A tree-based varying coefficient model

Henning Zakrisson*† Mathias Lindholm*

January 17, 2024

Abstract

The paper introduces a tree-based varying coefficient model (VCM) where the varying coefficients are modelled using the cyclic gradient boosting machine (CGBM) from Delong et al. (2023). Modelling the coefficient functions using a CGBM allows for dimension-wise early stopping and feature importance scores. The dimension-wise early stopping not only reduces the risk of dimension-specific overfitting, but also reveals differences in model complexity across dimensions. The use of feature importance scores allows for simple feature selection and easy model interpretation. The model is evaluated on the same simulated and real data examples as those used in Richman and Wüthrich (2023), and the results show that it produces results in terms of out of sample loss that are comparable to those of their neural network-based VCM called LocalGLMnet.

Keywords: *Generalised linear models, Multivariate gradient boosting, Feature selection, Interaction effects, Early stopping*

1 Introduction

Many machine learning models that yield accurate results and high predictive power struggle with another important factor – interpretability. This is especially a problem for deep learning models, see e.g. Marcinkevičs and Vogt (2023). One way to address this is to use a model structure that is inherently interpretable, such as a generalised linear model (GLM). However, without manual feature engineering, GLMs can not handle non-linear effects and interactions between features. A more flexible model family that retains some of the interpretability of GLMs are varying-coefficient models (VCMs), introduced in Hastie and Tibshirani (1993). In a VCM, the regression coefficients are replaced by feature-dependent regression coefficient functions, which can be of any, possibly non-linear, functional form. A recent example of a VCM is the LocalGLMnet model introduced in Richman and Wüthrich (2023), where the regression coefficient functions are modeled using a feed-forward neural network (FFNN). This creates a *locally* interpretable machine learning model, as those discussed in e.g. (Marcinkevičs and Vogt, 2023, Section 4.2.6). The LocalGLMnet shows promising accuracy on both simulated and real-world data, while also allowing for more insightful feature selection and local interpretability. Other examples of VCMs combined with neural networks are e.g. Alvarez Melis and Jaakkola (2018); Al-Shedivat et al. (2020); Thompson et al. (2023), and these are sometimes referred to as "contextual" models.

As an alternative to using feed-forward neural network (FFNN)-based models for the coefficient functions, tree-based models are generally easier to interpret, and they are often easier to tune and faster to train. In Wang and Hastie (2014), this idea is introduced, focusing on linear models and gradient boosting machines (GBMs) using vector-valued trees, i.e. trees

*Department of Mathematics, Stockholm University
†Corresponding author: zakrisson@math.su.se

with response of the same dimension as the vector of predictive features. Although the focus is on linear models, generalised linear models are also discussed. This ansatz, however, differs from applying a multidimensional GBM directly to the loss function induced by the VCM formulation. On the other hand, the original GBM algorithm (Friedman, 2001) and most of its successors such as XGBoost (Chen et al., 2015) and LightGBM (Ke et al., 2017) provide *univariate* estimators, meaning that the output of the models is one-dimensional. While these models are useful for estimating e.g. the mean parameter of a distribution, these models do not allow for fitting a local GLM structure with coefficient functions depending on two or more features. Recently, a number of *multivariate* GBM models have been proposed, such as Delta Boosting (Lee and Lin, 2018), NGBoost (Duan et al., 2020) and Cyclic Gradient Boosting (CGBM) (Delong et al., 2023). In Zhou and Hooker (2022), a GBM-based VCM is introduced where univariate GBMs are fitted to the partial derivatives simultaneously in each boosting iteration.

The current paper introduces a CGBM-based VCM. The main reason for using cyclic GBMs instead of the simultaneous updates using univariate GBMs as in Zhou and Hooker (2022) is that this allows for dimension-wise early stopping, something not covered in Wang and Hastie (2014); Zhou and Hooker (2022). By using dimension-wise early stopping, it is possible to reduce the risk of dimension-wise overfitting of the coefficient functions. Further, this also provides additional insights into differences in model complexity across dimensions. Moreover, the cyclic updates allow for the use of dimension-wise feature importance scores. These feature importance scores can be used both for model interpretation and removing redundant features from the model, reducing model complexity.

The main contribution of this paper is to use *cyclic* GBMs as a way to model the varying coefficient functions in a VCM. The framework is also closely related to the LocalGLM model from Richman and Wüthrich (2023), and the cyclically boosted VCM retrieves a tree-based version of the LocalGLMnet model as a special case. The algorithm focuses on interpretability and ease of use – by focusing the modeling effort on the coefficient functions and not allowing for structure in the intercept term β_0. Also, the CGBM algorithm allows for dimension-wise early stopping, which can be used to reduce the risk of overfitting, something that is not possible in the LocalGLMnet model, nor the boosted VCM from Zhou and Hooker (2022). The dimension-wise feature importance score calculations of the CGBM algorithm also allow for easier model tuning and interpretation of interactions between features.

The remainder of the paper is organised as follows: Section 2 introduces the VCM structure and the CGBM used to construct a cyclically boosted tree VCM. Section 3 discusses modeling considerations when training and using this model. Section 4 presents examples of using cyclically boosted VCMs on simulated and real-world data, using the same data sets as in Richman and Wüthrich (2023). The paper ends with concluding remarks in Section 5.

2 Model architecture

Before going into varying coefficient models and local generalised linear models, we briefly introduce the concept of generalised linear models.

2.1 Generalised linear models

Let $Y \in \mathbb{R}$ denote the response, e.g. number of claims or claim cost, and let $\boldsymbol{X} \in \mathbb{X} \subseteq \mathbb{R}^p$ be a feature vector. If we consider exponential dispersion family (EDF) models, see e.g. Jørgensen (1997) and generalised linear models (GLMs), see e.g. Nelder and Wedderburn (1972), we assume that

$$\mathrm{E}[Y \mid \boldsymbol{X}=\boldsymbol{x}] = \mu(\boldsymbol{x};\beta_0,\boldsymbol{\beta}) := u^{-1}\left(\beta_0+\boldsymbol{\beta}^\top\boldsymbol{x}\right), \tag{1}$$

where $\boldsymbol{\beta}\in\mathbb{R}^p$ and u is the so-called link function. When using EDF models there may be an additional dispersion parameter φ that is possible to neglect when estimating μ or $\boldsymbol{\beta}$. That is, neither the so-called deviance function nor the log-likelihood function w.r.t. μ of an EDF will depend on dispersion parameters. Moreover, in many actuarial applications, it is common to include weights, W, corresponding to e.g. policy duration. In order to ease the exposition, any potential dependence on weights will be suppressed until the numerical illustrations in Section 4. For a longer discussion of duration effects, see Lindholm et al. (2023) and Lindholm and Nazar (2023).

When it comes to estimation of model parameters in a GLM, given an independent sample $(\boldsymbol{x}_i,y_i)_{i=1}^n$, this can be expressed as

$$(\widehat{\beta}_0,\widehat{\boldsymbol{\beta}}) = \underset{(\beta_0,\boldsymbol{\beta})\in\mathbb{R}^{p+1}}{\arg\min}\left\{\sum_{i=1}^n \mathcal{L}\left(\mu(\boldsymbol{x}_i;\beta_0,\boldsymbol{\beta});y_i\right)\right\}, \tag{2}$$

where $\mathcal{L}(\mu;y)$ corresponds to either the negative log-likelihood or the deviance function for an observation y given mean parameter μ (Sundberg, 2019). Further, as seen from (2) the general optimisation problem does not explicitly depend on the link-transformed linear function from (1). Continuing, (2) can be solved using gradient descent where the partial derivatives

$$\begin{aligned} g_{ij} &:= \frac{\partial}{\partial\beta_j}\mathcal{L}(\mu(\boldsymbol{x}_i;\beta_0,\boldsymbol{\beta});y_i) \\ &= \frac{\partial}{\partial\mu}\mathcal{L}(\mu;y_i)\bigg|_{\mu=\mu(\boldsymbol{x}_i;\beta_0,\boldsymbol{\beta})}\cdot\frac{\partial}{\partial\beta_j}\mu(\boldsymbol{x}_i;\beta_0,\boldsymbol{\beta}) \\ &= x_{ij}\frac{\partial}{\partial\mu}\mathcal{L}(\mu;y_i)\bigg|_{\mu=\mu(\boldsymbol{x}_i;\beta_0,\boldsymbol{\beta})}\cdot\frac{\partial}{\partial v}u^{-1}(v)\bigg|_{v=\beta_0+\boldsymbol{\beta}^\top\boldsymbol{x}_i} \end{aligned} \tag{3}$$

appear. Note that (3) can be considered as a directional partial derivative. Further, note that if we use a log-link, i.e.

$$\mu(\boldsymbol{x};\beta_0,\boldsymbol{\beta}) := \exp\left\{\beta_0+\boldsymbol{\beta}^\top\boldsymbol{x}\right\}, \tag{4}$$

then (3) reduces to

$$g_{ij} := x_{ij}\mu(\boldsymbol{x}_i;\beta_0,\boldsymbol{\beta})\frac{\partial}{\partial\mu}\mathcal{L}(\mu;y_i)\bigg|_{\mu=\mu(\boldsymbol{x}_i;\beta_0,\boldsymbol{\beta})}.$$

Remark 1 *Note that the loss function $\mathcal{L}(\mu;y)$ has a parameter dimension of 1, given by $\mu\in\mathbb{R}$, whereas the parametrisation of the mean function $\mu(\boldsymbol{x};\beta_0,\boldsymbol{\beta})$ has $p+1$ dimensions since $\boldsymbol{\beta}\in\mathbb{R}^p$. Further, for a fixed observation i, the partial derivatives in* (3) *for different dimensions j will be dependent.*

2.2 VCMs and local GLMs

The link-transformed linear relationship between the features in a GLM provides models that are interpretable and numerically stable. The GLMs, however, are in general not able to compete with machine learning models such as gradient boosting machines (GBMs) and feed-forward neural networks (FFNNs) in terms of predictive accuracy. This is often the case even if extensive manual feature engineering has been conducted.

The varying-coefficient model (VCM) framework, introduced in Hastie and Tibshirani (1993), suggests that the regression coefficients β_0, $\boldsymbol{\beta}$ in the linear predictor of a GLM (see (1)) are replaced by feature dependent regression coefficient functions $\beta_0(\boldsymbol{z})$, $\boldsymbol{\beta}(\boldsymbol{z})$, where $\boldsymbol{z} \in \mathbb{Z}$. In Richman and Wüthrich (2023) it is suggested to replace the regression coefficients, $\boldsymbol{\beta}$, in (1) to regression coefficient functions $\boldsymbol{\beta}(\boldsymbol{x})$, whereas β_0 is kept constant. To generealize these propositions, we suggest to replace the regression coefficients, $\boldsymbol{\beta}$, in (1) to regression coefficient functions $\boldsymbol{\beta}(\boldsymbol{z})$, where $\boldsymbol{z} \in \mathbb{Z}$, whereas β_0 is kept constant. This yields the mean function

$$\mu(\boldsymbol{x}; \beta_0, \boldsymbol{\beta}(\boldsymbol{z})) := u^{-1}(\beta_0 + \boldsymbol{\beta}(\boldsymbol{z})^\top \boldsymbol{x}). \tag{5}$$

Note that it may be the case that $\mathbb{Z} \not\subset \mathbb{X}$. For example, $\mathbb{Z}$ may include time as a feature, whereas $\mathbb{X}$ does not.

Note that keeping β_0 as a constant intercept, independent of $\boldsymbol{z}$, is not the case in a standard VCM, but chosen here in order for as much of the functional structure as possible to be captured by the coefficient functions $\boldsymbol{\beta}(\boldsymbol{z})$. By analysing $\beta_j(\boldsymbol{z})x_j$, this also allows us to assess potential interactions between x_j and $\boldsymbol{z}$. If $\mathbb{X}_j \subset \mathbb{Z}$, where $\mathbb{X}_j$ is the jth feature space, then $\beta_j(\boldsymbol{z})$ will reveal non-linear dependence on $\boldsymbol{x}$. In Appendix A estimation of regression coefficient functions is discussed with a focus on the L^2 situation and population-based quantities. It is there seen that the jth regression coefficient function can be expressed as a certain expected value conditional on $\boldsymbol{Z}_j$. It is also seen that if the true mean function is of the form of a VCM, then the regression coefficient function estimators will retrieve, possibly noisy, versions of the corresponding true regression coefficient functions. In particular, if in addition the true regression coefficient functions are constants, these constants will be estimated correctly, as should be the case.

For more on interpretations and consequences of using (5), see Remark 2.3 in Richman and Wüthrich (2023), and, more generally, see Section 2 in Hastie and Tibshirani (1993). We also elaborate on these topics below in the numerical illustrations in Section 4.

Continuing, in Richman and Wüthrich (2023) the coefficient functions are modeled using feed-forward NNs. In the present paper, we instead model the coefficient functions using GBMs, but as commented on in Remark 1 the loss w.r.t. $\boldsymbol{\beta}$ has a parameter dimension of $p+1$. Thus, to implement a GBM-based VCM in accordance with (5) calls for using a multi-dimensional GBM applied to the directional derivatives from (3). One way of doing this is to use the cyclic GBM (CGBM) from Delong et al. (2023).

2.2.1 The Cyclic Gradient Boosting Machine

The cyclic gradient boosting machine (CGBM) is introduced in Delong et al. (2023), and is concerned with estimating an unknown d-dimensional parameter function $\boldsymbol{\theta} : \mathbb{X} \to \mathbb{R}^d$ using regression tress. The CGBM, in its basic formulation, is a multi-parametric generalisation of the standard GBM from Friedman (2001). The algorithm is presented in Algorithm 1. For further details, see Delong et al. (2023).

Algorithm 1 Cyclic Gradient Boosting Machine

Let:

- $\mathcal{L}(\boldsymbol{\theta}; y)$ be a loss function,
- $(y_i, \boldsymbol{x}_i)_{i=1}^n$ be independent observations,
- $\mathcal{A}$ be a disjoint partition of $\mathbb{X}$ into $|\mathcal{A}|$ regions,
- $\gamma_l \in \mathbb{R}$ be terminal node values, $l = 1, \dots, |\mathcal{A}|$,
- ϵ_j and κ_j be shrinkages and number of trees for parameter dimension j,
- $\boldsymbol{e}_j$ be the jth unit vector.

Initiate with $k = 0$,

$$\widehat{\boldsymbol{\theta}}(\boldsymbol{x}) = \widehat{\boldsymbol{\theta}}^{(0)} := \underset{\boldsymbol{\theta} \in \mathbb{R}^d}{\arg\min} \sum_{i=1}^n \mathcal{L}(\boldsymbol{\theta}; y_i). \tag{6}$$

While $k < \max\limits_{j=1,\dots,d} \kappa_j$:

$k \leftarrow k + 1$

For $j = 1, \dots, d$:

If $k \leq \kappa_j$:

(i) **Calculate** partial derivatives, for $i = 1, \dots, n$:

$$g_{ij} = \frac{\partial}{\partial \theta_j} \mathcal{L}(\boldsymbol{\theta}; y_i) \bigg|_{\boldsymbol{\theta} = \widehat{\boldsymbol{\theta}}(\boldsymbol{x}_i)}, \tag{7}$$

(ii) **Fit** tree regions

$$\widehat{\mathcal{A}}^{(k,j)} = \underset{\mathcal{A}}{\arg\min} \sum_{\mathcal{A}_l \in \mathcal{A}} \sum_{i : \boldsymbol{x}_i \in \mathcal{A}_l} (g_{ij} - \bar{g}_l)^2$$

where $\bar{g}_l$ is the average of g_{ij} for observations s.t. $\boldsymbol{x}_i \in \mathcal{A}_l$.

(iii) **Adjust** terminal node values

$$\widehat{\gamma}_l^{(k,j)} = \underset{\gamma \in \mathbb{R}}{\arg\min} \sum_{i : \boldsymbol{x}_i \in \widehat{\mathcal{A}}_l^{(k,j)}} \mathcal{L}(\widehat{\boldsymbol{\theta}}(\boldsymbol{x}_i) + \gamma \epsilon_j \boldsymbol{e}_j, y_i), \quad l = 1, \dots, |\widehat{\mathcal{A}}^{(k,j)}|.$$

(iv) **Update** parameter function

$$\widehat{\boldsymbol{\theta}}(\boldsymbol{x}) \leftarrow \widehat{\boldsymbol{\theta}}(\boldsymbol{x}) + \boldsymbol{e}_j \epsilon_j \sum_{\mathcal{A}_l \in \widehat{\mathcal{A}}^{(k,j)}} \mathbb{1}_{\{\boldsymbol{x} \in \mathcal{A}_l\}} \widehat{\gamma}_l^{(k,j)}.$$

Return

$$\widehat{\boldsymbol{\theta}}(\boldsymbol{x}) = \widehat{\boldsymbol{\theta}}^{(0)} + \sum_{j=1}^d \sum_{k=1}^{\kappa_j} \sum_{\mathcal{A}_l \in \widehat{\mathcal{A}}^{(k,j)}} \boldsymbol{e}_j \epsilon_j \mathbb{1}_{\{\boldsymbol{x} \in \mathcal{A}_l\}} \widehat{\gamma}_l^{(k,j)}$$

Note that the individual components of $\widehat{\boldsymbol{\theta}}(\boldsymbol{x})$ can be written as individual GBMs, i.e.

$$\widehat{\theta}_j(\boldsymbol{x}) = \widehat{\theta}_j^{(0)} + \sum_{k=1}^{\kappa_j} \sum_{\mathcal{A}_l \in \widehat{\mathcal{A}}^{(k,j)}} \epsilon_j \widehat{\gamma}_l^{(k,j)}$$

The standard, univariate, GBM from Friedman (2001) is obtained by setting $d = 1$.

Since GBMs are overparameterised, the CGBM is prone to overfitting. To regularise the CGBM, one needs to find appropriate hyperparameters (e.g. max depth and minimum samples per terminal region) for the trees, reasonable shrinkage factors ϵ_j and number of trees κ_j. Algorithm 3 in Delong et al. (2023) describes a procedure for finding appropriate values of κ_j using early stopping, given a fixed value of ϵ_j. The algorithm examines loss function contributions when adding new trees, and is based on the idea that trees should be added to dimension j of the parameter function until an additional tree no longer decreases the loss. The method utilises a validation data set or cross-validation to determine when to stop adding trees. This means that if there is no evidence of any structure in the parameter dimension j, i.e. $\theta_j(\boldsymbol{x}) \equiv \theta_j^{(0)}$, the algorithm will, given a sufficient amount of data, set $\kappa_j = 0$.

As mentioned earlier, GBMs are generally considered to be more interpretable than FFNNs. One reason for this is the ability to calculate *feature importance scores* as defined by Breiman et al. (1984). The feature importance score FI_j of a feature x_j in a regression tree is defined as the total reduction of the loss function when using feature x_j in any of the binary splits in the tree partition $\mathcal{A}$ of the feature space. Friedman (2001) uses the same score for the GBM, and proposes that the feature importance scores can be calculated for the GBM by summing the feature importance scores for each tree in the model. This property is inherited by the CGBM and can be used to assess the importance of features not only for the overall model – but also for each individual component $\theta_j(\boldsymbol{x})$. This is done by calculating the total reduction of the loss function when using feature x_k in any of the trees that are used in the boosting steps of parameter dimension j.

2.2.2 A cyclically boosted VCM

As discussed in the beginning of Section 2.2, the local GLM formulation (5) can be estimated using the CGBM using $\boldsymbol{\theta}(\boldsymbol{z}) = \boldsymbol{\beta}(\boldsymbol{z})$ based on the loss function $\mathcal{L}(\mu(\boldsymbol{x}; \beta_0, \boldsymbol{\beta}); y)$, where $\boldsymbol{\beta} \in \mathbb{R}^p$. This leads to the observation-specific directional partial derivatives

$$g_{ij} = x_{ij} \frac{\partial}{\partial \mu} \mathcal{L}(\mu; y_i) \bigg|_{\mu = \mu(\boldsymbol{x}_i; \beta_0, \boldsymbol{\beta})} \cdot \frac{\partial}{\partial v} u^{-1}(v) \bigg|_{v = \beta_0 + \boldsymbol{\beta}^\top \boldsymbol{x}_i}$$

from (3). That is, (7) can be written as

$$g_{ij} = x_{ij} \frac{\partial}{\partial \mu} \mathcal{L}(\mu; y_i) \bigg|_{\mu = \mu(\boldsymbol{x}_i; \widehat{\beta}_0, \widehat{\boldsymbol{\beta}}(\boldsymbol{z}_i))} \cdot \frac{\partial}{\partial v} u^{-1}(v) \bigg|_{v = \widehat{\beta}_0 + \widehat{\boldsymbol{\beta}}(\boldsymbol{z}_i)^\top \boldsymbol{x}_i}. \tag{8}$$

By using the above directional partial derivatives in the CGBM from Section 2.2.1 we obtain a cyclically boosted VCM. The natural initialisation of the model is using (6), which is equivalent to initialising with a standard GLM, see (2). See Algorithm 2 for the full training procedure.

Algorithm 2 A cyclically boosted VCM

Let:

- $\mathcal{L}(\mu; y)$ be a loss function,
- $(y_i, \boldsymbol{x}_i, \boldsymbol{z}_i)_{i=1}^n$ be independent observations,
- $\mu(\boldsymbol{x}; \beta_0, \boldsymbol{\beta}) = u^{-1}(\beta_0 + \boldsymbol{\beta}^\top \boldsymbol{x})$ be a mean function,
- $\mathcal{A}$ be a disjoint partition of $\mathbb{Z}$ into $|\mathcal{A}|$ regions,
- $\gamma_l \in \mathbb{R}$ be terminal node values, $l = 1, \dots, |\mathcal{A}|$,
- ϵ_j and κ_j be shrinkage parameters and number of trees for coefficient function j,

Initiate with $k = 0$,

$$(\widehat{\beta}_0, \widehat{\boldsymbol{\beta}}(\boldsymbol{z})) = (\widehat{\beta}_0, \widehat{\boldsymbol{\beta}}^{\text{GLM}}) := \underset{(\beta_0, \boldsymbol{\beta}) \in \mathbb{R}^{p+1}}{\arg\min} \sum_{i=1}^n \mathcal{L}(\mu(\boldsymbol{x}_i; \beta_0, \boldsymbol{\beta}), y_i). \tag{9}$$

While $k < \max_{j=1,\dots,p} \kappa_j$:

$k \leftarrow k + 1$

For $j = 1, \dots, p$:

If $k \leq \kappa_j$:

(i) **Calculate** partial derivatives, for $i = 1, \dots, n$:

$$g_{ij} = x_{ij} \frac{\partial}{\partial \mu} \mathcal{L}(\mu; y_i) \bigg|_{\mu = \mu(\boldsymbol{x}_i; \widehat{\beta}_0, \widehat{\boldsymbol{\beta}}(\boldsymbol{z}_i))} \cdot \frac{\partial}{\partial z} u^{-1}(v) \bigg|_{v = \widehat{\beta}_0 + \widehat{\boldsymbol{\beta}}(\boldsymbol{z}_i)^\top \boldsymbol{x}_i}.$$

(ii) **Fit** tree regions

$$\widehat{\mathcal{A}}^{(k,j)} = \underset{\mathcal{A}}{\arg\min} \sum_{\mathcal{A}_l \in \mathcal{A}} \sum_{i: \boldsymbol{z}_i \in \mathcal{A}_l} \left(g_{ij}^{(k)} - \bar{g}_l^{(k)} \right)^2 \tag{10}$$

where $\bar{g}_l^{(k)}$ is the average of $g_{ij}^{(k)}$ for observations s.t. $\boldsymbol{z}_i \in \mathcal{A}_l$.

(iii) **Adjust** terminal node values, for $l = 1, \dots, |\widehat{\mathcal{A}}^{(k,j)}|$:

$$\widehat{\gamma}_l^{(k,j)} = \underset{\gamma \in \mathbb{R}}{\arg\min} \sum_{i: \boldsymbol{z}_i \in \widehat{\mathcal{A}}_l^{(k,j)}} \mathcal{L}\left(u^{-1}\left(\widehat{\beta}_0 + \widehat{\boldsymbol{\beta}}(\boldsymbol{z}_i)^\top \boldsymbol{x}_i + \gamma x_{ij} \right), y_i \right).$$

(iv) **Update** coefficient function

$$\widehat{\beta}_j(\boldsymbol{z}) \leftarrow \widehat{\beta}_j(\boldsymbol{z}) + \epsilon_j \sum_{\mathcal{A}_l \in \widehat{\mathcal{A}}^{(k,j)}} \mathbb{1}_{\{\boldsymbol{z} \in \mathcal{A}_l\}} \widehat{\gamma}_l^{(k,j)}.$$

Restore unbiasedness

$$\widehat{\beta}_0 = \underset{\beta_0 \in \mathbb{R}}{\arg\min} \sum_{i=1}^n \mathcal{L}(u^{-1}(\beta_0 + \widehat{\boldsymbol{\beta}}(\boldsymbol{z}_i)^\top \boldsymbol{x}_i); y_i). \tag{11}$$

Return

$$\widehat{\mu}(\boldsymbol{x}) = u^{-1}(\widehat{\beta}_0 + \sum_{j=1}^p \widehat{\beta}_j(\boldsymbol{z}) x_j)$$

$$\widehat{\beta}_j(\boldsymbol{z}) = \widehat{\beta}_j^{\text{GLM}} + \epsilon_j \sum_{k=1}^{\kappa_j} \sum_{\mathcal{A}_l \in \widehat{\mathcal{A}}^{(k,j)}} \mathbb{1}_{\{\boldsymbol{z} \in \mathcal{A}_l\}} \widehat{\gamma}_l^{(k,j)}, \quad j = 1, \dots, p. \tag{12}$$

Note that the intercept term β_0 is not updated during the boosting iterations, and will be kept fixed at the initial value from (9). This is done to ensure the structure of the data is instead captured by the regression coefficient functions.

Using the initialisation in (9) it is clear that this model is a generalisation of a standard GLM, and that we may express the tree-based coefficient function estimates in (12) as

$$\widehat{\beta}_j(\boldsymbol{z}) = \widehat{\beta}_j^{\text{GLM}} + \widehat{\Delta}_j(\boldsymbol{z}), \tag{13}$$

where

$$\widehat{\Delta}_j(\boldsymbol{z}) := \epsilon_j \sum_{k=1}^{\kappa_j} \sum_{\mathcal{A}_l \in \widehat{\mathcal{A}}^{(k,j)}} \mathbb{1}_{\{\boldsymbol{z} \in \mathcal{A}_l\}} \widehat{\gamma}_l^{(k,j)}. \tag{14}$$

By using this representation, it becomes straightforward to assess in which dimensions, and parts of the effect modifier feature space that the model deviates from the corresponding GLM. Note also that if one uses a log-link, the estimate of the mean function can be written as

$$\widehat{\mu}(\boldsymbol{x}; \beta_0, \boldsymbol{\beta}(\boldsymbol{z})) = \exp\left\{ \widehat{\beta}_0^{\text{GLM}} + \sum_{j=1}^{p} \widehat{\beta}_j^{\text{GLM}} x_j + \sum_{j=1}^{p} \widehat{\Delta}_j(\boldsymbol{z}) x_j \right\} \tag{15}$$

$$= \widehat{\mu}^{\text{GLM}}(\boldsymbol{x}) \cdot \exp\left\{ \sum_{j=1}^{p} \widehat{\Delta}_j(\boldsymbol{z}) x_j \right\} \tag{16}$$

meaning that a boosted VCM can be seen as a GLM with a multiplicative correction term.

Once the model has been estimated, the balance property needs to be restored to achieve global unbiasedness. This is why the intercept term β_0 is updated in (11).

Remark 2 *Note that the regression coefficient functions $\beta_j(\boldsymbol{z})$ in the cyclically boosted VCM do not necessarily have to depend on the same effect modifiers $\boldsymbol{z}$. It is straightforward to extend the model to allow for different effect modifiers for each coefficient function, i.e. replacing the mean function in* (5) *with*

$$\mu(\boldsymbol{x}; \beta_0, \beta_1(\boldsymbol{z}_1), \ldots, \beta_p(\boldsymbol{z}_p)) := u^{-1}(\beta_0 + \sum_{j=1}^{p} \beta_j(\boldsymbol{z}_j) x_j).$$

The only difference in Algorithm 2 would be that the features used for splitting the trees in (10) *would be different for each coefficient function.*

Note that the feature importance scores FI_j discussed in Section 2.2.1 can be calculated for the individual coefficient functions $\beta_j(\boldsymbol{z})$, in order to see which features are important for each coefficient function. Wang and Hastie (2014) use the feature importance score of the total model to assess the importance of features for the individual coefficient functions. This can be a useful measure if one assumes that the predictive features are standardised to be on the same scale – and that $\mathbb{X} \cap \mathbb{Z} = \emptyset$. However, if $\mathbb{X} \cap \mathbb{Z} \neq \emptyset$, the feature importance scores for effect modifier x_j for the full model is no longer as useful – since the effect modifier x_j will then also effect the output by the term $\beta_j(\boldsymbol{z})x_j$, which will not be captured by the feature importance score. In Richman and Wüthrich (2023) the importance measure

$$\text{FI}_j^* = \frac{1}{n} \sum_{i=1}^{n} \left| \widehat{\beta}_j(\boldsymbol{z}_i) \right| \tag{17}$$

is proposed for local GLMs, see Section 3.5 in Richman and Wüthrich (2023), where $\widehat{\beta}_j(\boldsymbol{z}_i)$ is the coefficient function estimate $\widehat{\beta}_j(\boldsymbol{z})$ evaluated at $\boldsymbol{z}_i$ and $i = 1, \dots, n$ is the training sample. The scores from (17) are normalised to sum to one and are used to assess the importance of feature x_j based on its regression coefficient function. This is materially different from the feature importance scores used to assess the importance of feature x_j *within* a regression coefficient model. Note that FI_j^*, too, assumes that the features are standardised to be on the same scale, and will neglect the potential effect of feature x_j on other coefficient functions $\beta_k(\boldsymbol{z})$, $k \neq j$ in cases where $\mathbb{X} \cap \mathbb{Z} \neq \emptyset$.

2.2.3 Comments on convergence

As discussed in Section 2.2.2, the cyclically boosted VCM is constructed using the CGBM from Delong et al. (2023). The CGBM aims at estimating an unknown d-dimensional parameter function $\boldsymbol{\theta}(\boldsymbol{x})$. In the special case of the model, the unknown parameter function corresponds to the mean function from (5), which is expressed in terms of $p+1$ different components, i.e. the intercept β_0 together with the coefficient functions $\beta_i(\boldsymbol{x}), i = 1, \dots, p$. Further, for EDFs, modeling an unknown, but in the local GLM and VCM sense, structured mean function, it is natural to consider deviance loss-functions, see e.g. Chapter 1 in Jørgensen (1997), given by

$$D(\mu; (y_i)_{i=1}^n) = \sum_{i=1}^n d(\mu; y_i), \tag{18}$$

where

$$d(\mu; y_i) \propto \log L(y_i; y_i) - \log L(\mu; y_i),$$

where $\log L(\mu; y)$ is the log-likelihood for a single observation w.r.t. an unknown mean parameter μ in an EDF. Thus, by using $\mathcal{L}(\mu; y_i) = D(\mu; y_i)$, it follows that the MLE minimises the deviance loss. Moreover, from (18), it follows that the deviance loss-function attains its minimum, which is zero, when the saturated model is reached, i.e. when $\widehat{\mu}(\boldsymbol{x}_i) = y_i$. Consequently, running the CGBM algorithm without early stopping will result in in-sample convergence, see Proposition 1 in Delong et al. (2023) and the surrounding discussion, since no cyclic update can result in an increasing loss. In particular, if the partial derivative in a specific covariate dimension, say j, in iteration k, can not be improved, the added tree will have terminal node values equal to 0. Similarly, if the partial derivative attains the value 0, the added tree will again have terminal node values equal to 0. This, however, does not guarantee that the minimum is unique in terms of the coefficient functions, it only tells us that the algorithm will, in-sample, converge to a local minimum, see Lemma 1 in Delong et al. (2023).

The above discussion only focuses on in-sample convergence. The question of out-of-sample (population-based) consistency is considerably harder to address. In Zhou and Hooker (2022) consistency is discussed for the situation with L^2-boosting when the univariate tree-approximations are updated jointly by making use of the construction from Bühlmann (2002). This approach, however, does not apply to the CGBM; see the discussion in Section 5 in Zhou and Hooker (2022). For more on the convergence of the CGBM, see the discussion in Delong et al. (2023) relating to the results in Zhang and Yu (2005). These ideas are not pursued further in the current paper.

3 Modeling considerations

Three topics concerning modeling using VCMs, and in particular the tree-based VCM introduced in this paper, will be discussed further, namely feature selection, interaction effects, and the handling of categorical variables.

Remark 3 *The statements on the interpretation of coefficient functions in local GLMs, presented in Remark 2.3 of Richman and Wüthrich (2023) apply to VCMs in general, and the model proposed in this paper in particular, and not only to local GLMs. We re-iterate them here for completeness, with some additional comments.*

(i) If $\beta_j(\boldsymbol{z}) = \beta_j \neq 0$, i.e. the function is non-zero but does not depend on $\boldsymbol{z}$, then the structure apart from the GLM term, e.g. (14) *in a VCM, is unnecessary.*

(ii) If $\beta_j(\boldsymbol{z}) \equiv 0$, the term $\beta_j(\boldsymbol{z})x_j$ should not be included in the model. Note, however, that we can still use x_j and $\boldsymbol{z}$ as input to other coefficient functions, i.e. having $\mathbb{X}_j \subset \mathbb{Z}$.

(iii) If $\beta_j(\boldsymbol{z}) = \beta_j(z_k)$, i.e. the coefficient function for feature j only depends on z_k, there are no effects from other features any other feature for this term, and they can be removed as input to β_j (see Remark 2).

(iv) Note that when $\mathbb{Z} = \mathbb{X}$, as in Richman and Wüthrich (2023), we can never guarantee identifiability of the coefficient functions, since we can express any term $\beta_j(\boldsymbol{x})x_j$ as

$$\beta_j(\boldsymbol{x})x_j = \beta_k(\boldsymbol{x})x_k$$

using $\beta_k(\boldsymbol{x}) = \beta_j(\boldsymbol{x})x_j/x_k$. However, in the numerical illustration in Section 4.1, this does not seem to be a big problem since the "true" coefficient functions were estimated with reasonable accuracy. See also Appendix A.1 for a discussion on identifiability.

We continue with discussing specific properties of the CGBMs and their implications on the modeling of coefficient functions.

3.1 Feature selection

Feature selection, i.e. picking predictive features $\boldsymbol{x}$ (for picking effect modifier features $\boldsymbol{z}$, see Section 3.2), for the cyclically boosted VCM is included automatically in the dimension-dependent early stopping procedure. The early stopping scheme presented in Delong et al. (2023) uses different stopping times κ_j for different parameter dimensions j. In the cyclically boosted VCM, this means that a different number of trees can be used for different coefficient function estimates $\widehat{\beta}_j(\boldsymbol{z})$. The early stopping scheme is based on the idea that the model is trained until the loss function is no longer reduced by adding new trees. This is done for each dimension and for a feature j where there is no evidence of coefficient function estimate $\widehat{\beta}_j(\boldsymbol{x})$ providing any additional information over the initiation $\widehat{\beta}_j^{\text{GLM}}$, the hyperparameter κ_j will be close to zero. Also, if $\widehat{\beta}_j^{\text{GLM}} = 0$, the feature j will not be used in the model unless it is included in the effect modifier set $\mathbb{Z}$. Consequently, the early stopping procedure will automatically remove or at least reduce the impact of redundant features. This is a simplification compared to the method used for FNNs in Richman and Wüthrich (2023).

Further, by analysing the feature importance scores for each coefficient function (see Section 3.2), it is straightforward to manually reduce the model complexity by removing features that have little influence on a specific coefficient function. In order to avoid drawing conclusions based on noise, this step could, e.g., be combined with a bootstrap procedure.

3.2 Interaction effects

The fact that it is possible to compute dimension-wise feature importance scores for the coefficient function estimates in the cyclically boosted VCM makes it simple to assess interaction effects between effect modifiers $\boldsymbol{z}$ and predictive features $\boldsymbol{x}$. This is done by examining the feature importance of effect modifier z_k on coefficient function estimate $\widehat{\beta}_j(\boldsymbol{z})$. If a feature importance score for effect modifier z_k on coefficient function estimate $\widehat{\beta}_j(\boldsymbol{z})$ is ≈ 0, this indicates no interaction effect between z_k and x_j. The effect modifier z_k can then be removed from that coefficient function estimate, i.e. by removing z_k from $\boldsymbol{z}_j$ (see Remark 2). If an effect modifier gets no feature importance score for any coefficient function estimate, it can be removed from the effect modifier vector completely. Note that it might still be included in the predictive feature vector $\boldsymbol{x}$. This step could be combined with a bootstrap procedure. The above procedure is a simplification compared to the method used for FNNs in Richman and Wüthrich (2023).

3.3 Categorical features

The VCM structure from (5) allows for the use of categorical features in the effect modifier vector $\boldsymbol{z}$ as long as the chosen functional form of the coefficient functions $\beta_j(\boldsymbol{z})$ can handle categorical features. Since the coefficient functions $\beta_j(\boldsymbol{z})$ are modelled as GBMs in the model proposed in this paper, the model can handle categorical features natively. Note, however, that in order to use categorical features in the predictive features $\boldsymbol{x}$, one has to encode them as one-hot encoded features, as done in Richman and Wüthrich (2023). This means a categorical feature x_j is replaced by a set of binary features $\boldsymbol{x}_j^*$, where each feature $\boldsymbol{x}_{j,l}^*$ corresponds to a category c_l of feature x_j, i.e.

$$\boldsymbol{x}_j^* = \begin{pmatrix} \mathbb{1}_{\{x_j=c_1\}} \\ \vdots \\ \mathbb{1}_{\{x_j=c_{m_j}\}} \end{pmatrix},$$

where m_j is the number of categories of feature x_j. Note that one usually removes one of the categories when using this encoding in GLM settings since the intercept term β_0 is included in the model. This will not be done here, since the coefficient functions will be modeled as GBMs, whereas the intercept term β_0 will not.

Note that the update steps of a set of one-hot encoded predictive features can be performed in parallel since the multiplicative term structure makes the updates of coefficient function estimates $\widehat{\beta}_{j,l}(\boldsymbol{z})$ independent of each other for all $l = 1, \ldots, m_j$ (see Algorithm 2). This speeds up the estimation procedure, making it a practically implementable approach.

4 Examples

The following numerical examples illustrate the performance of the proposed method on simulated and real data. Throughout this section we will use $\mathbb{Z} = \mathbb{X}$, and let $\mu(\boldsymbol{x}) := \mu(\boldsymbol{x}; \beta_0, \boldsymbol{\beta}(\boldsymbol{x}))$ Both datasets are the same as the ones used in Section 3 of Richman and Wüthrich (2023). The `python` implementation of this special case of our VCM is available at `https://github.com/henningzakrisson/local-glm-boost` under the name `LocalGLMboost`.

4.1 Simulated data

The following simulated example reproduces the one in Section 3.1 of Richman and Wüthrich (2023). The data is generated using features $\boldsymbol{X} \in \mathbb{R}^8$, where $\boldsymbol{X}$ follows a multivariate normal distribution with mean vector $\mathbf{0}$ and unit variances. The features are independent, except for X_2 and X_8 which have a correlation of 0.5. The true regression function $\mu(\boldsymbol{x})$ is defined as

$$\mu(\boldsymbol{x}) = \boldsymbol{\beta}(\boldsymbol{x})^\top \boldsymbol{x}$$

Attention	Expression
$\beta_1(\boldsymbol{x})$	0.5
$\beta_2(\boldsymbol{x})$	$-\frac{1}{4}x_2$
$\beta_3(\boldsymbol{x})$	$\frac{1}{2}\text{sgn}(x_3)\sin(2x_3)$
$\beta_4(\boldsymbol{x})$	$\frac{1}{4}x_5$
$\beta_5(\boldsymbol{x})$	$\frac{1}{4}x_4$
$\beta_6(\boldsymbol{x})$	$\frac{1}{8}x_5^2$
$\beta_7(\boldsymbol{x})$	0
$\beta_8(\boldsymbol{x})$	0

Table 1: True coefficient functions $\beta_j(\boldsymbol{x})$ for the simulated data example.

where $\boldsymbol{\beta}(\boldsymbol{x}) = (\beta_1(\boldsymbol{x}), \dots, \beta_8(\boldsymbol{x}))^\top$ is defined in Table 1. Note that X_7 and X_8 are not included in the true regression function as either effect modifiers or predictive features, but X_8 is correlated with X_2. Note also that the coefficient function representation is not unique for this $\mu(\boldsymbol{x})$ (see Remark 3 and Appendix A.3). Responses y are generated according to

$$Y_i \mid \boldsymbol{X} = \boldsymbol{x}_i \sim \mathcal{N}\left(\mu\left(\boldsymbol{x}_i\right), 1\right).$$

In Appendix A the $\beta_j(\boldsymbol{x})$s from Table 1 are identified as the optimal solutions to a local GLM, or VCM, when knowing that the true $\mu(\boldsymbol{x})$ is of the form (5). Henceforth, the $\beta_j(\boldsymbol{x})$s from Table 1 will be referred to as the true coefficient functions.

A total number of 200 000 observations are generated and split into a training and a test set of equal size. The cyclically boosted tree-based VCM presented in Section 2.2.2, henceforth referred to as the TVCM, is tuned using dimension-wise early stopping according to Algorithm 3 in Delong et al. (2023), using two equally sized training and validation splits. This produces a set of stopping times κ_j, $j = 1, \dots, 8$. This algorithm relies on setting the other tree and boosting hyperparameters to reasonable values without any in-depth evaluation; the max depth is set to 2, the minimum samples per leaf node is set to 10, and the learning rate ϵ_j is set to 0.01, $j = 1, \dots, 8$.

The results are compared to the true regression function $\mu(\boldsymbol{x})$ as well as the results of the LocalGLMnet model from Richman and Wüthrich (2023). An intercept-only model and a GLM without interactions are also included for comparison. Since the negative log-likelihood of the normal distribution is proportional to the Mean Squared Error (MSE), the MSE is used as the evaluation metric.

The MSE results for the training and test data can be seen in Table 2. From Table 2

	Train	Test
True	1.002	0.996
Intercept	1.791	1.792
GLM	1.524	1.527
TVCM	1.008	1.013
LocalGLMnet	1.002	1.005

Table 2: MSE results for the simulated data example.

it is seen that the TVCM is able to capture the true regression function well, with an MSE close to the expected (being 1.00). It is also seen that an intercept-only model and a GLM without interactions perform poorly. The performance of the TVCM is close to that of the LocalGLMnet.

The estimated means $\widehat{\mu}(\boldsymbol{x})$ compared with the true model means $\mu(\boldsymbol{x})$ on 500 randomly selected observations from the test set for the GLM and TVCMs respectively are shown in Figure 1. It is apparent from Figure 1 that the TVCM model is able to capture the true

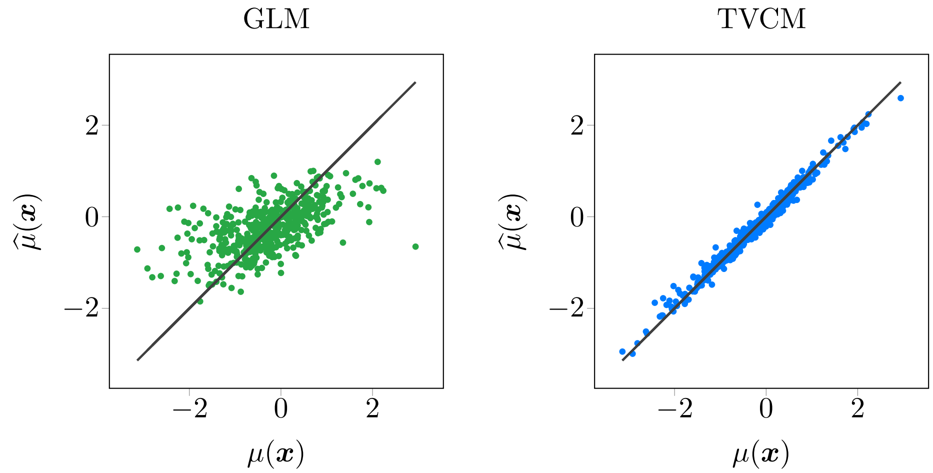

Figure 1: Estimated means $\widehat{\mu}(\boldsymbol{x})$ vs true regression function $\mu(\boldsymbol{x})$ for 500 randomly selected observations from the test set for the GLM and TVCMs respectively.

regression function well, while the GLM model is not. These results are similar to those of the LocalGLMnet (see Figure 1 in Richman and Wüthrich (2023)).

The number of parameters κ_j from the early stopping procedure as well as the constant parameters $\widehat{\beta}_j^{\text{GLM}}$, are shown in Table 3. The table also presents th FI_j^* from (17) for the TVCM. Coefficient function estimate $\widehat{\beta}_1$ got the expected number of estimators, $\kappa_1 = 0$, since

	κ_j	$\widehat{\beta}_j^{\text{GLM}}$	FI_j^*
x_1	0	0.500	0.33
x_2	469	-0.001	0.04
x_3	563	0.033	0.24
x_4	271	0.008	0.13
x_5	363	-0.001	0.13
x_6	321	0.123	0.08
x_7	24	-0.001	0.00
x_8	220	-0.001	0.04

Table 3: Hyperparameters κ_j, GLM estimates $\widehat{\beta}_j^{(0)}$ and FI_j^* scores for the TVCM in the simulated data example.

this function contains no $\boldsymbol{x}$-dependent terms. For $\widehat{\beta}_7$, the number of estimators κ_7 is non-zero even though $\beta_7 \equiv 0$ in the true regression function, which means that the model is overfitting to the training data. The number of estimators is, however, not very large compared to the other coefficient function estimates. The fact that κ_8 is not zero is likely a consequence of the correlation between X_2 and X_8.

As for the GLM terms $\widehat{\beta}_j^{\text{GLM}}$, note that the expected value of the coefficient functions, $\mathbb{E}[\beta_j(X)]$, is equal to 0 for $j = 2, 4, 5, 7, 8$, which is mirrored in the GLM terms $\widehat{\beta}_j^{\text{GLM}}$ being close to zero. Also, note that $\mathbb{E}[\beta_1(X)] = 0.5$ and $\mathbb{E}[\beta_6(X)] = 0.125$, and that the GLM terms $\widehat{\beta}_1^{\mathrm{GLM}}$ and $\widehat{\beta}_6^{\mathrm{GLM}}$ are close to these values. For β_3, the GLM term $\widehat{\beta}_3^{\mathrm{GLM}}$ is close to zero, which is not expected since $\mathbb{E}[\beta_3(X)] \approx 0.255$.

The FI_j^*s from (17) for the TVCM in Table 3 is the highest for coefficient function estimate $\widehat{\beta}_1$, which is expected since the constant β_1 is larger than the other coefficient functions are on average. The term $\widehat{\beta}_3$ gets the second highest FI_j^*, which is also expected since $\beta_3(\boldsymbol{x})$ has a large amplitude. In Figure 3 the true coefficient functions $\beta_j(\boldsymbol{x})$ are compared with their estimates based on the TVCM.

The feature importance score FI_j, as defined by Friedman (2001), of each effect modifier feature on each individual coefficient function for the TVCM are shown in Table 2. The

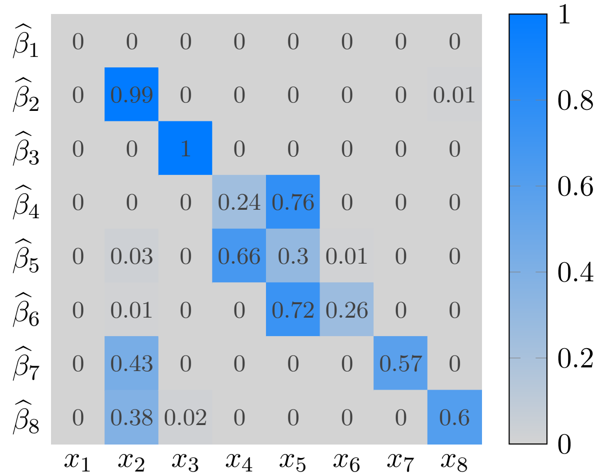

Figure 2: Feature importance scores for the simulated data example. Cell $(x_j, \widehat{\beta}_k)$ corresponds to the feature importance score of feature x_j for the coefficient function $\widehat{\beta}_k$, i.e. the relative loss reduction from splits in the trees of coefficient function estimate $\widehat{\beta}_k$ that used feature x_j. All rows are normalised to sum to one. Note that $\widehat{\beta}_0$ is the intercept and does not depend on any features, and that $\kappa_1 = 0$ means that coefficient function estimate $\widehat{\beta}_1$ consists of the GLM initiation only.

feature importances in Figure 2 are mainly in line with expectations, with the important features for each coefficient function estimate being the ones specified in Table 1. For β_4 and β_5, the coefficient function estimates however also seem to use their respective features x_4 and x_5 to model the interaction term x_4x_5. Interestingly, $\widehat{\beta}_5$ does not use x_6 at all, which means that the interaction term $x_5^2x_6$ is modeled solely in $\widehat{\beta}_6$. Also, $\widehat{\beta}_8$ is mostly using x_2 as a feature, which indicates that it is using the correlation between x_2 and x_8 to model the x_2^2 term. The feature interaction analysis in Richman and Wüthrich (2023) indicate largely the same results (see Figure 4 in Richman and Wüthrich (2023) and the corresponding discussion).

The estimated coefficient functions $\widehat{\beta}_j(\boldsymbol{x})$ from the TVCM are shown in Figure 3. The function estimates are shown as compared to the feature values of the effect modifier in the true regression function in Table 1. The coefficient function estimates from the TVCM are compared with the corresponding $\beta_j(\boldsymbol{x})$s from Table 1. For $\widehat{\beta}_1$, $\widehat{\beta}_7$ and $\widehat{\beta}_8$, the x-axis uses the feature values of the effect modifier with the highest feature importance score from Table 2, since the true coefficient functions are not feature-dependent for these coefficient functions.

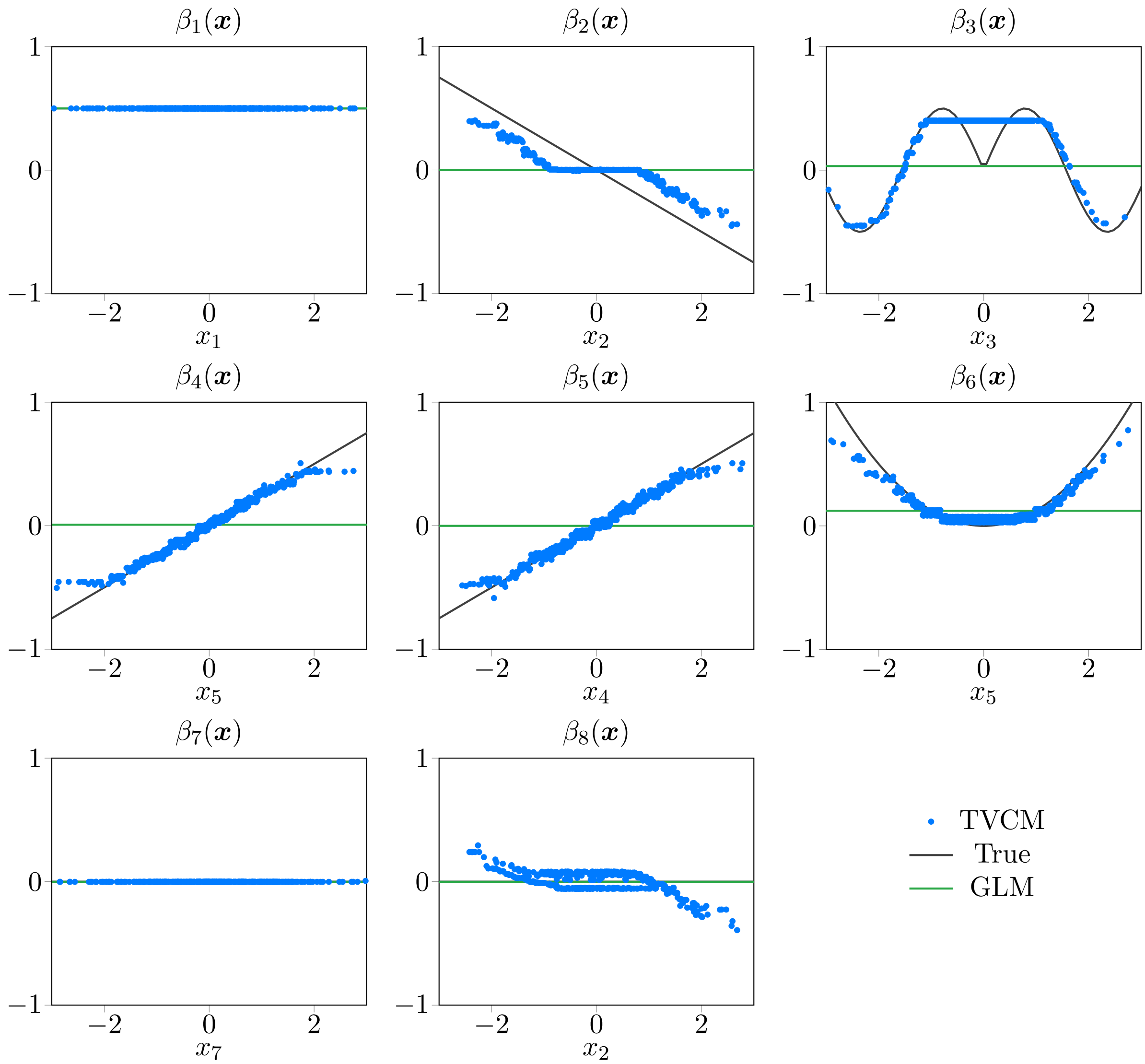

Figure 3: Coefficient function estimates $\widehat{\beta}_j(\boldsymbol{x})$ for the TVCM estimates on the simulated data set for different values of x_l, where l is the effect modifier with the highest feature importance score for each coefficient function. The GLM initiation $\widehat{\beta}_j^{(0)}$ is also shown, as well as the true regression definitions from Table 1.

From Table 2 and Figure 3 it is seen that the TVCM is able to capture the true coefficient functions well, and that the terms specified by Table 1 also are the ones that the TVCM is modeling. This implies that the identification problem discussed in Remark 3 is not material. Note that the coefficient function estimates $\widehat{\beta}_j$ for $j = 2, 3$ is not very accurate for observations close to $x_j = 0$. This is not surprising, since the term $\beta_j(\boldsymbol{x})x_j$ is small in those observations and does, hence, not affect the mean $\mu(\boldsymbol{x})$ much. Also, the shape of the coefficient function estimate $\widehat{\beta}_8$ indicates that this is affected by the correlation between x_2 and x_8. While Figure 3 is not identical to Figure 2 in Richman and Wüthrich (2023), which is used for feature selection, it shows similar results for the estimated regression attentions. For coefficient function estimates $\widehat{\beta}_1$, $\widehat{\beta}_2$, $\widehat{\beta}_3$, and $\widehat{\beta}_7$, both models are able to capture the true coefficient functions well. For β_1 and β_7, the TVCM is better at capturing the flat structure of the coefficient, where the LocalGLMnet model is more noisy and seems to be slightly overfitting.

For β_8, the LocalGLMnet coefficient function estimates are more noisy, but of the same magnitude as the TVCM equivalents.

4.2 Real data

In order to be able to compare with the results for the LocalGLMnet model from Richman and Wüthrich (2023), the real world example is based on the `FreMTPLfreq2` dataset from Dutang et al. (2020). The dataset is a widely used non-life insurance dataset with the number and severity of claims for a portfolio of French motor third-party liability insurance policies (see (Wüthrich and Merz, 2023, Chapter 13.1) for a detailed description).

As in Richman and Wüthrich (2023), the focus will be on claim count modelling using Poisson regression models with duration weights. That is, it is assumed that

$$Y \mid \boldsymbol{X} = \boldsymbol{x}_i, W = w_i \sim \text{Pois}(w_i \mu(\boldsymbol{x}_i)).$$

where Y is the number of claims, $\boldsymbol{X}$ is a vector of policy data, and W is the duration of exposure. By considering the number of claims as the response Y_i, the duration of exposure as weights w_i, and policy data $\boldsymbol{x}_i$, the dataset can be used to fit the TVCM, using the Poisson deviance as the loss function. Note that this requires that the duration weights be incorporated in the loss function, which is a simple extension of the algorithms described in this paper.

The dataset contains 678 008 observations, where each observation corresponds to a policy. Further, the data cleaning and train-test-split from Wüthrich and Merz (2023) will be used. This results in a dataset $(y_i, w_i, \boldsymbol{x}_i)_i$ which is split into a training dataset of size 610 206 and a test dataset of size 67 801. The feature vector $\boldsymbol{x}_i$ contains 6 ordinal features and 3 categorical features. The categorical features (`Region`, `VehGas` and `VehBrand`) are handled using one-hot encoding in line with the discussion in Section 3.3. The one-hot encoding of the categorical features is done for the effect modifier features as well. This is due to limitations in the `python` implementation, which relies on the `DecisionTreeRegressor` from `sklearn`, which does not support categorical features.

The TVCM is tuned using the early stopping algorithm given by Algorithm 3 in Delong et al. (2023) using two equally sized training-validation splits. The other tree and boosting hyperparameters are set to reasonable values without any in-depth evaluation; the max depth is set to 2, the minimum samples per leaf node is set to 20, and the learning rate ϵ_j is set to 0.01 for all j.

The loss results in terms of Poisson deviance for the training and test data are summarised in Table 4. As can be seen, the TVCM achieves a lower out-of-sample Poisson deviance than

	Train	Test
Intercept	25.213	25.445
GLM	24.177	24.217
GBM	23.846	23.890
TVCM	23.751	23.839
LocalGLMnet	23.728	23.945

Table 4: Poisson deviance results (10^{-2}) for the real data example.

both the standard GBM and the LocalGLMnet model from Richman and Wüthrich (2023). While the deviance score differences are small, this shows that the training algorithm is able to achieve a fit that is comparable to its peers. It is also apparent that the TVCM is able to achieve a lower out-of-sample deviance than the GLM, which indicates that there is structure in the data beyond the linear term structure of a GLM.

Figure 4 shows the outcome on the test set compared to model predictions. The LocalGLMnet model from Richman and Wüthrich (2023) is not included in this figure since we do not have access to the individual predictions from this model. As can be seen in Figure 4, the

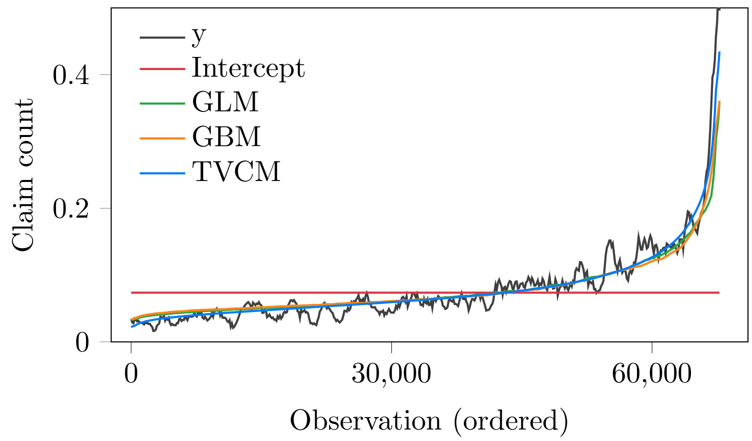

Figure 4: Out of sample predictions of $\mu(\boldsymbol{x})$ on the real data set, where $\mu(\boldsymbol{x}) = \mathbb{E}[Y_i \mid \boldsymbol{X} = \boldsymbol{x}_i, W = w_i]/w_i$. The predictions have been ordered according to the $\mu(\boldsymbol{x})$-predictions from the TVCM and averaged using a rolling mean of 1 000. For the true observations y, the rolling mean is taken over the true number of claims and divided by the rolling mean of the duration of exposure.

TVCM captures more of the structure in the data than the GLM and the GBM, especially for more extreme values. For the more moderate values, the three models more or less coincide.

The number of estimators κ_j for the different features can be seen in Table 5, together with the constant terms for every coefficient function, $\widehat{\beta}_j^{\mathrm{GLM}}$. This table also presents the FI_j^* scores as defined in (17). Note that the FI_j^* scores are not calculated for the categorical features since they are not on the same scale as the other features, see Richman and Wüthrich (2023) for a discussion. From Table 5, it is seen that most features are used in the coefficient

	κ_j	$\widehat{\beta}_j^{\mathrm{GLM}}$	FI_j^*
VehPower	33	0.07	0.08
VehAge	46	-0.10	0.12
Density	18	-0.01	0.01
DrivAge	241	0.08	0.11
BonusMalus	59	0.41	0.49
Area	91	0.15	0.18
VehGas	(139,187)	(-0.70,-0.87)	-
VehBrand	(6,145)	(-0.05,0.01)	-
Region	(0,187)	(-0.01,0.25)	-

Table 5: Number of estimators κ_j and GLM terms $\widehat{\beta}_j^{(0)}$ for the real data example. For the categorical features (`VehGas`, `VehBrand`, and `Region`), intervals of κ_j and β_{j0} are presented.

functions, except for parts of the one-hot encoded regions in `Region`. The FI_j^* scores of (17) for the different features indicate that the `BonusMalus` feature gets the largest absolute values of its coefficient function estimates, which agrees with Richman and Wüthrich (2023). However, the feature with the second highest FI_j^* according to the TVCM is `Area`, which contradicts the findings in Richman and Wüthrich (2023), whose analysis suggests that the feature should be dropped. This could be because `Area` is highly correlated with `Density`, which is given a higher value of FI_j^* in Richman and Wüthrich (2023).

The feature importance score FI_j, as defined by Friedman (2001), of each effect modifier feature on each individual coefficient function for the TVCM can be seen in Figure 5. Similarly

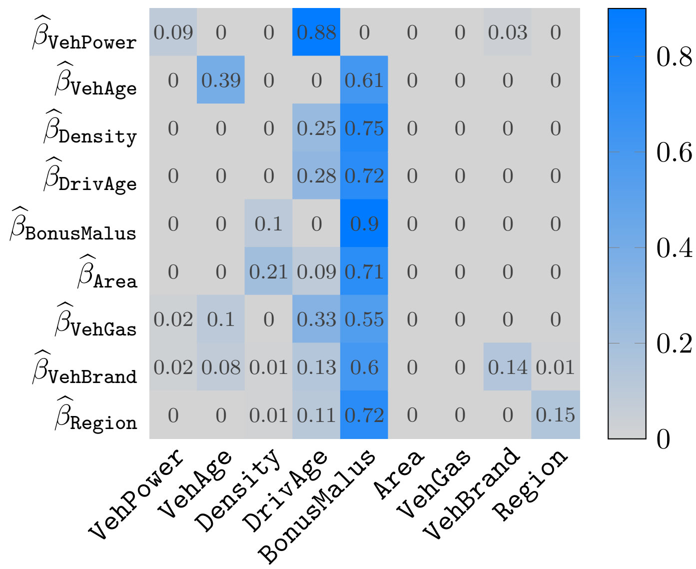

Figure 5: Feature importance scores for the real data example. Cell $(j, \widehat{\beta}_k)$ corresponds to the feature importance score of feature j for the coefficient function estimate $\widehat{\beta}_k$, i.e. the relative loss reduction from splits in the trees of coefficient function estimate $\widehat{\beta}_k$ that used feature j. All rows are normalised to sum to one. For the categorical features (`VehGas`, `VehBrand`, and `Region`), the feature importance scores have been summed over the different categories before normalisation.

to the findings in Richman and Wüthrich (2023), see their Figure 8, `BonusMalus` is the most influential feature on the coefficient function estimates as a whole. Both the TVCM and LocalGLMnet models agree that `DrivAge` is the second most influential feature for most coefficient function estimates. The fact that many feature importance scores in Figure 5 are close to zero suggests that the regression models can be reduced by removing features from the modeling. This, however, should be backed by, e.g., a bootstrap analysis.

A histogram of the coefficient function estimates $\widehat{\beta}_j(\boldsymbol{x})$ for the continuous predictive features x_j is shown in Figure 6. As can be seen, most of the coefficient function estimates do not vary much from the constant term $\widehat{\beta}_j^{(0)}$. This differs from the corresponding Figure 5 in Richman and Wüthrich (2023), where the coefficient function estimates vary more dramatically. Further, the TVCM seems to produce coefficient function estimates that are closer to the GLM than the LocalGLMnet model from Richman and Wüthrich (2023). This is likely a consequence of the fact that the local GLM formulation comes with a native identification problem – there are likely many different local GLMs that provide drastically different coefficient function estimates but still produce similar predictions – a hypothesis that is supported by the loss results in Table 4. Without access to the individual predictions from the LocalGLMnet model from Richman and Wüthrich (2023), it is not possible to analyse this hypothesis further.

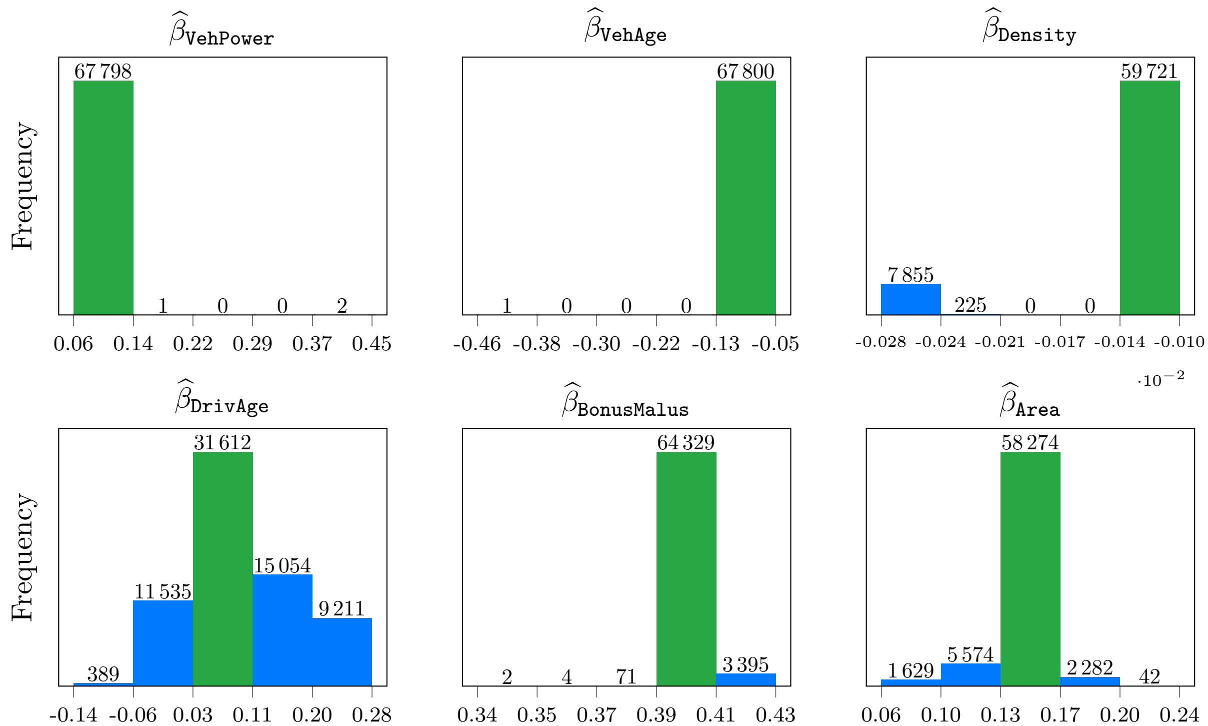

Figure 6: Histograms of the coefficient function estimates $\widehat{\beta}_j(\boldsymbol{x})$ for the TVCM estimates on the test set of the real data set for the different continuous predictive features x_j. The bin containing the constant term $\widehat{\beta}_j^{(0)}$ is coloured in green.

5 Conclusion

This paper introduces an interpretable tree-based VCM built using cyclic GBMs, following the ideas as those that underpin the neural network based LocalGLMnet from Richman and Wüthrich (2023), in an approach that is similar to that of the boosted VCM from Zhou and Hooker (2022). The cyclic GBM from Delong et al. (2023) allows for dimension-wise early stopping in the training of the regression coefficient functions, as well as the use of coefficient function-wise feature importance, which makes model tuning and interpretation easier than for the models in Richman and Wüthrich (2023) and Zhou and Hooker (2022). By using the dimension-wise feature importance scores it is also easy to identify potentially redundant features that could be removed from the modeling of the regression coefficient functions. This procedure is simpler than the corresponding model reduction techniques discussed for the LocalGLMnet in Richman and Wüthrich (2023). In Richman and Wüthrich (2023) splines are used to assess interaction effects, whereas the current analyses rely on feature importance scores. By using splines it is possible to obtain more granular information on *within*-feature dynamics; something which is not possible when using feature importance scores. There is, however, nothing that prevents us from analysing tree-based VCMs using splines.

The results of the tree-based VCM on simulated and real data shows that it outperforms GLM and GBM models for the same regression problems, and produces results in terms of out of sample loss that are comparable to those of the LocalGLMnet model. Further, additional analyses indicates that the resulting tree-based VCM produces a model structure that is similar to that of the LocalGLMnet model.

Concerning extensions, in Wang and Hastie (2014) the use of elastic nets are discussed, which is also possible to use for the current tree-based VCM.

Acknowledgements

The authors would like to thank Professor Mario V. Wüthrich for comments on an early draft as well as for providing the data from Richman and Wüthrich (2023). The authors would also like to thank Ronald Richman for discussions on the topic of this paper on an early stage. M. Lindholm acknowledges financial support from Stiftelsen Länsförsäkringsgruppens Forsknings- och Utvecklingsfond, grant P9/20 "Machine learning methods in non-life insurance". The `python` implementation of the special case of the VCM presented in Section 4 is available at `https://github.com/henningzakrisson/local-glm-boost` under the name `LocalGLMboost`.

References

Al-Shedivat, M., Dubey, A., and Xing, E. (2020). Contextual explanation networks. *The Journal of Machine Learning Research*, 21(1):7950–7993.

Alvarez Melis, D. and Jaakkola, T. (2018). Towards robust interpretability with self-explaining neural networks. *Advances in neural information processing systems*, 31.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and regression trees*. Belmont, Calif. Wadsworth Inc.

Bühlmann, P. L. (2002). Consistency for L2boosting and matching pursuit with trees and tree-type basis functions. In *Research report/Seminar für Statistik, Eidgenössische Technische Hochschule (ETH)*, volume 109. Seminar für Statistik, Eidgenössische Technische Hochschule (ETH).

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., Chen, K., Mitchell, R., Cano, I., Zhou, T., et al. (2015). Xgboost: extreme gradient boosting. *R package version 0.4-2*, 1(4):1–4.

Delong, Ł., Lindholm, M., and Zakrisson, H. (2023). On cyclic gradient boosting machines. *Available at SSRN 4352505*.

Duan, T., Anand, A., Ding, D. Y., Thai, K. K., Basu, S., Ng, A., and Schuler, A. (2020). Ngboost: Natural gradient boosting for probabilistic prediction. In *International conference on machine learning*, pages 2690–2700. PMLR.

Dutang, C., Charpentier, A., and Dutang, M. C. (2020). *CASdatasets: A Package of Insurance Datasets Provided by the Casualty Actuarial Society*. Accessed: 2023-11-06.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Hastie, T. and Tibshirani, R. (1993). Varying-coefficient models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 55(4):757–779.

Jørgensen, B. (1997). *The Theory of Dispersion Models*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30.

Lee, S. C. and Lin, S. (2018). Delta boosting machine with application to general insurance. *North American Actuarial Journal*, 22(3):405–425.

Lindholm, M., Lindskog, F., and Palmquist, J. (2023). Local bias adjustment, duration-weighted probabilities, and automatic construction of tariff cells. *Scandinavian Actuarial Journal*, pages 1–28.

Lindholm, M. and Nazar, T. (2023). On duration effects in non-life insurance pricing. *Available at SSRN 4474908*.

Marcinkevičs, R. and Vogt, J. E. (2023). Interpretable and explainable machine learning: A methods-centric overview with concrete examples. *WIREs Data Mining and Knowledge Discovery*, 13(3):e1493.

Nelder, J. A. and Wedderburn, R. W. (1972). Generalized linear models. *Journal of the Royal Statistical Society Series A: Statistics in Society*, 135(3):370–384.

Ohlsson, E. and Johansson, B. (2010). *Non-life insurance pricing with generalized linear models*, volume 174. Springer.

Richman, R. and Wüthrich, M. V. (2023). LocalGLMnet: interpretable deep learning for tabular data. *Scandinavian Actuarial Journal*, 2023(1):71–95.

Sundberg, R. (2019). *Statistical Modelling by Exponential Families*, volume 12 of *Institute of Mathematical Statistics Textbooks*. Cambridge University Press, Cambridge, UK.

Thompson, R., Dezfouli, A., and Kohn, R. (2023). The contextual lasso: Sparse linear models via deep neural networks. *arXiv preprint arXiv:2302.00878*.

Wang, J. C. and Hastie, T. (2014). Boosted varying-coefficient regression models for product demand prediction. *Journal of Computational and Graphical Statistics*, 23(2):361–382.

Wüthrich, M. V. and Merz, M. (2023). *Statistical foundations of actuarial learning and its applications*. Springer Nature.

Zhang, T. and Yu, B. (2005). Boosting with early stopping: Convergence and consistency. *The Annals of Statistics*, 33(4):1538–1579.

Zhou, Y. and Hooker, G. (2022). Decision tree boosted varying coefficient models. *Data Mining and Knowledge Discovery*, 36(6):2237–2271.

A Comments on regression coefficient functions in VCMs

A.1 Interpretation of regression coefficient functions in VCMs

Concerning the interpretation of the coefficient functions, consider the situation with an L^2-loss with identity link-function and assume that you know the correct $\boldsymbol{z}_j$s, $j = 1, \ldots, p$; here neglecting the constant intercept term. It then holds that

$$\begin{aligned}\beta_j^*(\boldsymbol{Z}_j) &= \frac{\mathrm{E}[X_j(Y - \sum_{k\neq j}\beta_k^*(\boldsymbol{Z}_k)X_k) \mid \boldsymbol{Z}_j]}{\mathrm{E}[X_j^2 \mid \boldsymbol{Z}_j]} \\ &= \mathrm{E}\left[\frac{X_j^2}{\mathrm{E}[X_j^2 \mid \boldsymbol{Z}_j]}\frac{Y - \sum_{k\neq j}\beta_k^*(\boldsymbol{Z}_k)X_k}{X_j} \;\middle|\; \boldsymbol{Z}_j\right] \end{aligned} \tag{19}$$

minimises

$$\mathrm{E}[(Y - \sum_{j=1}^{p}\beta_k^*(\boldsymbol{Z}_k)X_k)^2],$$

see (6) in Hastie and Tibshirani (1993) together with the surrounding discussion. Note that we use the notation β_j^* to stress that we assume an infinite amount of data, since (19) is based on population quantities. Not surprisingly, from (19) it is seen that the coefficient functions need to be estimated jointly. Moreover, from (19) it is seen that the $\beta_j^*(\boldsymbol{Z}_j)$s corresponds to weighted conditional expectations, with weights proportional to X_j^2. In particular, note that each $\beta_j^*(\boldsymbol{Z}_j)$ corresponds to the remaining conditional expectation to be explained after having adjusted for the effect of the coefficient functions $\beta_k^*(\boldsymbol{Z}_k), k \neq j$.

Further, note that if the true mean function $\mu(\boldsymbol{X}) = \mathrm{E}[Y \mid \boldsymbol{X}]$ is given by

$$\mu(\boldsymbol{X}) := \sum_{j=1}^{p}\gamma_j(\boldsymbol{Z}_j)X_j,$$

it directly follows from (19) that

$$\beta_j^*(\boldsymbol{Z}_j) = \gamma_j(\boldsymbol{Z}_j) + \mathrm{E}\left[\frac{X_j^2}{\mathrm{E}[X_j^2 \mid \boldsymbol{Z}_j]}\frac{\sum_{k\neq j}(\gamma_k(\boldsymbol{Z}_k) - \beta_k^*(\boldsymbol{Z}_k))X_k}{X_j} \;\middle|\; \boldsymbol{Z}_j\right]. \tag{20}$$

Thus, from (20) it follows that $\beta_j^*(\boldsymbol{Z}_j)$ is, a possibly noisy, estimator of $\gamma_j(\boldsymbol{Z}_j)$; see also the discussion in Section 4. This also connects to identifiability issues discussed in Remark 3. For the special case where $\gamma_j(\boldsymbol{Z}_j) \equiv \gamma_j$, it follows that (20) reduces to precisely $\beta_j^* = \gamma_j$, see the separate calculations in Appendix A.2.

As a final note, the above L^2-discussion can be directly generalised to the Tweedie family, by taking conditional expected values of the corresponding estimating equations, see e.g. (3.39) in Ohlsson and Johansson (2010).

A.2 Details on L^2 linear model special case of VCM

In the special case with a linear model we have that

$$Y = \sum_{j=1}^{p}\gamma_j X_j + \epsilon,$$

where ϵ is a random variable with mean 0 and variance σ^2. This means that if we know that $\beta_j^*(\boldsymbol{Z}_j) = \beta_j^*$ the conditional expectations in (20) are turned into unconditional expectations. That is, (20) can be simplified to

$$\begin{aligned}\beta_j^* &= \gamma_j + \sum_{k\neq j}(\gamma_k - \beta_k^*)\frac{\mathrm{E}[X_jX_k]}{\mathrm{E}[X_j^2]} \\ &= \gamma_j + \sum_{k\neq j}(\gamma_k - \beta_k^*)c_{j,k},\end{aligned}$$

which can be rephrased according to

$$\beta_j^* = \gamma_j + (\boldsymbol{\gamma} - \boldsymbol{\beta}^*)^\top \boldsymbol{c}_j,$$

where $(\boldsymbol{c}_j)_k = c_{j,k}$ for all $k \neq j$ and $(\boldsymbol{c}_j)_j = 0$. Further, by introducing the $p \times p$-matrix $\boldsymbol{C}$ with columns given by $\boldsymbol{c}_j$ it follows that

$$(\boldsymbol{\beta}^*)^\top = \boldsymbol{\gamma}^\top + (\boldsymbol{\gamma} - \boldsymbol{\beta}^*)^\top \boldsymbol{C},$$

which can be re-written according to

$$(\boldsymbol{\beta}^*)^\top (\boldsymbol{I} + \boldsymbol{C}) = \boldsymbol{\gamma}^\top (\boldsymbol{I} + \boldsymbol{C}),$$

where $\boldsymbol{I}$ is a $p \times p$ identity matrix. Thus, when knowing the form of the $\boldsymbol{Z}_j$s the relations given by (20) provides the solution $\boldsymbol{\beta}^* = \boldsymbol{\gamma}$, without any additional constraints on the $\boldsymbol{X}_j$s or $\boldsymbol{Z}_j$s, as it should.

A.3 Explicit L^2-calculations for the simulated example in Section 4.1

We will now analyse the simulated Gaussian example from Section 4.1 based on the discussion in Section 2.2 on L^2-losses that build on Hastie and Tibshirani (1993). The aim is to identify the correct basis of comparison for the estimated coefficient functions for the situation where the true $\mu(\boldsymbol{x})$-function is known and satisfies the local GLM, or varying-coefficient regression formulation, from (5) in Section 2.2.

From Table 1 it follows that $\boldsymbol{Z}_1 = \boldsymbol{Z}_7 = \boldsymbol{Z}_8 = \{\emptyset\}$, $\boldsymbol{Z}_2 = X_2$, $\boldsymbol{Z}_3 = X_3$, $\boldsymbol{Z}_4 = X_5$, $\boldsymbol{Z}_5 = X_4$, and $\boldsymbol{Z}_6 = X_5$. Further, from the construction of the example in Section 4.1 it follows that all X_js are independent except for X_2 and X_8. Combining all of the above results in that (20) reduces to

$$\begin{aligned}
\beta_1^*(\boldsymbol{Z}_1) &= \beta_1, \\
\beta_2^*(\boldsymbol{Z}_2) &= \beta_2(X_2) + c_2/X_2, \\
\beta_3^*(\boldsymbol{Z}_3) &= \beta_3(X_3) + c_3/X_3, \\
\beta_4^*(\boldsymbol{Z}_4) &= \beta_4(X_5) + c_4 X_5, \\
\beta_5^*(\boldsymbol{Z}_5) &= \beta_5(X_4) + c_5 X_4, \\
\beta_6^*(\boldsymbol{Z}_6) &= \beta_6(X_5) + c_6 X_6, \\
\beta_7^*(\boldsymbol{Z}_7) &= \beta_7, \\
\beta_8^*(\boldsymbol{Z}_8) &= \beta_8,
\end{aligned}$$

where c_j are constants, where a perfect fit corresponds to that all $c_j = 0$. Consequently, a perfect fit when assuming perfect a priori knowledge of the $\boldsymbol{Z}_j$s and the functional form of $\mu(\boldsymbol{X})$ agrees with the local GLM, or varying-coefficient regression structure, from (5), retrieves the original coefficient functions. That is, ideally $\beta_j^*(\boldsymbol{Z}_j) = \beta_j(\boldsymbol{Z}_j)$, $j = 1, \ldots, p$. This motivates using the $\beta_j(\boldsymbol{Z}_j)$s from Table 1 as a basis for comparison with the corresponding coefficient functions from the cyclically boosted VCM in Section 4.1.