used to estimate all base-learners $\hat{h}_{11}, \ldots, \hat{h}_{1p_1}, \hat{h}_{21}, \ldots, \hat{h}_{4p_4}$. The improvement in the empirical risk is then calculated for each base-learner of every distribution parameter, and only the overall best-performing base-learner (w.r.t. the outer loss) is updated. Instead of the using the inner loss, the whole selection process is hence based on the *outer loss* (empirical risk), and the method is named accordingly.

The noncyclical fitting algorithm is shown in Algorithm 3. The *inner* and *outer* variants solely differ in step (3c).

A major advantage of both noncyclical variants compared to the cyclical fitting algorithm (Algorithm 1) is that m_{stop} is always scalar. The updates of each distribution parameter estimate are adaptively chosen. The optimal partitioning (and sequence) of base-learners between different parameters is done automatically while fitting the model. Such a scalar optimization can be done very efficiently using standard cross-validation methods without the need for a multi-dimensional grid search.

3 Simulation study

In a first step, we carry out simulations to evaluate the performance of the new noncyclical fitting algorithms regarding

Algorithm 3 "Noncyclical" component-wise gradient boosting in multiple dimensions

```
Initialize
1. Initialize the additive predictors $\hat{\eta}^{[0]} = (\hat{\eta}_{\theta_1}^{[0]}, \hat{\eta}_{\theta_2}^{[0]}, \hat{\eta}_{\theta_3}^{[0]}, \hat{\eta}_{\theta_4}^{[0]})$ with offset values.
2. For each distribution parameter $\theta_k, k = 1, \dots, 4$, specify a set of base-learners, i.e., for parameter $\theta_k$ define $h_{k1}(\cdot), \dots, h_{kJ_k}(\cdot)$ where $J_k$ is the cardinality of the set of base-learners specified for $\theta_k$.
Boosting in multiple dimensions
For $m = 1$ to $m_{\text{stop}}$:
3. For $k = 1$ to 4:
    (a) Compute negative partial derivatives $-\frac{\partial}{\partial \eta_{\theta_k}} \rho(y, \eta)$ and plug in the current estimates $\hat{\eta}^{[m-1]}(\cdot)$:
        $u_k = \left( \frac{\partial}{\partial \eta_{\theta_k}} \rho(y, \eta) \Big|_{\eta=\hat{\eta}^{[m-1]}(x^{(i)}), y=y^{(i)}} \right)_{i=1,\dots,n}$
    (b) Fit each of the base-learners $u_k$ contained in the set of base-learners specified for the distribution parameter $\theta_k$ in step (2) to the negative gradient vector.
    (c) Select the best-fitting base-learner $h_{kj^*}$ either by
        • the inner loss, i.e., the residual sum of squares of the base-learner fit w.r.t. $u_k$:
            $j^* = \underset{j \in 1,\dots,J_k}{\operatorname{argmin}} \sum_{i=1}^{n} (u_k^{(i)} - \hat{h}_{kj}(x^{(i)}))^2$
        • the outer loss, i.e., the negative log likelihood of the modeled distribution after the potential update:
            $j^* = \underset{j \in 1,\dots,J_k}{\operatorname{argmin}} \sum_{i=1}^{n} \rho \left( y^{(i)}, \hat{\eta}_{\theta_k}^{[m-1]}(x^{(i)}) + \text{sl} \cdot \hat{h}_{kj}(x^{(i)}) \right)$
    (d) Compute the possible improvement of this update regarding the outer loss
        $\Delta\rho_k = \sum_{i=1}^{n} \rho \left( y^{(i)}, \hat{\eta}_{\theta_k}^{[m-1]}(x^{(i)}) + \text{sl} \cdot \hat{h}_{kj^*}(x^{(i)}) \right)$
4. Update, depending on the value of the loss reduction $k^* = \operatorname{argmin}_{k \in 1,\dots,4}(\Delta\rho_k)$ only the overall best-fitting base-learner:
    $\hat{\eta}_{\theta_{k^*}}^{[m]} = \hat{\eta}_{\theta_{k^*}}^{[m-1]} + \text{sl} \cdot \hat{h}_{k^*j^*}(x)$
5. Set $\hat{\eta}_{\theta_k}^{[m]} := \hat{\eta}_{\theta_k}^{[m-1]}$ for all $k \neq k^*$.
```

convergence, convergence speed and runtime. In a second step, we analyze the variable selection properties if the new variant is combined with stability selection.

3.1 Performance of the noncyclical algorithms

The response y_i is drawn from a normal distribution $N(\mu_i, \sigma_i)$, where μ_i and σ_i depend on 4 covariates each. The $x_i, i = 1, \dots, 6$, are drawn independently from a uniform distribution on $[-1, 1]$, i.e., $n = 500$ samples are drawn independently from $U(-1, 1)$. Two covariates x_3 and x_4 are shared between both μ_i and σ_i, i.e., they are informative for both parameters, which means that there are $p_{\text{inf}} = 6$ informative variables overall. The resulting predictors look like

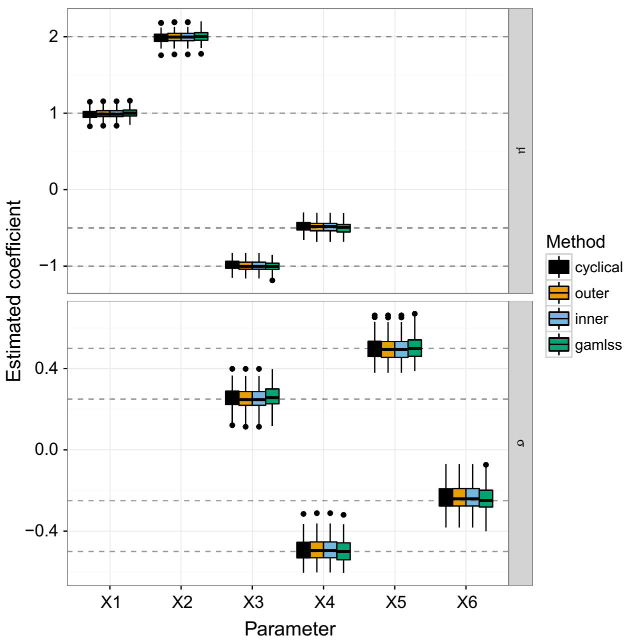

Fig. 1 Distribution of coefficient estimates from $B = 100$ simulation runs. The *dashed lines* show the true parameters. All algorithms were fitted until convergence

$$\mu_i = x_{1i} + 2x_{2i} + 0.5x_{3i} - x_{4i}$$
$$\log(\sigma_i) = 0.5x_{3i} + 0.25x_{4i} - 0.25x_{5i} - 0.5x_{6i}.$$

Convergence First, we compare the new noncyclical boosting algorithms and the cyclical approach with the classical estimation method based on penalized maximum likelihood (as implemented in the R package **gamlss**, Rigby et al. 2008). The results from $B = 100$ simulation runs are shown in Fig. 1. All four methods converge to the correct solution.

Convergence speed Second, we compare the convergence speed in terms of boosting iterations. Therefore, noninformative variables are added to the model. Four settings are considered with $p_{\text{n-inf}} = 0, 50, 250$ and 500 additional noninformative covariates independently sampled from a $U(-1, 1)$ distribution. With $n = 500$ observations, both $p_{\text{n-inf}} = 250$ and $p_{\text{n-inf}} = 500$ are high-dimensional situations ($p > n$) as we have two distribution parameters. In Fig. 2, the mean risk over 100 simulated data sets is plotted against the number of iterations. The m_{stop} value of the cyclical variant shown in Fig. 2 is the sum of the number of updates on every distribution parameter. Outer and inner loss variants of the noncyclical algorithm have exactly the same risk profiles in all four settings. Compared to the cyclical algorithm, the convergence is faster in the first 500 iterations. After more than 500 iterations, the risk reduction is the

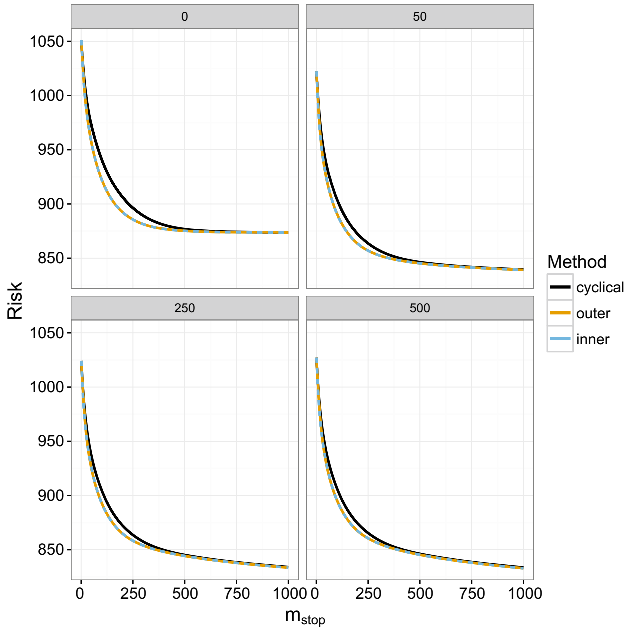

Fig. 2 Convergence speed (regarding the number of boosting iterations m) with 6 informative and $p_{\text{n-inf}} = 0, 50, 250$ and 500 additional noise variables

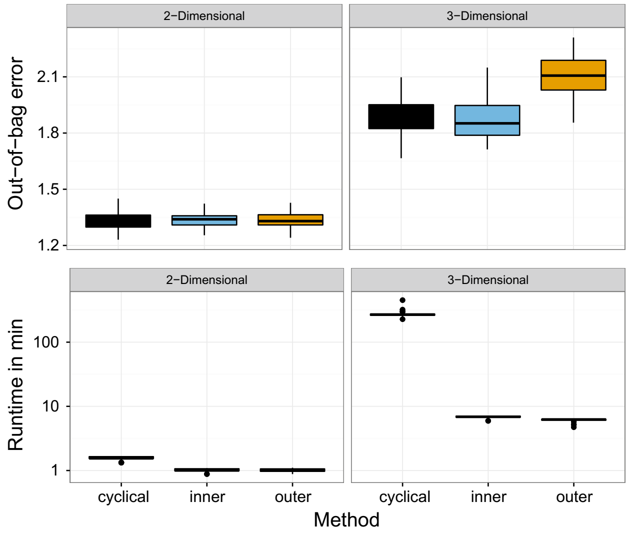

Fig. 3 Out-of-bag error (*top*) and optimization time in minutes (logarithmic scale) Out-of-bag error (*top*) and optimization time in minutes (logarithmic scale; *bottom*) for a two-dimensional (*left*) and three-dimensional distribution (*right*) based on 25-fold bootstrap

same for all three methods. The margin between cyclical and both noncyclical algorithms decreases with a larger number of noise variables.

Runtime The main computational effort of the algorithms is the base-learner selection, which is different for all three methods. The runtime is evaluated in context of cross-validation, which allows us to see how out-of-bag error and runtime behave in different settings. We consider two scenarios—a two-dimensional ($d = 2$) and a three-dimensional ($d = 3$) distribution. The data are generated according to setting 1A and 3A of Sect. 3.2. In each scenario, we sample $n = 500$ observations, but do not add any additional noise variables. For optimization of the model, the out-of-bag prediction error is estimated via a 25-fold bootstrap. A grid of length 10 is created for the cyclical model, with an maximum m_{stop} of 300 for each distribution parameter. The grid is created with the `make.grid` function in **gamboostLSS** (refer to the package documentation for details on the arrangement of the grid points). To allow the same complexity for all variants, the noncyclical methods are allowed up to $m_{\text{stop}} = d \times 300$ iterations.

The results of the benchmark are shown in Fig. 3. The out-of-bag error in the two-dimensional setting is similar for all three methods, but the average number of optimal iterations is considerably smaller for the noncyclical methods (`cyclical`: 360 vs. `inner`: 306, `outer`: 308). In the three-dimensional setting, the outer variant of the noncyclical fitting results in a higher error, whereas the inner variant results in a slightly better performance compared to the cyclical variant. In this setting, the optimal number of iterations is similar for all three methods but near the edge of the searched grid. It is possible that the outer variant will result in a comparable out-of-bag error if the range of the grid is increased.

3.2 Stability selection

After having analyzed the properties of the new noncyclical boosting algorithms for fitting GAMLSS, the remaining question is how they perform when combined with stability selection. In the previous subsection, no differences in the model fit (Fig. 1) and convergence speed (Fig. 2) could be observed, but the optimization results in a three-dimensional setting (Fig. 3) was worse for the outer algorithm. Taking this into consideration, we will only compare the inner and cyclical algorithms here.

We consider three different distributions: (1) the *normal* distribution with two parameters, mean μ_i and standard deviation σ_i. (2) The *negative binomial* distribution with two parameters, mean μ_i and dispersion σ_i. (3) The *zero-inflated negative binomial* (ZINB) distribution with three parameters, μ_i and σ_i identical to the negative binomial distribution, and probability for zero inflation ν_i.

Furthermore, two different partitions of six informative covariates shared between the distribution parameters are evaluated:

(A) *Balanced case* For normal and negative binomial distribution, both μ_i and σ_i depended on four informative covariates, where two are shared. In case of the ZINB distribution, each parameter depends on three informative covariates, each sharing one with the other two parameters.

(B) *Unbalanced case* For normal and negative binomial distribution, μ_i depends on five informative covariates, while σ_i only on one. No informative variables are shared between the two parameters. For the ZINB distribution, μ_i depends on five informative variables, σ_i on two, and ν_i on one. One variable is shared across all three parameters.

To summarize these different scenarios for a total of six informative variables, $x_1, \dots, x_6$:

(1A, 2A)

$$\mu_i = \beta_{1\mu}x_{1i} + \beta_{2\mu}x_{2i} + \beta_{3\mu}x_{3i} + \beta_{4\mu}x_{4i}$$
$$\log(\sigma_i) = \beta_{3\sigma}x_{3i} + \beta_{4\sigma}x_{4i} + \beta_{5\sigma}x_{5i} + \beta_{6\sigma}x_{6i}$$

(1B, 2B)

$$\log(\mu_i) = \beta_{1\mu}x_{1i} + \beta_{2\mu}x_{2i} + \beta_{3\mu}x_{3i}$$
$$+\beta_{4\mu}x_{4i} + \beta_{5\mu}x_{5i}$$
$$\log(\sigma_i) = \beta_{6\sigma}x_{6i}$$

(3A)

$$\log(\mu_i) = \beta_{1\mu}x_{1i} + \beta_{2\mu}x_{2i} + \beta_{3\mu}x_{3i}$$
$$\log(\sigma_i) = \beta_{3\sigma}x_{3i} + \beta_{4\sigma}x_{4i} + \beta_{5\sigma}x_{5i}$$
$$\text{logit}(\nu_i) = \beta_{1\nu}x_{1i} + \beta_{5\nu}x_{5i} + \beta_{6\nu}x_{6i}$$

(3B)

$$\log(\mu_i) = \beta_{1\mu}x_{1i} + \beta_{2\mu}x_{2i} + \beta_{3\mu}x_{3i}$$
$$+\beta_{4\mu}x_{4i} + \beta_{5\mu}x_{5i}$$
$$\log(\sigma_i) = \beta_{5\sigma}x_{5i} + \beta_{6\sigma}x_{6i}$$
$$\text{logit}(\nu_i) = \beta_{6\nu}x_{6i}$$

To evaluate the performance of stability selection, two criteria have to be considered. First, the *true-positive rate*, or the number of *true positives* (TP, number of correctly identified informative variable). Secondly, the *false-positive rate*, or the number of *false positives* (FP, number of noninformative variable that were selected as stable predictors).

Considering stability selection, the most obvious control parameter to influence false- and true-positive rates is the threshold π_{thr}. To evaluate the algorithms depending on the settings of stability selection, we consider several values for the number of variables to be included in the model $q \in \{8, 15, 25, 50\}$ and the threshold π_{thr} (varying between 0.55 and 0.99 in steps of 0.01). A third factor is the number of (noise) variables in the model: We consider $p = 50$, 250 or 500 covariates (including the six informative ones). It should be noted that the actual number of possible base-learners is p times the number of distribution parameters, as each covariate can be included in one or more additive predictors. To visualize the simulation results, the progress of true and false positives is plotted against the threshold π_{thr} for different values of p and q, where true and false positives are aggregated over all distribution parameters. Separate figures for each distribution parameter can be found in the web supplement. The setting $p = 50, q = 50$ is an edge case that would work for some assumptions about the distribution of selection probabilities (Shah and Samworth 2013). Since the practical application of this scenario is doubtful, we will not further examine it here.

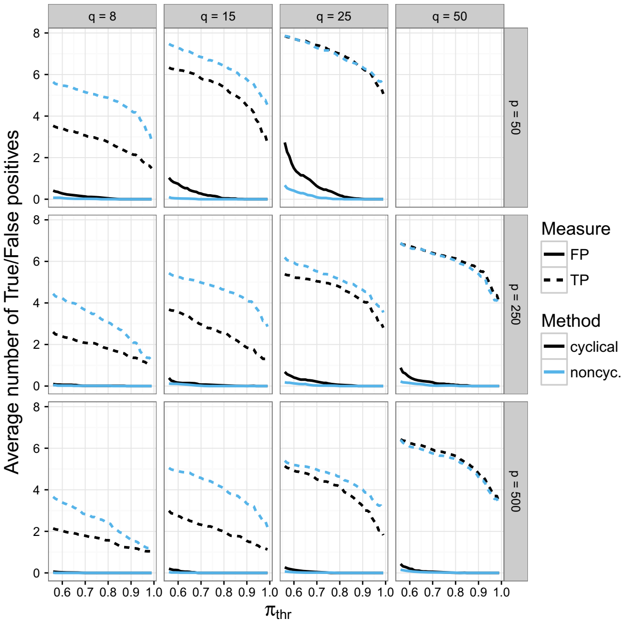

Fig. 4 Balanced case with normal distribution (Scenario 1A)

3.2.1 Results

It can be observed that with increasing threshold π_{thr}, the number of true positives as well as the number of false positives declines in all six scenarios (see Figs. 4, 5, 6, 7, 8, 9) and for every combination of p and q. This is a natural consequence as the threshold is increased, the less variables are selected. Furthermore, the PFER, which is to be controlled by stability selection, decreases with increasing threshold π_{thr} (see Eq. 9).

Results for the normal distribution

In the balanced case (Fig. 4), a higher number of true positives for the noncyclical algorithm can be observed compared to the cyclical algorithm for most simulation settings. Particularly for smaller q values ($q \in \{8, 15\}$), the true-positive rate was always higher compared to the cyclical variant. For

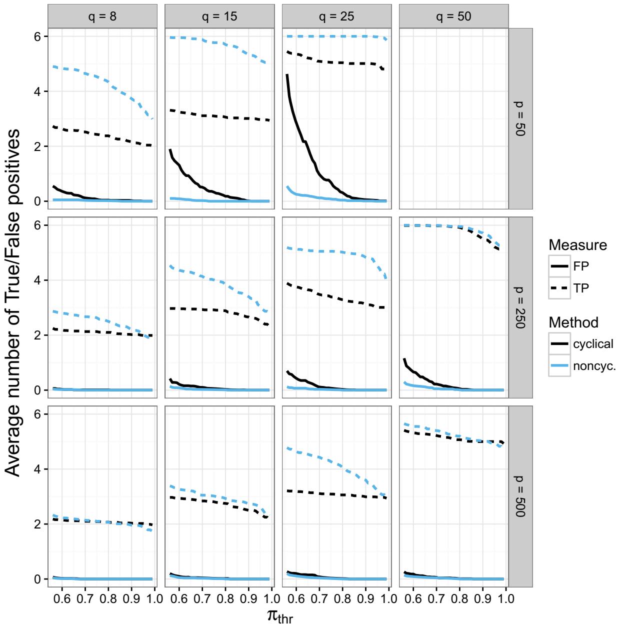

Fig. 5 Unbalanced case with normal distribution (Scenario 1B)

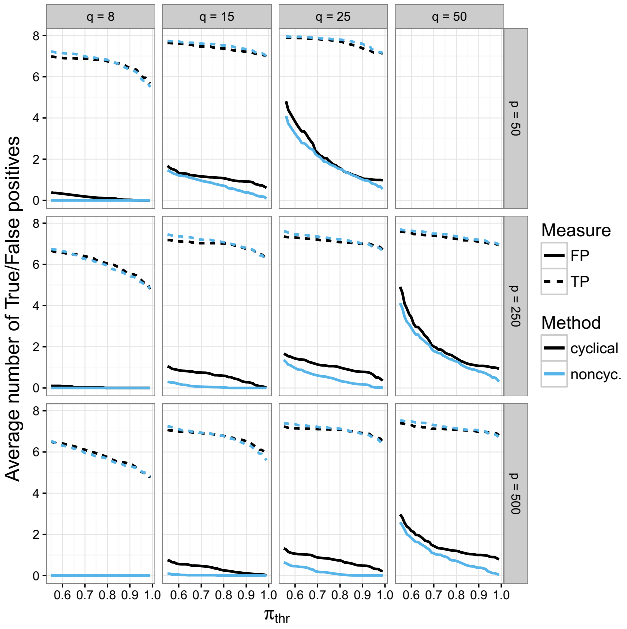

Fig. 6 Balanced case with negative binomial distribution (Scenario 2A)

higher q values, the margin decreases and for the highest settings both methods have approximately the same progression over π_{thr}, with slightly better results for the cyclical algorithm. Overall, the number of true positives increases with a higher value of q. Hofner et al. (2015) found similar results for boosting with one-dimensional prediction functions, but also showed that the true-positive rate decreases again after a certain value of q. This could not be verified for the multi-dimensional case.

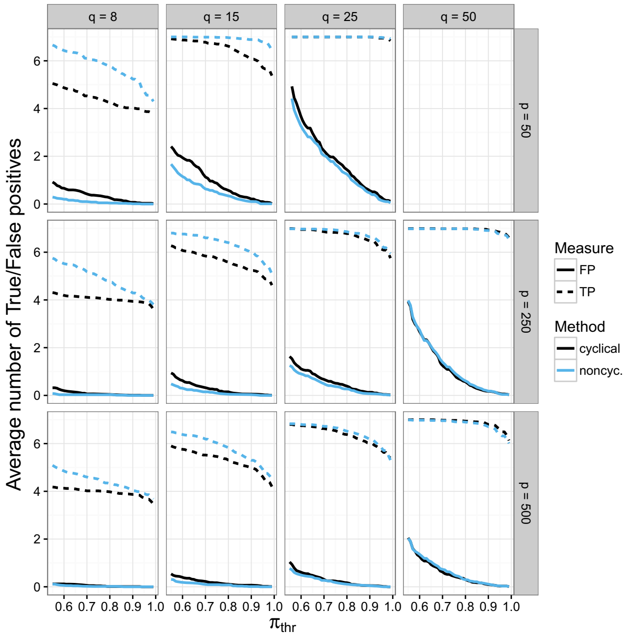

Fig. 7 Unbalanced case with negative binomial distribution (Scenario 2B)

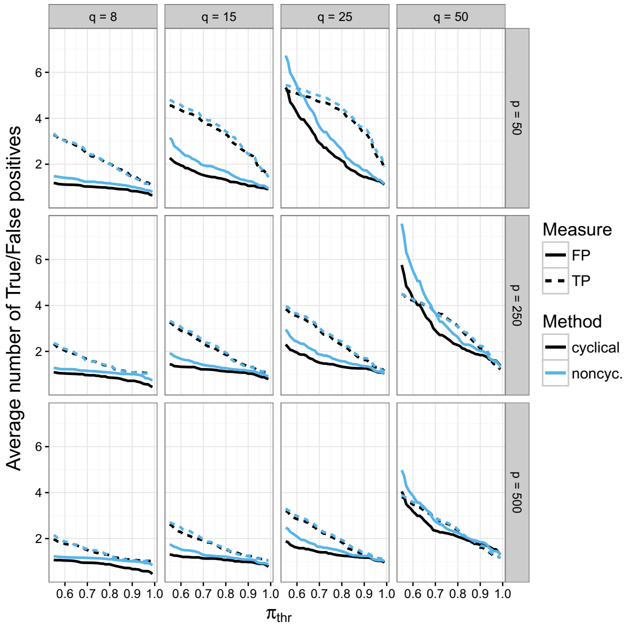

Fig. 8 Balanced case with zero-inflated negative binomial distribution (Scenario 3A)

The false-positive rate is extremely low for both methods, especially in the high-dimensional settings. The noncyclical fitting method has a constantly smaller or identical false-positive rate and the difference reduces for higher π_{thr}, as expected. For all settings, the false-positive rate reaches zero

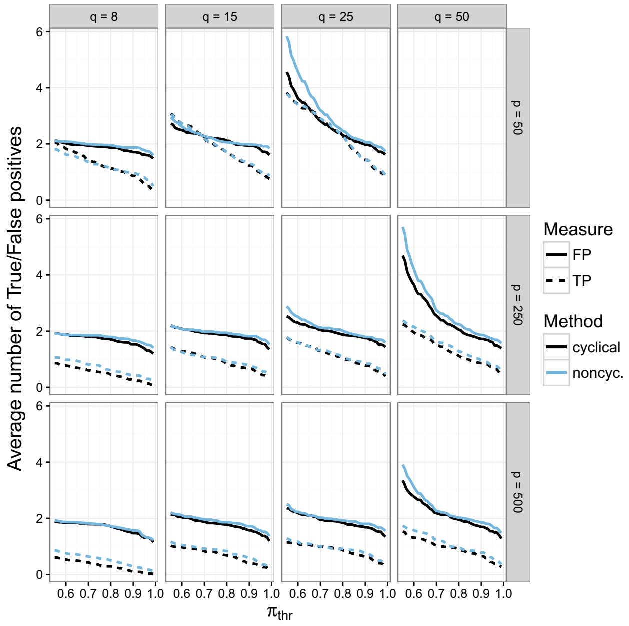

Fig. 9 Unbalanced case with zero-inflated negative binomial distribution (Scenario 3B)

for a threshold higher than 0.9. The setting with the highest false-positive rate is $p = 50$ and $q = 25$, a low-dimensional case with a relatively high threshold. This is also the only setting where on average all 8 informative variables are found (for a threshold of 0.55).

In the unbalanced case (Fig. 5), the results are similar. The number of false positives for the noncyclical variant is lower compared to the cyclical approach in almost all settings. The main difference between the balanced and the unbalanced case is that the number of true positives for the $p = 50, q = 25$ setting is almost identical in the former case, whereas in the latter case the noncyclical variant is dominating the cyclical algorithm. On the other hand, in the high-dimensional case with a small q ($p = 500$, $q = 8$) both fitting methods have about the same true-positive rate for all possible threshold values.

In summary, it can be seen that the novel noncyclical algorithm is generally better, but at least comparable, to the cyclical method in identifying informative variables. Furthermore, the false-positive rate is less or identical to the cyclical method. For some scenarios in which the scale parameter σ_i is higher compared to the location parameter μ_i, the cyclical variant achieves slightly better results than the noncyclical variant regarding true positives at high p and q values.

Results for the negative binomial distribution

In the balanced case of the negative binomial distribution (Fig. 6), the number of true positives is almost identical for the cyclical and noncyclical algorithm in all settings, while the number of true positives is generally quite high. It varies between 6 and 8 in almost all settings, except for the cases with a very small value of q (=8) where it is slightly lower. This is consistent with the results for stability selection with one-dimensional boosting (Hofner et al. 2015; Mayr et al. 2016). The number of false positives in the noncyclical variants is smaller or identical to the cyclical variant in all tested settings.

In the unbalanced case, the true-positive rate of the noncyclical variant is higher compared to the cyclical variant, whereas the difference reduces for larger values of q. The results are consistent with the normal distribution setting but with smaller differences between both methods.

Results for ZINB distribution

The third considered distribution in our simulation setting is the ZINB distribution, which features three parameters to fit.

In Fig. 8, the results for the balanced case (scenario 3A) are visualized. The tendency of a larger number true positives in the noncyclical variant, which could be observed for both two-parametric distributions, is not present here. For all settings, except for high-dimensional settings with a low q (i.e., $p = 250, 500$ and $q = 50$), the cyclical variant has a higher number of true positives. Additionally, the number of false positives is constantly higher for the noncyclical variant. For the unbalanced setting (Fig. 9), the results are similar in true positives and negatives between both methods.

The number of true positives is overall considerably smaller compared to all other simulation settings. Particularly in the high-dimensional cases ($p = 250, 500$), not even half of the informative covariates are found. In settings with smaller q, the number of true positives is lower than two. Both algorithms obtain approximately the same number of true positives for all settings. In cases with a very low or a very high number q (i.e., $q = 8$ or 50), the noncyclical algorithm is slightly better. The number of false positives is very high, especially compared with the number of true positives and particularly for the unbalanced case. For a lot of settings, more than half of the included variables are noninformative. The number of false positives is higher for the noncyclical case. The difference are especially present in settings with a high q and a low π_{thr}, those settings which also have the highest numbers of true positives.

Altogether, the trend from the simulated two-parameter distributions is not present in the three parametric settings. The cyclical algorithm overall is not worse or even better with regard to both true and false positives for almost all tested scenarios.

4 Modeling sea duck abundance

A recent analysis by Smith et al. (2017) investigated the abundance of wintering sea ducks in Nantucket Sound, Massachusetts, USA. Spatiotemporal abundance data for

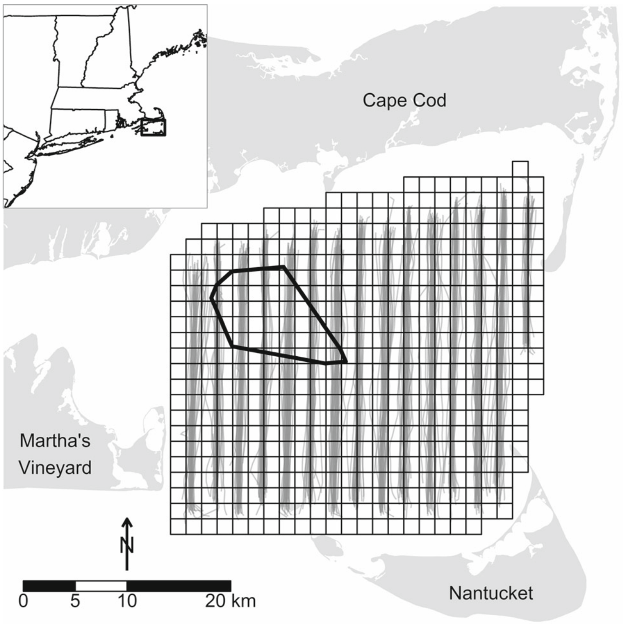

Fig. 10 Nantucket sound—research area of the seabird study by Smith et al. (2017). *Squares* are the discretized segments in which bird abundance was studied. *Gray lines* indicate all aerial transects flown over the course of the study. The *black polygon* indicates the location of permitted wind energy development on Horseshoe Shoal

common eider (among other species) were collected between 2003 and 2005 by counting sea ducks on multiple aerial strip transects from a small plane. For the subsequent analysis, the research area was split in 2.25 km^2 segments (see Fig. 10). Researchers were interested in variables that explained and predicted the distribution of the common eider in the examined area.

As the data were zero-inflated (75% of the segments contained no birds) and highly skewed (a small number of segments contained up to 30,000 birds), a hurdle model (Mullahy 1986) was used for estimation. Therefore, the model was split into an occupancy model (zero part) and an abundance model (count part). The occupancy model estimated if a segment was populated at all and was fitted by boosting a generalized additive model (GAM) with binomial loss, i.e., an additive logistic regression model. In the second step, the number of birds in populated segments was estimated with a boosted GAMLSS model. Because of the skewed and long-tailed data, the (zero-truncated) *negative binomial* distribution was chosen for the abundance model (compare Mullahy 1986).

We reproduce the common eider model reported by Smith et al. (2017) but apply the novel noncyclical algorithm; Smith *et al.* used the cyclic algorithm to fit the GAMLSS model. As discussed in Sect. 3.2, we apply the noncyclical algorithm with inner loss. In short, both distribution parameters, mean *and* overdispersion of the abundance model, and the probability of bird sightings in the occupancy model were regressed on a large number of biophysical covariates, spatial and spatiotemporal effects, and some pre-defined interactions. A complete list of the considered effects can be found in the web supplement. To allow model selection (i.e., the selection between modeling alternatives), the covariate effects were split into linear and nonlinear base-learners (Hothorn et al. 2011; Hofner et al. 2011). The step length was set to sl $= 0.3$, and the optimal number of boosting iterations m_{stop} was found via 25-fold subsampling with sample size $n/2$ (Mayr et al. 2012). Additionally, we used stability selection to obtain sparser models. The numbers of variables to be included per boosting run was set to $q = 35$, and the per-family error rate was set to 6. With unimodality assumption, this resulted in a threshold of $\pi_{\text{thr}} = 0.9$. These settings were chosen identically to the original choices in Smith et al. (2017).

4.1 Results

Subsampling yielded an optimal m_{stop} of 2231, split in $m_{\text{stop},\mu} = 1871$ and $m_{\text{stop},\sigma} = 336$. The resulting model selected 46 out of 48 possible covariates in μ and 8 out of 48 in σ, which is far too complex of a model (especially in μ) to be useful.

With stability selection (see Fig. 12), 10 effects were selected for the location: the intercept, relative sea surface temperature (smooth), chlorophyll a levels (smooth), chromophoric dissolved organic material levels (smooth), sea floor sediment grain size (linear and smooth), sea floor surface area (smooth), mean epibenthic tidal velocity (smooth), a smooth spatial interaction, the presence of nearby ferry routes (yes/no) and two factors to account for changes in 2004 and 2005 compared to the year 2003. For the overdispersion parameter, 5 effects were selected: sea surface temperature (linear), bathymetry (linear), the mean (smooth) and standard deviation (linear) of the epibenthic tidal velocity, and the linear spatial interaction. For the location, all metric variables entered the model nonlinearly. Only sediment grain size was selected linearly as well as nonlinearly in the model. The converse was true for the overdispersion parameter: Only the mean epibenthic velocity was selected as a smooth effect, and all others were selected as linear effects. In Fig. 11, the spatial effects for the mean and overdispersion can be seen. The segment size is based on the spatial covariate with the coarsest resolution, so different resolutions of the segments and their stability were not explored further, though it is likely that this will have some influence on the model performance and the results in Fig. 11.

4.2 Comparison to results of the cyclic method

Comparing the model with the results of Smith et al. (2017), the noncyclical model was larger in μ (10 effects, compared

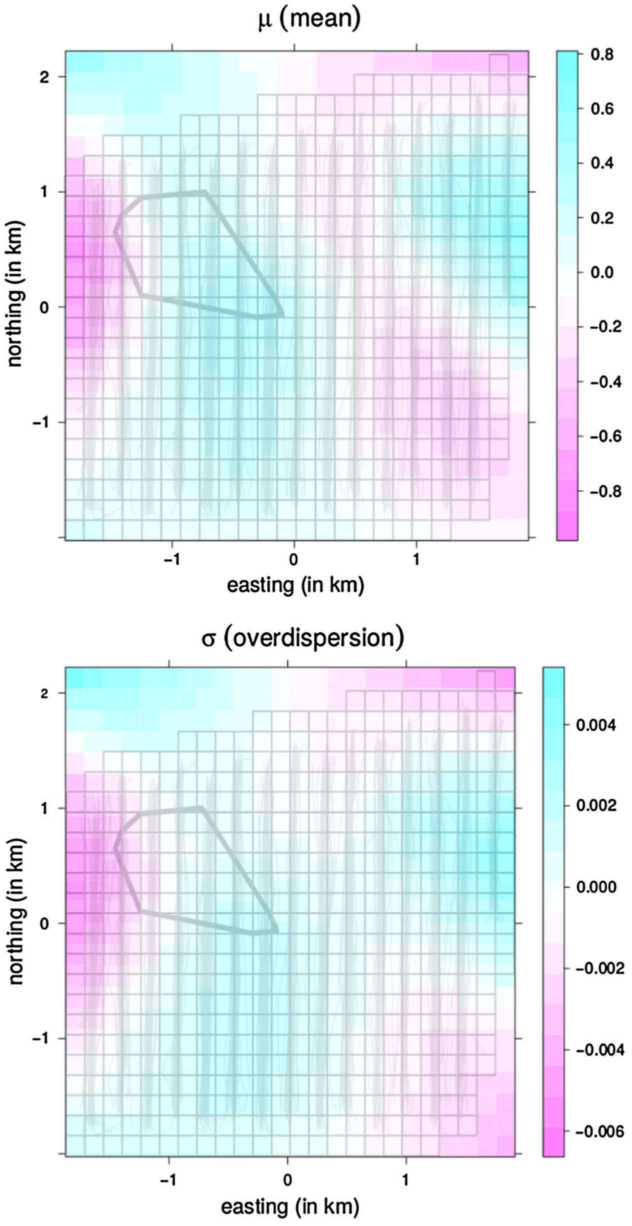

Fig. 11 Spatial effects for mean (*upper figure*) and overdispersion (*lower figure*) of seabird population. The *shaded areas* in the both figures show the research area

to 8 effects), but smaller in σ (5 effects, compared to 7 effects). Chlorophyll a levels, mean epibenthic tidal velocity, smooth spatial variation and year were not selected for the mean by stability selection with the cyclical fitting algorithm. On the other hand, bathymetry was selected by the cyclical fitting method, but not by the noncyclical. For the overdispersion parameter, the cyclical algorithm selected the year and the northing of a segment (the north–south position of a segment relative to the median) in addition to all effects selected by the noncyclical variant. Most effects were selected by both the cyclical and the noncyclical algorithm, and the differences in the selected effects were rather small.

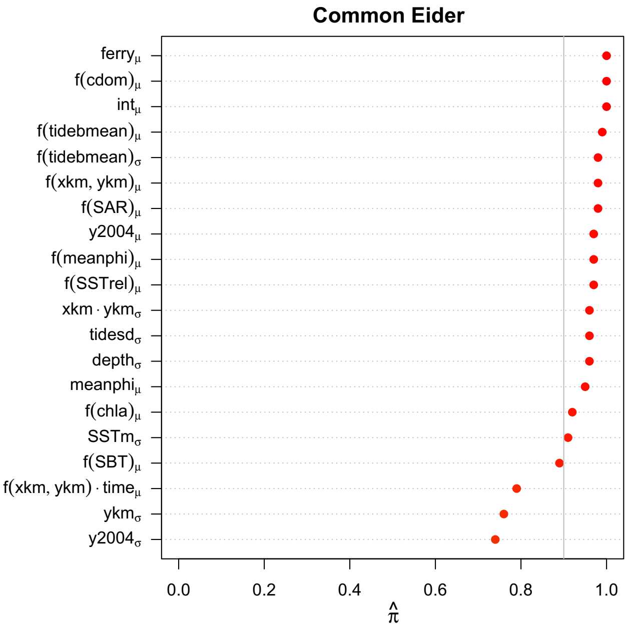

Fig. 12 Selection frequencies of the 20 most frequently selected biophysical covariate base-learners of common eider abundance, determined by stability selection with $q = 35$ and PFER $= 6$. The *gray line* represents the corresponding threshold of 0.9

In the simulation study for the negative binomial distribution (Sect. 3), the noncyclical variant had a smaller false-positive rate and a higher true-positive rate. Even though the simulation was simplified compared to this application (only linear effects, known true number of informative covariates, uncorrelated effects), the results suggest to prefer the noncyclical variant. Nonetheless, the interpretation of selected covariate effects and final model assessment rests ultimately with subject matter experts.

5 Conclusion

The main contribution of this paper is a statistical model building algorithm that combines the three approaches of gradient boosting, GAMLSS and stability selection. As shown in our simulation studies and the application on sea duck abundance in Sect. 4, the proposed algorithm incorporates the flexibility of structured additive regression modeling via GAMLSS, while it simultaneously allows for a data-driven generation of sparse models.

Being based on the gamboostLSS framework by Mayr et al. (2012), the main feature of the new algorithm is a new "noncyclical" fitting method for boosted GAMLSS models. As shown in the simulation studies, this method does not only increase the flexibility of the variable selection mechanism used in gamboostLSS, but is also more time efficient than the traditional cyclical fitting algorithm. In fact, even though the initial runtime to fit a single model may be higher (especially if the base-learner selection is done via the outer loss approach), this time is regained while finding the optimal number of boosting iterations via cross-validation approaches. Furthermore, the convergence speed of the new algorithm proved to be faster, and consequently, fewer boosting iterations were needed in total.

Regarding stability selection, we observed that the noncyclical algorithm often had fewer false positives as well as more true positives compared to the cyclical variant in the two-parameter distribution tested in our simulation study. For high-dimensional cases, however, the differences between both methods reduced and, especially with regard to the number of true positives, approximately equal results were achieved. For three-parameter distribution, the cyclical variant achieved better values throughout with respect to both true- and false-positive rates. This may be due to the fact that for more complex distributions, similar densities can be achieved with different parameter settings. For example, in a zero-inflated negative binomial setting, a small location may be hard to distinguish from a large zero inflation. Obviously, the behavior of the cyclical variant is more robust in these situations than the noncyclical variant, which tends to fit very different models on each subsample and consequently selects a higher amount of noninformative variables.

In summary, we have developed a framework for model building in GAMLSS that simplifies traditional optimization approaches to a great extent. For practitioners and applied statisticians, the main consequence of the new methodology is the incorporation of fewer noise variables in the GAMLSS model, leading to sparser and thus more interpretable models. Furthermore, the tuning of the new algorithm is far more efficient and leads to much shorter run times, particularly for complex distributions.

6 Implementation

The derived fitting methods for `gamboostLSS` models are implemented in the R add-on package **gamboostLSS** (Hofner et al. 2017). The fitting algorithm can be specified via the `method` argument. By default, `method` is set to `"cyclical"` which is the originally proposed algorithm. The new inner variant of the noncyclical fitting can be selected with `method = "noncyclic"`. Based on the results of our simulation study, we decided to only support the inner variant in the final package. To ensure reproducibility of the experiments, the state of the package with both inner and outer variants is kept in a separate github branch for this publication, which can be found at http://www.github.com/boost-R/gamboostLSS/tree/stco_paper.

Base-learners and some of the basic methods are implemented in the R package **mboost** (Hothorn et al. 2010; Hofner et al. 2014; Hothorn et al. 2017). The basic fitting algorithm for each distribution parameter is also implemented in **mboost**. For a tutorial and an explanation of technical details of **gamboostLSS**, see Hofner et al. (2016). Stability selection is implemented in the R package **stabs** (Hofner and Hothorn 2017; Hofner et al. 2015), with a specialized function for `gamboostLSS` models, which is included in **gamboostLSS** itself. The development of `mboost`, `gamboostLSS` and `stabs` is hosted openly at

http://www.github.com/boost-R/mboost
http://www.github.com/boost-R/gamboostLSS
http://www.github.com/hofnerb/stabs.

Bug reports and requests should be made there. All packages are also available for installation directly from CRAN.

Acknowledgements We thank Mass Audubon for the use of common eider abundance data.

References

Aho, K., Derryberry, D.W., Peterson, T.: Model selection for ecologists: the worldviews of AIC and BIC. Ecology **95**, 631–636 (2014)

Anderson, D.R., Burnham, K.P.: Avoiding pitfalls when using information-theoretic methods. J. Wildl. Manag. 912–918 (2002)

Bühlmann, P., Hothorn, T.: Boosting algorithms: regularization, prediction and model fitting. Stat. Sci. **22**, 477–505 (2007)

Bühlmann, P., Hothorn, T.: Twin boosting: improved feature selection and prediction. Stat. Comput. **20**, 119–138 (2010)

Bühlmann, P., Yu, B.: Boosting with the L_2 loss: regression and classification. J. Am. Stat. Assoc. **98**, 324–339 (2003)

Bühlmann, P., Yu, B.: Sparse boosting. J. Mach. Learn. Res. **7**, 1001–1024 (2006)

Bühlmann, P., Gertheiss, J., Hieke, S., Kneib, T., Ma, S., Schumacher, M., Tutz, G., Wang, C., Wang, Z., Ziegler, A., et al.: Discussion of "the evolution of boosting algorithms" and "extending statistical boosting". Methods Inf. Med. **53**(6), 436–445 (2014)

Dormann, C.F., Elith, J., Bacher, S., Buchmann, C., Carl, G., Carre, G., Marquez, J.R.G., Gruber, B., Lafourcade, B., Leitao, P.J., Münkemüller, T., McClean, C., Osborne, P.E., Reineking, B., Schröder, B., Skidmore, A.K., Zurell, D., Lautenbach, S.: Collinearity: A review of methods to deal with it and a simulation study evaluating their performance. Ecography **36**, 27–46 (2013)

Flack, V.F., Chang, P.C.: Frequency of selecting noise variables in subset regression analysis: a simulation study. Am. Stat. **41**(1), 84–86 (1987)

Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). Ann. Stat. **28**(2), 337–407 (2000)

Hastie, T.J., Tibshirani, R.J.: Generalized Additive Models, vol. 43. CRC Press, Boca Raton (1990)

Hofner, B., Boccuto, L., Göker, M.: Controlling false discoveries in high-dimensional situations: boosting with stability selection. BMC Bioinf. **16**(1), 144 (2015)

Hofner, B., Hothorn, T.: stabs: stability selection with error control (2017). http://CRAN.R-project.org/package=stabs. R package version 0.6-2

Hofner, B., Hothorn, T., Kneib, T., Schmid, M.: A framework for unbiased model selection based on boosting. J. Comput. Gr. Stat. **20**, 956–971 (2011)

Hofner, B., Mayr, A., Fenske, N., Thomas, J., Schmid, M.: gamboostLSS: boosting methods for GAMLSS models (2017). http://CRAN.R-project.org/package=gamboostLSS. R package version 2.0-0

Hofner, B., Mayr, A., Robinzonov, N., Schmid, M.: Model-based boosting in R—A hands-on tutorial using the R package mboost. Comput. Stat. **29**, 3–35 (2014)

Hofner, B., Mayr, A., Schmid, M.: gamboostLSS: an R package for model building and variable selection in the GAMLSS framework. J. Stat. Softw. **74**(1), 1–31 (2016)

Hothorn, T., Buehlmann, P., Kneib, T., Schmid, M., Hofner, B.: Model-based boosting 2.0. J. Mach. Learn. Res. **11**, 2109–2113 (2010)

Hothorn, T., Buehlmann, P., Kneib, T., Schmid, T., Hofner, B.: mboost: model-based boosting (2017). http://CRAN.R-project.org/package=mboost. R package version 2.8-0

Hothorn, T., Müller, J., Schröder, B., Kneib, T., Brandl, R.: Decomposing environmental, spatial, and spatiotemporal components of species distributions. Ecol. Monogr. **81**, 329–347 (2011)

Huang, S.M.Y., Huang, J., Fang, K.: Gene network-based cancer prognosis analysis with sparse boosting. Genet. Res. **94**, 205–221 (2012)

Li, P.: Robust logitboost and adaptive base class (abc) logitboost (2012). arXiv preprint arXiv:1203.3491

Mayr, A., Binder, H., Gefeller, O., Schmid, M., et al.: The evolution of boosting algorithms. Methods Inf. Med. **53**(6), 419–427 (2014)

Mayr, A., Binder, H., Gefeller, O., Schmid, M., et al.: Extending statistical boosting. Methods Inf. Med. **53**(6), 428–435 (2014)

Mayr, A., Fenske, N., Hofner, B., Kneib, T., Schmid, M.: Generalized additive models for location, scale and shape for high-dimensional data—a flexible approach based on boosting. J. R. Stat. Soc. Ser. C Appl. Stat. **61**(3), 403–427 (2012)

Mayr, A., Hofner, B., Schmid, M.: Boosting the discriminatory power of sparse survival models via optimization of the concordance index and stability selection. BMC Bioinf. **17**(1), 288 (2016)

Mayr, A., Hofner, B., Schmid, M., et al.: The importance of knowing when to stop. Methods Inf. Med. **51**(2), 178–186 (2012)

Meinshausen, N., Bühlmann, P.: Stability selection. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **72**(4), 417–473 (2010)

Messner, J.W., Mayr, G.J., Zeileis, A.: Nonhomogeneous boosting for predictor selection in ensemble postprocessing. Mon. Weather Rev. **145**(1), 137–147 (2017). doi:10.1175/MWR-D-16-0088.1

Mullahy, J.: Specification and testing of some modified count data models. J. Econom. **33**(3), 341–365 (1986)

Murtaugh, P.A.: Performance of several variable-selection methods applied to real ecological data. Ecol. Lett. **12**, 1061–1068 (2009)

Opelt, A., Fussenegger, M., Pinz, A., Auer, P.: Weak hypotheses and boosting for generic object detection and recognition. In: European Conference on Computer Vision, pp. 71–84. Springer (2004)

Osorio, J.D.G., Galiano, S.G.G.: Non-stationary analysis of dry spells in monsoon season of Senegal River Basin using data from regional climate models (RCMs). J. Hydrol. **450–451**, 82–92 (2012)

Rigby, R.A., Stasinopoulos, D.M.: Generalized additive models for location, scale and shape. J. R. Stat. Soc. Ser. C (Appl. Stat.) **54**(3), 507–554 (2005)

Rigby, R.A., Stasinopoulos, D.M., Akantziliotou, C.: Instructions on how to use the gamlss package in R (2008). http://www.gamlss.org/wp-content/uploads/2013/01/gamlss-manual.pdf

Schmid, M., Hothorn, T.: Boosting additive models using componentwise P-splines. Comput. Stat. Data Anal. **53**(2), 298–311 (2008)

Schmid, M., Potapov, S., Pfahlberg, A., Hothorn, T.: Estimation and regularization techniques for regression models with multidimensional prediction functions. Stat. Comput. **20**(2), 139–150 (2010)

Shah, R.D., Samworth, R.J.: Variable selection with error control: Another look at stability selection. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **75**(1), 55–80 (2013)

Smith, A.D., Hofner, B., Osenkowski, J.E., Allison, T., Sadoti, G., McWilliams, S.R., Paton, P.W.C.: Spatiotemporal modelling of sea duck abundance: implications for marine spatial planning (2017). arXiv preprint arXiv:1705.00644