A Tweedie Compound Poisson Model in Reproducing Kernel Hilbert Space

Yi Lian [*], Yi Yang [†], Boxiang Wang [‡], Peng Shi [§], Robert William Platt [¶]

August 30, 2021

Abstract

Tweedie models can be used to handle non-negative continuous data with a probability mass at zero. There have been wide applications in natural science, healthcare research, actuarial science and other fields. The performance of existing Tweedie models can be limited by today's complex data problems with challenging characteristics such as high-dimensionality, nonlinear effects, high-order interactions. Motivated by these challenges, we propose a kernel Tweedie model with integrated variable selection. The non parametric nature of the the proposed method along with the abundance of available kernel functions provides much needed modeling flexibility and capability. The resulting sparsity due to the variable selection also improves the interpretability and the prediction accuracy. We perform extensive simulation studies to justify the prediction and variable selection accuracy of our method, and demonstrate the applications in ratemaking and loss-reserving in general insurance. The model is implemented in an efficient and user-friendly `R` package.

Keywords: Tweedie compound Poisson models; Reproducing kernel Hilbert space; Sparse kernel methods; Insurance claims data; Loss-reserving; Ratemaking; Zero inflated data.

[*]Department of Epidemiology, Biostatistics and Occupational Health, McGill University
[†]Department of Mathematics and Statistics, McGill University (yi.yang6@mcgill.ca)
[‡]Department of Statistics and Actuarial Science, University of Iowa
[§]Risk and Insurance Department, Wisconsin School of Business, University of Wisconsin-Madison
[¶]Corresponding author, Department of Epidemiology, Biostatistics and Occupational Health, McGill University

1 Introduction

The Tweedie compound Poisson distribution (Tweedie, 1984), which belongs to the exponential dispersion (ED) family (Jorgensen, 1997), has been extensively studied and widely used in applications. Examples include the modeling of the total precipitation in meteorology and climatology (Dunn, 2004; Hasan and Dunn, 2011; Dons et al., 2016; Dzupire et al., 2018), the biomass of a certain species in ecology (El-Shaarawi et al., 2011; Foster and Bravington, 2013), the total catch in fishery (Shono, 2008), and the aggregate losses in insurance (Smyth and Jorgensen, 2002; Shi et al., 2016) . The medical research community has also found its use in various ways (Moshitch and Nelken, 2014; Kurz, 2017; Islam et al., 2021).

A number of statistical models has been developed based on the Tweedie compound distribution, and the learning in these Tweedie models range across a wide spectrum – from highly structural approaches such as Tweedie GLMs (TGLM; Jørgensen and de Souza 1994; Smyth and Jorgensen 2002; Shi et al. 2016) and Tweedie mixed models (Zhang, 2013; Yang et al., 2019) to more flexible nonparametric approaches such as regression trees (Yang et al., 2018; Lee and Lin, 2018; Zhou et al., 2020) and neural networks (Blier-Wong et al., 2021). The TGLM relates the conditional mean of a Tweedie response to a linear function of predictors through a link function. The regularized version of the TGLM (Qian et al., 2016; Fontaine et al., 2020) copes well with high-dimensional predictors. To improve the flexibility of the TGLM, the Tweedie generalized additive model (TGAM) (Hastie and Tibshirani, 1990; Wood, 2011) uses splines to model nonlinear functional components of predictors. Although allowing for the specification of nonlinear relationship between the response and predictors, the TGAM is limited to additive nonlinear effects. To further mitigate the risk of model mis-specification, Yang et al. (2018) and Zhou et al. (2020) proposed a tree-based gradient boosting algorithm (Friedman, 2001) for the Tweedie model (TDboost). It features the ability to handle discontinuity, nonlinear effects, and high-order interactions of the predictors. However, the performance of the TDboost algorithm can be compromised when the dimension of the training data is high and/or the size is small.

In this paper, we propose a fully nonparametric Tweedie model in reproducing kernel Hilbert

spaces (RKHS). It is well known that the Tikhonov regularized models in RKHS have deep connections with the support vector machine (Vapnik, 2013). The proposed method enjoys several key features: First, the method is data-driven in that the model structure is fully learned from the data, and hence can mitigate potential model mis-specification; Second, the model can capture complex nonlinear relationships and high-order interactions among variables. This is due to the flexible choices of corresponding kernel functions at our disposal. Researchers have conducted extensive research on the use and combined use of kernel functions to accommodate complex relationships in data (Rasmussen, 2003; Duvenaud, 2014); Third, the method allows for automatic variable selection. In the high-dimensional setting where the number of predictors is close to or larger than the sample size, most predictors are usually non-informative or noisy, and the error accumulated when estimating the noise variables can lead to bad prediction performance (Fan and Fan, 2008). Inspired by Allen (2013), Yang et al. (2016a) and Chen et al. (2018), we incorporate variable selection using weighted kernels along with the sparsity-inducing regularization. Specifically, we introduce variable weights to the kernel as parameters of interest, and employ regularization to introduce sparsity among weights associated with each variable. The method allows us to integrate variable selection with parameter estimation.

We demonstrate the performance of the proposed method in both simulation studies and real data analyses. In the simulation, we test the predictive accuracy of our model in the presence of nonlinear effects and high-order interactions of predictors. The performance of variable selection is also investigated. In the real data analysis, we examine the applications of the Tweedie model in two key insurance operations, ratemaking and claims reserving. The former concerns the determination of the premium for future risks, and the latter concerns the prediction of outstanding liability from existing risks. The outcome of interest is the aggregate loss in both cases, with the focus being an individual policy in ratemaking and a portfolio of policies in claims reserving. The Tweedie random variable, constructed by a random sum, has a semi-continuous probability distribution with a non-zero probability at zero along with a positive continuous support, which makes it a natural choice for insurance applications. The Tweedie models have received extensive attention by the

actuarial community and insurance practitioners, see Jørgensen and de Souza (1994), Smyth and Jorgensen (2002), Shi et al. (2016), Halder et al. (2019) for ratemaking applications and Peters et al. (2009), Shi (2014), Taylor (2019) for claims reserving applications. We show the superior performance of the proposed method to the competing Tweedie models in the two cases.

The rest of the paper is organized as follows. In Section 2, we formally introduce the Tweedie compound Poisson distribution, lay out the kernel Tweedie model, and provide the intuition and formulation of the integrated variable selection component. Section 3 proposes the optimization algorithms for model learning and analyzes the convergence property. We discuss various aspects of the model implementation, including the profile likelihood approach for estimating the additional parameters in the Tweedie model and the procedures to improve the computational efficiency of the proposed algorithms. In Section 4, we test the prediction accuracy of the model and examine the performance of variable selection. Section 5 showcases the performance of our model in the aforementioned ratemaking and loss-reserving applications. Section 6 concludes the paper. Technical details and additional empirical results are provided in the Appendix.

2 Methodology

Assume N to be a Poisson random variable $N \sim \text{Poisson}(v\lambda)$ associated with a weight of observation v, and conditional on N, for $d = 0, 1, \ldots, N$, Z_d's are $i.i.d$ gamma distributed $Z_d \sim \text{Gamma}(\alpha, \gamma)$. Define Y as the conditional sum of N $i.i.d.$ gamma random variables standardized by the weight:

$$Y = \begin{cases} 0 & \text{if } N = 0 \\ (Z_1 + Z_2 + \cdots + Z_N)/v & \text{if } N = 1, 2, \ldots. \end{cases} . \tag{1}$$

The distribution of Y is referred to as the compound Poisson distribution. For example, in insurance applications, N is the number of claims for a risk, Z_d is the amount of losses for the d-th claim, and v is the exposure (e.g. duration of the policy in years), thus Y represents the aggregate loss amount per

unit at risk (Jørgensen and de Souza, 1994; Shi, 2016; Smyth and Jorgensen, 2002). Note that $Y = 0$ if and only if $N = 0$, thus Y has a probability mass at 0, i.e. $\Pr(Y = 0) = \Pr(N = 0) = \exp(-v\lambda)$. Additionally, Y conditional on $N = n$ follows a gamma distribution with shape $n\alpha$ and scale γ/v . It has been shown that the compound Poisson distribution is related to a special class of the ED family known as the Tweedie distribution (Tweedie, 1984; Jørgensen, 1987; Smyth, 1996). The density function of the ED family follows

$$g\left(y|\theta,\varphi\right) = a\left(y,\varphi\right)\exp\left\{\frac{y\theta-\kappa(\theta)}{\varphi}\right\}, \tag{2}$$

with the natural parameter $\theta \in \mathbb{R}$ and the dispersion parameter $\varphi \in \mathbb{R}^+$. The normalizing function $a(\cdot)$ and the cumulant function $\kappa(\cdot)$ are both known. By the property of the ED family, we have that

$$\mu \equiv \mathbb{E}(Y) = \dot{\kappa}(\theta), \qquad \text{Var}(Y) = \varphi\ddot{\kappa}\left(\dot{\kappa}^{-1}(\mu)\right) = \varphi\ddot{\kappa}(\theta), \tag{3}$$

where $\dot{\kappa}\left(\theta\right)$ and $\ddot{\kappa}\left(\theta\right)$ are the first and second order derivative of $\kappa(\theta)$, respectively. For the Tweedie distribution, θ, $\kappa(\theta)$ and φ have the specific forms:

$$\theta = \frac{\mu^{1-\rho}}{1-\rho}, \quad \kappa(\theta) = \frac{\mu^{2-\rho}}{2-\rho}, \quad \dot{\kappa}(\theta) = \mu, \quad \ddot{\kappa}(\theta) = \mu^{\rho}, \quad \varphi = \phi/v, \tag{4}$$

for some index parameter (also called power parameter) $\rho \in (1, 2)$ and the dispersion parameter $\phi \in \mathbb{R}^+$. Using (2), the density of the Tweedie distribution $\text{Tw}(\mu, \phi/v, \rho)$ can be written as

$$g(y|\mu,\phi,\rho) = a(y,\phi,\rho)\exp\left\{\frac{v}{\phi}\left(\frac{y\mu^{1-\rho}}{1-\rho}-\frac{\mu^{2-\rho}}{2-\rho}\right)\right\}, \tag{5}$$

where the exact form of $a(\cdot)$ can be found in Section 3 of Yang et al. (2018). The mean and variance relationship of the Tweedie distribution becomes $\text{Var}(Y) = \phi/v \cdot \mu^{\rho}$.

The Tweedie family of distributions includes several distributions, specified by the index parameter ρ (Tweedie, 1984; Jørgensen, 1987). For example, it degenerates to the normal distribution when $\rho = 0$, to the Poisson distribution when $\rho = 1$, and to the gamma distribution when $\rho = 2$. When

$1 < \rho < 2$, it has been shown that the Tweedie distribution is equivalent to the aforementioned compound Poisson distribution, if we reparameterize $(\lambda, \alpha, \gamma)$ by (μ, ϕ, ρ) as following,

$$\lambda = \frac{1}{\phi}\frac{\mu^{2-\rho}}{2-\rho}, \quad \alpha = \frac{2-\rho}{\rho-1}, \quad \gamma = \phi(\rho-1)\mu^{\rho-1}. \tag{6}$$

From now on we will refer to the compound Poisson distribution as the Tweedie distribution or the Tweedie model for convenience.

2.1 Kernel learning of Tweedie compound Poisson models

Consider a dataset $\mathbf{D} = \{(y_i, \mathbf{x}_i, v_i)\}_{i=1}^n$ that contains n independent observations, where, for the i-th observation, $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})^\top$ is a p-dimensional vector of exogenous predictors and y_i is the outcome variable observed under a exposure of v_i. Then Y_i under duration v_i follows $Y_i/v_i \sim \text{Tw}(\mu_i, \phi/v_i, \rho)$. We model the mean μ_i as a function of the predictors $\mathbf{x}_i \in \mathbb{R}^p$ using a log link function as $\log(\mu_i) = f(\mathbf{x}_i)$, where f belongs to a function space $\mathcal{F}$. For instance, $f(\mathbf{x}_i) = \mathbf{x}_i^\top \boldsymbol{\beta}$ corresponds to the Tweedie GLM (Jørgensen and de Souza, 1994). Using the aforementioned model setup, the log-likelihood function can be written as

$$\ell(f(\cdot), \phi, \rho|\mathbf{D}) = \sum_{i=1}^n \frac{v_i}{\phi}\left\{ y_i \frac{\exp\big[(1-\rho)f(\mathbf{x}_i)\big]}{1-\rho} - \frac{\exp\big[(2-\rho)f(\mathbf{x}_i)\big]}{2-\rho} \right\} . \tag{7}$$

We propose a nonparametric Tweedie model, in which the function f is chosen from a reproducing kernel Hilbert space $\mathcal{H}_K$. To learn the function f from the data $\mathbf{D} = \{(y_i, \mathbf{x}_i, v_i)\}_{i=1}^n$, we minimize the following penalized negative log-likelihood function

$$\hat{f}(\cdot) = \underset{f \in \mathcal{H}_K}{\arg\min} \left[-\ell(f(\cdot), \phi, \rho|\mathbf{D}) + \lambda \|f\|_{\mathcal{H}_K}^2 \right], \tag{8}$$

where $\|f\|_{\mathcal{H}_K}^2$ is a generalized Tikhonov regularization defined in the Hilbert space. The representer theorem (Wahba, 1990) shows that f can be parameterized by a combination of kernel functions

$$f(\mathbf{x}_i) = \alpha_0 + \sum_{i'=1}^{n} \alpha_{i'} K(\mathbf{x}_i, \mathbf{x}_{i'}) = \alpha_0 + \mathbf{K}_i^\top \boldsymbol{\alpha}, \tag{9}$$

where $\mathbf{K}_i$ is the i-th row of the $n \times n$ kernel matrix $\mathbf{K} = (K(\mathbf{x}_i, \mathbf{x}_{i'}))_{n\times n}$, generated by a positive definite kernel function $K(\cdot, \cdot)$, and α_0 and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)^\top$ are the coefficients. Consequently, (8) is equivalent to

$$(\widehat{\alpha}_0, \widehat{\boldsymbol{\alpha}}) = \underset{\alpha_0, \boldsymbol{\alpha}}{\arg\min} \left\{ -\sum_{i=1}^{n} \frac{v_i}{\phi} \left(\frac{y_i e^{(1-\rho)(\alpha_0 + \mathbf{K}_i^\top \boldsymbol{\alpha})}}{1-\rho} - \frac{e^{(2-\rho)(\alpha_0 + \mathbf{K}_i^\top \boldsymbol{\alpha})}}{2-\rho} \right) + \lambda \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha} \right\}, \tag{10}$$

which minimizes a smooth convex function of $(\alpha_0, \boldsymbol{\alpha})$. We refer to this model as the *Ktweedie* model. The algorithms for solving (10) will be discuss in Section 3.1.

2.2 Integrated variable selection via weighted kernels

We further extend the Ktweedie model to integrate automatic variable selection. Several methods have considered weighting variables within the kernel to achieve variable selection (Weston et al., 2000; Grandvalet and Canu, 2002; Gilad-Bachrach et al., 2004; Li et al., 2005; Argyriou et al., 2006; Cao et al., 2007), where weights are found using a separate procedure and variable selection is not integrated in the estimation. Other approaches such as COSSO (Lin et al., 2006) can simultaneously estimate the nonlinear functional component and select important variables, but they are limited to the additive models. Inspired by the work of Allen (2013) and Chen et al. (2018), we achieve variable selection in the Ktweedie model through a certain sparse penalization on the variable weights in the kernel function. This additional feature can potentially improve the interpretability of the result and the prediction accuracy. Specifically, we modify the objection function (10) in two aspects: First, variable weights are used in the kernel matrix such that:

$$f(\mathbf{x}_i) = \alpha_0 + \sum_{i'=1}^{n} \alpha_{i'} K(\mathbf{w} \odot \mathbf{x}_i, \mathbf{w} \odot \mathbf{x}_{i'}), \tag{11}$$

where $\mathbf{w} \in [0,1]^p$ is a p-dimensional weight vector that controls the contribution of each variable in $\mathbf{x}$ and "$\odot$" denotes element-wise multiplication. That is, the kernel matrix $\mathbf{K}$ in (9) is replaced by the weighted kernel matrix $\mathbf{K}(\mathbf{w})$ with the following form,

$$\mathbf{K}(\mathbf{w}) = \begin{bmatrix} K(\mathbf{w}\odot\mathbf{x}_1, \mathbf{w}\odot\mathbf{x}_1) & \cdots & K(\mathbf{w}\odot\mathbf{x}_1, \mathbf{w}\odot\mathbf{x}_n) \\ \vdots & \ddots & \vdots \\ K(\mathbf{w}\odot\mathbf{x}_n, \mathbf{w}\odot\mathbf{x}_1) & \cdots & K(\mathbf{w}\odot\mathbf{x}_n, \mathbf{w}\odot\mathbf{x}_n) \end{bmatrix}. \tag{12}$$

Second, a sparsity-inducing regularization is applied on the weights $\mathbf{w}$ (Allen, 2013). The resulting modified objective function is

$$\begin{aligned}(\widehat{\alpha}_0, \widehat{\boldsymbol{\alpha}}, \widehat{\mathbf{w}}) = \underset{\alpha_0,\boldsymbol{\alpha},\mathbf{w}}{\arg\min} \Bigg\{ & -\sum_{i=1}^{n} \frac{v_i}{\phi} \left(\frac{y_i e^{(1-\rho)(\alpha_0 + \mathbf{K}(\mathbf{w})_i^\top \boldsymbol{\alpha})}}{1-\rho} - \frac{e^{(2-\rho)(\alpha_0 + \mathbf{K}(\mathbf{w})_i^\top \boldsymbol{\alpha})}}{2-\rho} \right) \\ & + \lambda_1 \boldsymbol{\alpha}^\top \mathbf{K}(\mathbf{w}) \boldsymbol{\alpha} + \lambda_2 \mathbf{1}^\top \mathbf{w} \Bigg\} \\ & \text{s.t. } w_j \in [0,1],\ j = 1, \ldots, p. \end{aligned} \tag{13}$$

We refer to the model (13) as the *SKtweedie* model, and discuss its fitting in Section 3.2. The regularization term $\mathbf{1}^\top \mathbf{w}$ together with the constraints $w_j \in [0,1]$ in (13) can induce sparsity to $\mathbf{w}$. As a result, when the estimated weight of the j-th variable is zero, the contribution of the j-th variable is removed from the model. To see this, let us consider a simplistic two-dimensional example for some loss function $\ell(w_1, w_2)$ with a constrained form of the penalization

$$(\widehat{w}_1, \widehat{w}_2) = \underset{w_1,w_2}{\arg\min}\, \ell(w_1, w_2),\ \text{s.t. } w_1 + w_2 < t \text{ and } w_1, w_2 \in [0,1]. \tag{14}$$

The constraint region of (14), depending on the value of t, varies among three different scenarios (a), (b) and (c) as shown in Figure 1. In particular, when $t \leq 1$, the constraint region becomes a right triangle that is similar to a constraint region induced by an ℓ_1 norm.

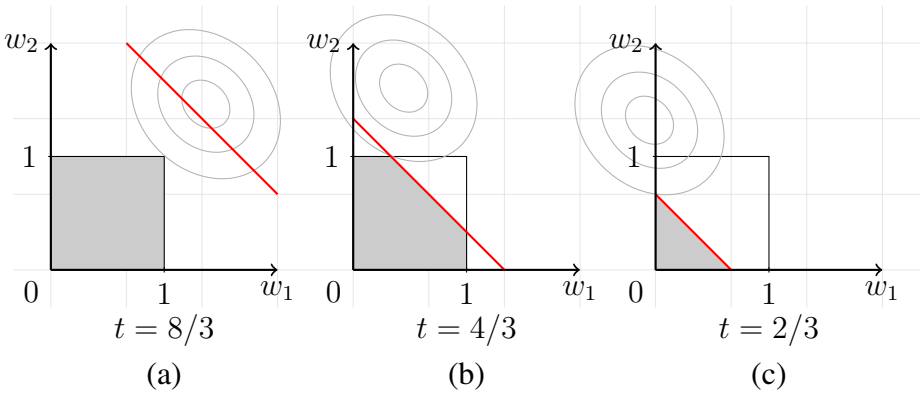

Figure 1: Geometric interpretation of the regularization term $\mathbf{1}^\top \mathbf{w}$. The shaded areas are the constraint regions $w_1, w_2 \in [0, 1]$ and $w_1 + w_2 < t$ for different t's. The ellipses are the contours of the loss function as a function of w_1 and w_2. Sparsity is induced in (c).

3 Computation

In this section, we introduce the algorithms for optimizing the Ktweedie and SKtweedie models described in Sections 2.1 and 2.2 respectively. For the Ktweedie model with the objective function (10), we adopt an inverse Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm (Broyden, 1970; Fletcher, 1970; Goldfarb, 1970; Shanno, 1970; Nocedal and Wright, 2006). For the SKtweedie model with the objective function (13), we propose an alternating optimization method. For this moment, we assume that both ρ and ϕ are given, and we discuss the procedure for estimating their values in Section 3.3.

3.1 Fitting the Ktweedie model

We propose an inverse BFGS algorithm, which belongs to the Quasi-Newton methods, to solve the optimization problem (10). Solving the Ktweedie model requires additional considerations for the intercept α_0, which is discussed in Section 3.3. Here we first solve a simpler variant: $\widehat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} g(\boldsymbol{\alpha})$, where $g(\boldsymbol{\alpha})$ is the objective function in (10) without the intercept. We use superscript (k) to indicate k-th iteration of our algorithm. We first update $\boldsymbol{\alpha}^{(k+1)}$ by

$$\boldsymbol{\alpha}^{(k+1)} = \boldsymbol{\alpha}^{(k)} - t^{(k)} \mathbf{B}^{(k)} \nabla g(\boldsymbol{\alpha}^{(k)}), \tag{15}$$

where $t^{(k)}$ is the step size, $\mathbf{B}^{(k)}$ is an approximate inverse Hessian matrix of the objective $g(\boldsymbol{\alpha})$ and $\nabla g(\boldsymbol{\alpha})$ is the gradient:

$$\nabla_{\alpha_j} g(\boldsymbol{\alpha}) = \frac{1}{n}\sum_{i=1}^{n} \frac{v_i}{\phi}\left(-y_i K_{ij} e^{(1-\rho)\mathbf{K}_i^\top \boldsymbol{\alpha}} + K_{ij} e^{(2-\rho)\mathbf{K}_i^\top \boldsymbol{\alpha}}\right) + 2\lambda \mathbf{K}_j^\top \boldsymbol{\alpha},\ j = 1, \ldots, p.$$

Given $\alpha^{(k+1)}$, we update B by

$$\mathbf{B}^{(k+1)} = \left(\mathbf{I}_n - \frac{\mathbf{s}^{(k)}\mathbf{z}^{(k)\top}}{\mathbf{z}^{(k)\top}\mathbf{s}^{(k)}}\right)\mathbf{B}^{(k)}\left(\mathbf{I}_n - \frac{\mathbf{z}^{(k)}\mathbf{s}^{(k)\top}}{\mathbf{z}^{(k)\top}\mathbf{s}^{(k)}}\right) + \frac{\mathbf{s}^{(k)}\mathbf{s}^{(k)\top}}{\mathbf{z}^{(k)\top}\mathbf{s}^{(k)}}, \tag{16}$$

where

$$\mathbf{s}^{(k)} = \boldsymbol{\alpha}^{(k+1)} - \boldsymbol{\alpha}^{(k)} \text{ and } \mathbf{z}^{(k)} = \nabla g(\boldsymbol{\alpha}^{(k+1)}) - \nabla g(\boldsymbol{\alpha}^{(k)}). \tag{17}$$

The update of $\boldsymbol{\alpha}$ and $\mathbf{B}$ is repeated until the convergence of $g(\boldsymbol{\alpha})$ or $\boldsymbol{\alpha}$. The details of the algorithm are summarized in Algorithm 1. The appropriate step size $t^{(k)}$ in (15) is chosen by using a bisection line-search summarized in Algorithm 4 (Appendix A.1) so as to satisfy the *Wolfe conditions* (Wolfe, 1971)

$$\begin{cases} g(\boldsymbol{\alpha} + t\mathbf{p}) \leq g(\boldsymbol{\alpha}) + c_1 t \nabla g(\boldsymbol{\alpha})^\top \mathbf{p} & \text{Condition 1} \\ \nabla g(\boldsymbol{\alpha} + t\mathbf{p})^\top \mathbf{p} \geq c_2 \nabla g(\boldsymbol{\alpha})^\top \mathbf{p} & \text{Condition 2} \end{cases}, \tag{18}$$

where $\mathbf{p} = -\mathbf{B}\nabla g(\boldsymbol{\alpha})$, $c_1 \in (0, 1)$ and $c_2 \in (c_1, 1)$ are some constants. Condition 1 is commonly referred to as the *Armijo condition* (Armijo, 1966) and Condition 2 is called the *curvature condition* (Nocedal and Wright, 2006). The two conditions serve as the upper and lower bounds for the step size t that warrants a reasonable progress.

There is a convergence guarantee for the proposed BFGS algorithm:

Theorem 1 (Global Convergence of BFGS)**.** *The updating sequence* $\{\boldsymbol{\alpha}^{(k)}\}$ *generated by the BFGS update* (15) *converges to the minimizer* $\boldsymbol{\alpha}^*$ *of the objective function* $g(\boldsymbol{\alpha})$ *at a superlinear rate.*

Proof. The detailed proof is given in Appendix B. □

Algorithm 1: (Inverse) BFGS algorithm for Ktweedie

Input: $\mathbf{K}, \mathbf{y}, \lambda$

Output: $\widehat{\boldsymbol{\alpha}}$

Initialization: $k = 0, \mathbf{B}^{(0)} = \mathbf{I}_n, \boldsymbol{\alpha}^{(0)}$;

repeat BFGS loop

  $\mathbf{p}^{(k)} = -\mathbf{B}^{(k)} \nabla g(\boldsymbol{\alpha}^{(k)})$

  call Algo. 4 to find step size $t^{(k)}$

    if *cannot find proper* $t^{(k)}$ **then** exit;

  $\mathbf{s}^{(k)} = t^{(k)} \mathbf{p}^{(k)}$

  $\boldsymbol{\alpha}^{(k+1)} = \boldsymbol{\alpha}^{(k)} + \mathbf{s}^{(k)}$

  $\mathbf{z}^{(k)} = \nabla g(\boldsymbol{\alpha}^{(k+1)}) - \nabla g(\boldsymbol{\alpha}^{(k)})$

  $\mathbf{B}^{(k+1)} = \left(\mathbf{I}_n - \frac{\mathbf{s}^{(k)} \mathbf{z}^{(k)\top}}{\mathbf{z}^{(k)\top} \mathbf{s}^{(k)}}\right) \mathbf{B}^{(k)} \left(\mathbf{I}_n - \frac{\mathbf{z}^{(k)} \mathbf{s}^{(k)\top}}{\mathbf{z}^{(k)\top} \mathbf{s}^{(k)}}\right) + \frac{\mathbf{s}^{(k)} \mathbf{s}^{(k)\top}}{\mathbf{z}^{(k)\top} \mathbf{s}^{(k)}}$

  $k := k + 1$

until *convergence*;

$\widehat{\boldsymbol{\alpha}} = \boldsymbol{\alpha}^{(k)}$

In practice, due to a potential precision loss, the bisection line-search in the BFGS may not be able to find a proper step size t. To fix this issue, we make the algorithm transition to a standard gradient descent with the backtracking line-search in Algorithm 5 (Appendix A.2) when the bisection line-search in the BFGS fails.

3.2 Fitting the SKtweedie model

We propose an alternating optimization method (Algorithm 3) to solve the objective function (13) in Section 2.2. The algorithm alternates between the model parameters $\boldsymbol{\alpha}$ and the weights $\mathbf{w}$ to perform joint estimation of the two, which achieves simultaneous estimation and variable selection. Specifically, in each outer loop iteration m, the inner $\mathbf{w}$-loop updates $\mathbf{w}^{(m)}$ to $\mathbf{w}^{(m+1)}$ with $\boldsymbol{\alpha}$ fixed at $\boldsymbol{\alpha}^{(m)}$, then the inner $\boldsymbol{\alpha}$-loop updates $\boldsymbol{\alpha}^{(m)}$ to $\boldsymbol{\alpha}^{(m+1)}$ using the new weights $\mathbf{w}^{(m+1)}$. We run the outer loops until convergence.

In the inner $\mathbf{w}$-loops, the gradient descent with backtracking line-search is used to update the weights $\mathbf{w}$ (Algorithm 2). Updated weights are projected to the interval $[0, 1]$ by the operation $\text{proj}(w_j) = \min(\max(w_j, 0), 1),\ j = 1, \ldots, p$, due to the restrictions. Denote the k-th update of w_j

within the m-th outer loop iteration by $w_j^{(m,k)}$. Then the inner $\mathbf{w}$-loop update has the form

$$w_j^{(m,k+1)} = \text{proj}\left(w_j^{(m,k)} - t^{(m,k)} \cdot \nabla_{w_j} g(\boldsymbol{\alpha}^{(m)}, \mathbf{w}^{(m,k)})\right), \tag{19}$$

where $\nabla_{w_j} g(\boldsymbol{\alpha}^{(m)}, \mathbf{w}^{(m,k)})$ denotes the gradient of the objective function (13) with respect to w_j, and its specific form is provided in Appendix C.

The details of the alternating optimization are given in Algorithm 3.

Algorithm 2: Gradient descent for weight

Input: $\mathbf{X}, \mathbf{y}, \lambda_1, \lambda_2, \boldsymbol{\alpha}^{(m)}, \mathbf{w}^{(m)}$
Output: $\mathbf{w}^{(m+1)}$

Initialization: $k = 0, \mathbf{w}^{(m,0)} = \mathbf{w}^{(m)}$;
repeat gradient descent loop
  Generate new kernel matrix $\mathbf{K}(\mathbf{w}^{(m,k)})$ as defined in (12)
  call Algo. 5 to find step size $t^{(m,k)}$
  for $j = 1, \ldots, p$ **do**
    Compute $w_j^{(m,k+1)}$ using (19)
  end
  $k := k + 1$
  if $\mathbf{w}^{(m,k+1)} = \mathbf{0}_p$ **then** exit;
until *convergence*;
$\mathbf{w}^{(m+1)} = \mathbf{w}^{(m,k)}$

3.3 Implementation details

Profile likelihood. As mentioned in Section 2.1, although the primary interest is the estimation of μ in the Tweedie model, we also need to estimate ρ and ϕ in order to characterize the variance of Y_i through the mean-variance relationship $\text{Var}(Y_i) = \phi\mu_i^\rho$ in (3). Following Dunn and Smyth (2005), we use the profile likelihood to estimate ρ and ϕ. It is straightforward to see from (5) that the estimation of μ does not depend on ϕ. Taking advantage of this fact, for any given ρ, we can estimate

Algorithm 3: Alternating Optimization Algorithm for the SKtweedie

Input: $\mathbf{X}, \mathbf{y}, \lambda_1, \lambda_2$
Output: $\widehat{\boldsymbol{\alpha}}, \widehat{\mathbf{w}}$

Initialization: $\boldsymbol{\alpha}^{(0)} = \mathbf{0}_n, \mathbf{w}^{(1)} = \mathbf{1}_p, m = 1$;
call Algo. 1 with $\boldsymbol{\alpha}$ initialized at $\boldsymbol{\alpha}^{(0)}$
    $\boldsymbol{\alpha}^{(1)} = \arg\min_{\boldsymbol{\alpha}} g(\boldsymbol{\alpha}, \mathbf{w}^{(1)})$
repeat outer loop
    call Algo. 2 with $\mathbf{w}$ initialized at $\mathbf{w}^{(m)}$
        $\mathbf{w}^{(m+1)} = \arg\min_{\mathbf{w}} g(\boldsymbol{\alpha}^{(m)}, \mathbf{w})$ s.t. $\mathbf{w} \in [0,1]^p$
    if $\mathbf{w}^{(m+1)} = \mathbf{0}_p$ **then** exit;
    $\mathbf{K} = \mathbf{K}(\mathbf{w}^{(m+1)})$ as defined in 12
    call Algo. 1 with $\boldsymbol{\alpha}$ initialized at $\boldsymbol{\alpha}^{(m)}$
        $\boldsymbol{\alpha}^{(m+1)} = \arg\min_{\boldsymbol{\alpha}} g(\boldsymbol{\alpha}, \mathbf{w}^{(m+1)})$
    $m := m + 1$
until *convergence*;
$\widehat{\boldsymbol{\alpha}} = \boldsymbol{\alpha}^{(m)}$
$\widehat{\mathbf{w}} = \mathbf{w}^{(m)}$

μ using the estimators for $(\alpha_0, \boldsymbol{\alpha})$ in $\mu(\rho) = e^{\alpha_0 + \mathbf{K}_i^\top \boldsymbol{\alpha}}$ that minimizes (10) in the Ktweedie model. Denote by $(\widehat{\alpha}_0(\rho), \widehat{\boldsymbol{\alpha}}(\rho))$ and $\widehat{\mu}(\rho) = e^{\widehat{\alpha}_0(\rho) + \mathbf{K}_i^\top \widehat{\boldsymbol{\alpha}}(\rho)}$ the estimators for $(\alpha_0, \boldsymbol{\alpha})$ and $\mu(\rho)$ respectively for the given ρ. Conditioning on ρ and $\widehat{\mu}(\rho)$, the likelihood function in (7) becomes a univariate function of ϕ. The optimal $\widehat{\phi}$ can then be obtained by using a combination of golden section search and successive parabolic interpolation (Brent, 2013). We estimate the optimal $\widehat{\mu}(\rho)$ and $\widehat{\phi}(\rho)$ for a equally spaced sequence of ρ's of length l on the interval $(1, 2)$, and choose the optimal $\widehat{\rho}$ that gives the maximum profile likelihood

$$\widehat{\rho} = \underset{\rho \in \{\rho_1, \rho_2, \ldots, \rho_l\}}{\arg\max} \ \ell(\widehat{\mu}(\rho), \widehat{\phi}(\rho), \rho). \tag{20}$$

The resulting estimates of (μ, ϕ, ρ) from the profile likelihood are $(\widehat{\mu}(\widehat{\rho}), \widehat{\phi}(\widehat{\rho}), \widehat{\rho})$, which gives us an estimated variance of Y_i as $\widehat{\phi}(\widehat{\rho})[\widehat{\mu}_i(\widehat{\rho})]^{\widehat{\rho}}$. However, if the main goal of the data analysis is only to predict the response, the profile likelihood procedure is unnecessary and we can simply estimate μ with any arbitrary ρ, e.g. $\rho = 1.5$ in (10). This is due to fact that parameter μ is statistically orthogonal to both ϕ and ρ in the Tweedie likelihood (see details in Appendix D). Thus the estimation

of μ is almost independent to ϕ and ρ in large sample sense. We also observe such phenomenon in the simulation discussed in Section 4.2.

Warm start and active set. For the SKtweedie model in Section 3.2, a warm start option is implemented to improve computational efficiency. Specifically, the inner $\boldsymbol{\alpha}$-loop within the outer loop iteration m can be set to initialize at $\boldsymbol{\alpha}^{(m-1)}$, which may be closer to the final solution than the otherwise default initial value $\mathbf{0}$. On the other hand, the inner $\mathbf{w}$-loop within the iteration m always initializes at $\mathbf{w}^{(m-1)}$. This is related to the active set of $\mathbf{w}$, which is another feature that is implemented to improve efficiency.

The active set includes the indices of all the elements in $\mathbf{w}$ that are not equal to 0 in the previous inner loop iteration k: $\mathcal{A}^{(m,k)} = \{j : w_j^{(m,k)} \in (0, 1],\ j = 1, \ldots, p\}$. The weights not in the active set, i.e. $w_j^{(m,k)} = 0$ are not updated anymore and the corresponding variables are not involved in the subsequent calculation of the weighted kernel matrix $\mathbf{K}(\mathbf{w})$. The rationale is that, for any $w_j^{(m,k)} = 0$ within the m-th outer loop iteration, its partial derivative $\nabla_{w_j} g(\boldsymbol{\alpha}^{(m)}, \mathbf{w}^{(m,k)})$ is equal to λ_2 in many kernel functions such as the Gaussian radial basis function (RBF; Appendix C), and thus the gradient descent update in this direction will be $w_j^{(m,k+1)} = \text{proj}(0 - t^{(m,k)}\lambda_2) = 0, \forall \lambda_2 > 0$ according to (19). As a result, the weight w_j will remain 0 for the rest of the inner $\mathbf{w}$-loop as well as the outer loop if we keep updating it. By maintaining and updating the active set $\mathcal{A}$, we can avoid much unnecessary computation, especially when the number of noise variables is large.

Fitting the Ktweedie model with an intercept. This section discusses the implementation details when there is an intercept term in the model. Denote by $g(\alpha_0, \boldsymbol{\alpha})$ the objective function in (10). It is convex in $(\alpha_0, \boldsymbol{\alpha})$, which allows convenient alternating minimization. Based on Algorithm 1, after updating $\boldsymbol{\alpha}^{(k)}$ to $\boldsymbol{\alpha}^{(k+1)}$ with α_0 fixed at $\alpha_0^{(k)}$ in each iteration k (Line 6), we update $\alpha_0^{(k)}$ to $\alpha_0^{(k+1)}$. This can be done by solving the equation $\frac{\partial g(\alpha_0, \boldsymbol{\alpha}^{(k+1)})}{\partial \alpha_0} = 0$ analytically,

$$\alpha_0^{(k+1)} \leftarrow \log \frac{\sum_{i=1}^n y_i \exp[(1-\rho)\mathbf{K}_i^\top \boldsymbol{\alpha}^{(k+1)}]}{\sum_{i=1}^n \exp[(2-\rho)\mathbf{K}_i^\top \boldsymbol{\alpha}^{(k+1)}]}. \tag{21}$$

We also provide an `R` package called Ktweedie to implement the proposed method with all the aforementioned features. The core of the software is written in `Fortran` to maximize efficiency.

4 Simulation

We compare the Ktweedie and SKtweedie with the a number of existing models mentioned in Section 1 in terms of their prediction performance. These include the TGLM (Jørgensen and de Souza, 1994) implemented in the `R` package `statmod` (Giner and Smyth, 2016; Smyth et al., 2021; hereinafter TGLM), the TGAM model (Wood, 2011) in the `R` package `mgcv` (Wood, 2021; hereinafter MGCV), and the Gradient Tree-Boosted Tweedie model (Yang et al., 2018) in the `R` package `TDboost` (Yang et al., 2016b; hereinafter TDboost). The model tuning for the TDboost is replicated from Yang et al. (2018). We consider the RBF kernel and the Laplace kernel in the Ktweedie for demonstration purpose, whereas the users have the freedom to choose the kernel functions based on the characteristics of their own data. Throughout this section, we denote the true $f(\cdot)$ used in the simulations by $F(\cdot)$.

4.1 Case I

We compare the prediction accuracies of different models under the following two scenarios where the true target functions $F(\mathbf{x}) = \log(\mu)$ are nonlinear functions of the predictors.

Model 1: $\log(\mu) = F(\mathbf{x}) = 0.5\mathrm{I}(x > 0.5)$

Here the true $\log(\mu)$ is a non-smooth function of the one-dimensional predictor x. We assume that $x \sim \mathrm{Unif}(0, 1)$, and $y \sim \mathrm{Tw}(\mu, \phi, \rho)$ with $\rho = 1.5$ and $\phi = (0.1, 0.5, 1.0, 2.0)$.

Model 2: $\log(\mu) = F(\mathbf{x}) = \exp[-5(1 - x_1)^2 + x_2^2] + \exp[-5x_1^2 + (1 - x_2)^2]$

Here the true $\log(\mu)$ is a smooth nonlinear function of the predictors (x_1, x_2) with complex interactions. We assume that $x_1, x_2 \sim \mathrm{Unif}(0, 1)$, and $y \sim \mathrm{Tw}(\mu, \phi, \rho)$ with $\rho = 1.5$ and $\phi = (0.1, 0.5, 1.0, 2.0)$.

We generate training datasets with $n = 400$ observations and test datasets with $n' = 400$. The training dataset is fitted with Ktweedie. The inverse kernel width σ of the RBF and Laplace kernel functions and the regularization coefficient λ are determined using five-fold cross-validation based on the likelihood of the validation set. The criterion for the performance is the mean absolute deviation (MAD) of the predicted $\widehat{F}(\mathbf{x}) = \log(\widehat{\mu})$ from the true $F(\mathbf{x}) = \log(\mu)$ as follows

$$\text{MAD} = \frac{1}{n'} \sum_{i=1}^{n'} \left| F(x_i) - \widehat{F}(x_i) \right|.$$

The resulting MADs based on 100 replications are reported in Tables 1 and 2, and some sample predictions are plotted in Figures 2 and 3, for Model 1 and Model 2 respectively. In Model 1, the Ktweedie is not as good as the TDboost but is on par with the MGCV and better than the TGLM. This is as expected under a non-smooth setting due to the tree-based nature of the TDboost. In Model 2, the Ktweedie with the RBF kernel and the Laplace kernel outperforms MGCV, TDboost and TGLM in the smooth function case.

Table 1: The mean and standard errors of the mean absolute deviations of the predicted $\widehat{F}(\mathbf{x}) = \log(\widehat{\mu})$ from the true $F(\mathbf{x}) = \log(\mu)$ in Model 1 based on 100 replications for different values of ϕ.

ϕ	MGCV	TDboost	TGLM	RBF	Laplace
0.1	0.049 (0.0008)	0.030 (0.0012)	0.106 (0.0004)	0.059 (0.0007)	0.058 (0.0015)
0.5	0.086 (0.0019)	0.071 (0.0021)	0.112 (0.0007)	0.090 (0.0018)	0.085 (0.0024)
1.0	0.101 (0.0023)	0.091 (0.0031)	0.115 (0.0012)	0.106 (0.0021)	0.094 (0.0025)
2.0	0.131 (0.0031)	0.131 (0.0048)	0.134 (0.0027)	0.135 (0.0034)	0.120 (0.0036)

In addition, we study the estimation of ϕ and ρ using the profile likelihood approach proposed in Section 3.3. The setups of Model 1 and Model 2 are adopted with true $\rho = 1.5$ and true $\phi = 0.5$.

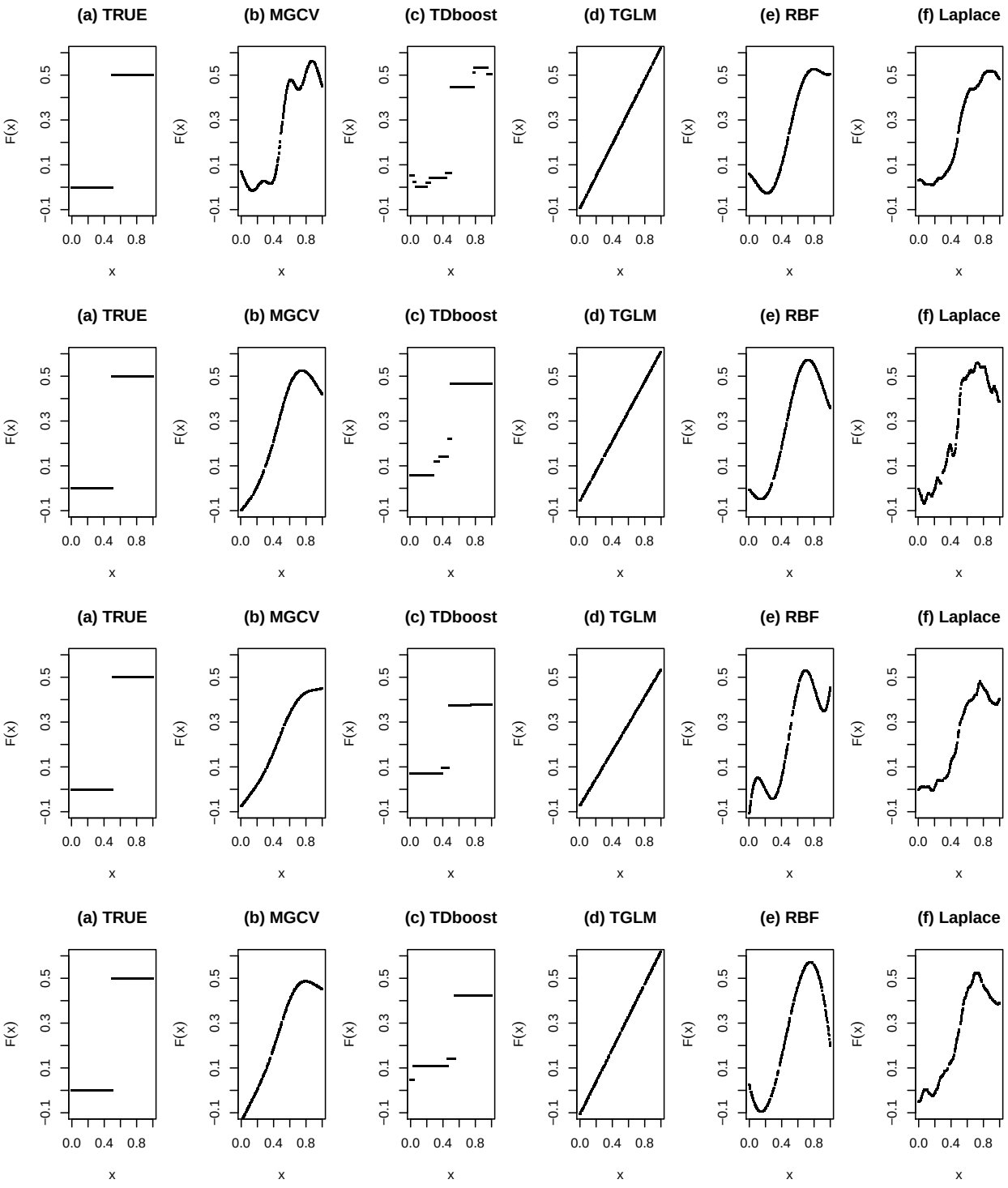

Figure 2: Fitted $\widehat{F}(\mathbf{x})$ vs. true $F(\mathbf{x})$ in Model 1 from a sample run (top to bottom $\phi = 0.1, 0.5, 1.0, 2.0$).

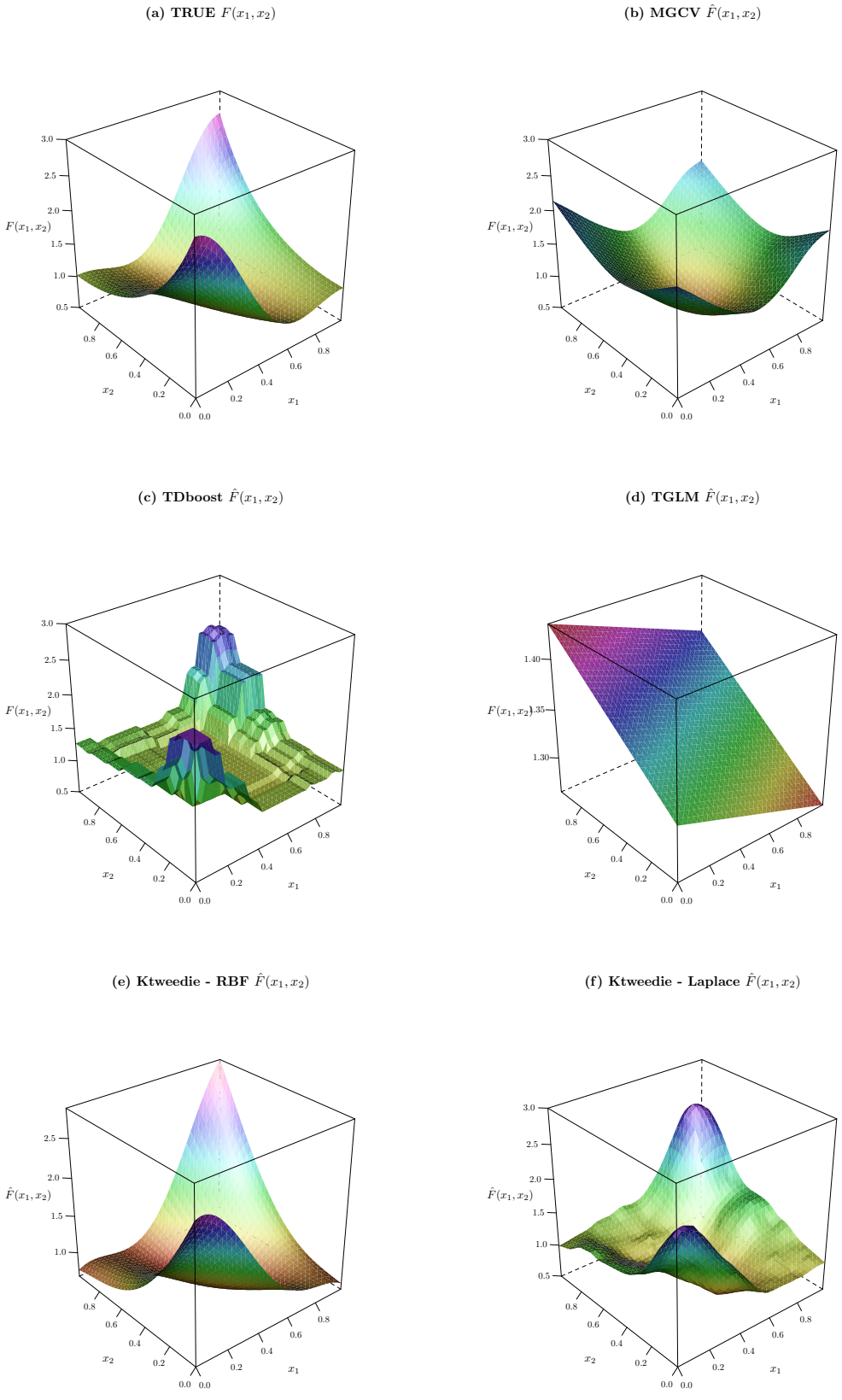

Figure 3: Fitted $\widehat{F}(\mathbf{x})$ vs. true $F(\mathbf{x})$ in Model 2 from a sample run ($\phi = 0.5$).

Table 2: The mean and standard errors of the mean absolute deviations of the predicted $\widehat{F}(\mathbf{x}) = \log(\widehat{\mu})$ from the true $F(\mathbf{x}) = \log(\mu)$ in Model 2 based on 100 replications for different values of ϕ.

ϕ	MGCV	TDboost	TGLM	RBF	Laplace
0.1	0.241 (0.0012)	0.084 (0.0007)	0.348 (0.0016)	0.065 (0.0009)	0.073 (0.0021)
0.5	0.248 (0.0012)	0.129 (0.0014)	0.345 (0.0020)	0.085 (0.0014)	0.095 (0.0017)
1.0	0.251 (0.0016)	0.156 (0.0023)	0.349 (0.0023)	0.102 (0.0021)	0.126 (0.0022)
2.0	0.264 (0.0023)	0.191 (0.0026)	0.354 (0.0029)	0.135 (0.0033)	0.170 (0.0037)

We only consider the RBF kernel in this analysis. In the procedure, a series of candidate ρ's are generated: $(1.02, 1.04, 1.06, \ldots, 1.96, 1.98)$, and the one that generates the greatest log likelihood as in (20) is used in the subsequent estimation of ϕ and μ. In Figure A1, the estimated likelihoods at different candidate ρ's from a sample run for each mode is plotted. On the left, $\widehat{\rho} = 1.52$ for Model 1, and on the right, $\widehat{\rho} = 1.58$ for Model 2. The estimations of ρ and ϕ and the MADs from 20 independent replications are summarized in Table A1. Note that, the MADs are slightly higher than the corresponding entries in Tables 1 & 2 due to a price of not knowing the true ϕ and ρ. Overall, the profile likelihood approach coupled with the Ktweedie (RBF kernel) is able to acquire good estimates of the true ρ and ϕ. Although this approach also provides good estimates of the true μ, we show numerically in Section 4.2 that estimating ϕ and ρ can be redundant when μ is the only parameter of interest.

4.2 Case II

We evaluate the performance of the Ktweedie in comparison with the MGCV and TDboost with complicated interactions among the predictors. The random function generator (RFG) model by Friedman (2001) is used in the simulation to generate the true target function $F(\cdot)$. Specifically, $F(\cdot)$ is randomly generated as a linear expansion of functions $\{g_k\}_{k=1}^{20}$:

$$F(\mathbf{x}) = \sum_{k=1}^{20} b_k g_k(\mathbf{z}_k), \tag{22}$$

where the b_k's are the coefficients generated from $\mathrm{Unif}[-1,1]$. In addition, $g_k(\mathbf{z}_k)$ is a function of $\mathbf{z}_k$, which is a subset of the p-dimensional variable $\mathbf{x}$ with size p_k,

$$\mathbf{z}_k = \{\mathbf{x}_{\psi_k(j)}\}_{j=1}^{p_k} \tag{23}$$

where each ψ_k is an independent permutation of the integers $\{1, 2, \ldots, p\}$. The size p_k is equal to $\min(\lfloor 2.5 + r_k \rfloor, p)$, where r_k is generated from an exponential distribution with mean 2. Thus $\mathbf{x}_{\psi_k(j)}$ in (23) indicates the j-th element of the permuted vector $\mathbf{x}_{\psi_k}$. By adopting this setup, we expect that each $g_k(\mathbf{z}_k)$ should have between 2 and p predictors ($\mathbf{x}$) and at least a few of the $g_k(\mathbf{z}_k)$'s should involve high-order interactions. For each k, the function $g_k(\mathbf{z}_k)$ is a p_k-dimensional Gaussian function:

$$g_k(\mathbf{z}_k) = \exp\left\{-\frac{1}{2}(\mathbf{z}_k - \mathbf{u}_k)^\top \mathbf{V}_k (\mathbf{z}_k - \mathbf{u}_k)\right\}, \tag{24}$$

where $\mathbf{u}_k$'s are mean vectors generated from $\mathrm{N}(\mathbf{0}, \mathbf{I}_{p_k})$ independently, and the covariance matrix $\mathbf{V_k}$ is defined by

$$\mathbf{V}_k = \mathbf{U}_k \mathbf{D}_k \mathbf{U}_k^\top,$$

where $\mathbf{U}_k$ was a random orthonormal matrix, $\mathbf{D}_k = \mathrm{diag}(d_k[1], d_k[2], \ldots, d_k[p_k])$ with $\sqrt{d_k[j]} \overset{i.i.d}{\sim} \mathrm{Unif}(0.1, 2.0)$. We generate the data $\{y_i, \mathbf{x}_i\}_{i=1}^n$ according to the Tweedie distribution,

$$y_i \sim \mathrm{Tw}(\mu_i, \phi, \rho), \quad \mathbf{x}_i \sim \mathrm{N}(\mathbf{0}, \mathbf{I}_p), \quad i = 1, 2, \ldots, n$$

where $\mu_i = \exp[F(\mathbf{x}_i)]$.

For the simulation, we set the sample size $n = 100$, the dimension $p = 10$ for both training data and test data, the Tweedie parameters $\rho = 1.5$ and $\phi = (0.1, 0.5, 1.0, 2.0)$. The models are fitted with known ρ. The simulation is replicated for 100 times. The MADs for MGCV, TDboost and Ktweedie are summarized in Table 3 and plotted in Figure 4. The results suggest that in the presence of complicated interactions among the predictors, the Ktweedie outperforms both the MGCV and TDboost.

Table 3: The mean and standard errors of the mean absolute deviations in Case II for different values of ϕ based on 100 replications.

ϕ	MGCV	TDboost	RBF	Laplace
0.1	0.764 (0.034)	0.680 (0.039)	0.615 (0.038)	0.650 (0.036)
0.5	0.983 (0.047)	0.894 (0.054)	0.832 (0.027)	0.767 (0.031)
1.0	1.328 (0.059)	1.136 (0.052)	1.073 (0.051)	1.044 (0.052)
2.0	0.985 (0.040)	0.572 (0.030)	0.562 (0.031)	0.569 (0.030)

We further examine how the index parameter ρ used in the model fitting affects the estimation accuracy of μ. Recall that the data is generated using $\rho = 1.5$ and $\phi = 0.5$. We consider the Ktweedie with the RBF kernel and use a series of different ρ's in the model fitting. As shown in Figure 5, the estimation accuracy of μ is almost unaffected by ρ.

4.3 Case III

We study the performance of SKtweedie under a simulation setting with a large number of predictors ($n = 100$, $p = 50$). We consider a simplified version of Case II described in Section 4.2. Specifically,

1. Replace (22) with $F(\mathbf{x}) = \sum_{k=1}^{5} b_k g_k(\mathbf{z}_k)$
2. The size p_k in $\mathbf{z}_k = \{x_{\psi_k(j)}\}_{j=1}^{p_k}$ is equal to $\min(\lfloor 1.5 + r_k \rfloor, 2)$, where r_k is generated from an exponential distribution with mean 1. As a result, each $\mathbf{z}_k$ contains 1 to 2 variables. In addition, only 5 of the total 50 variables can be used in the generation of $F(\mathbf{x})$ - the other 45 are noise variables.
3. Replace (24) with $g_k(\mathbf{z}_k) = \mathbf{z}_k^\top \mathbf{z}_k$ to reduce the variation in the data generation.

All other simulation settings remain the same as in Case II. Due to the relatively large dimension of $\mathbf{x}$, the MGCV becomes computationally infeasible thus is not included in this simulation. It is

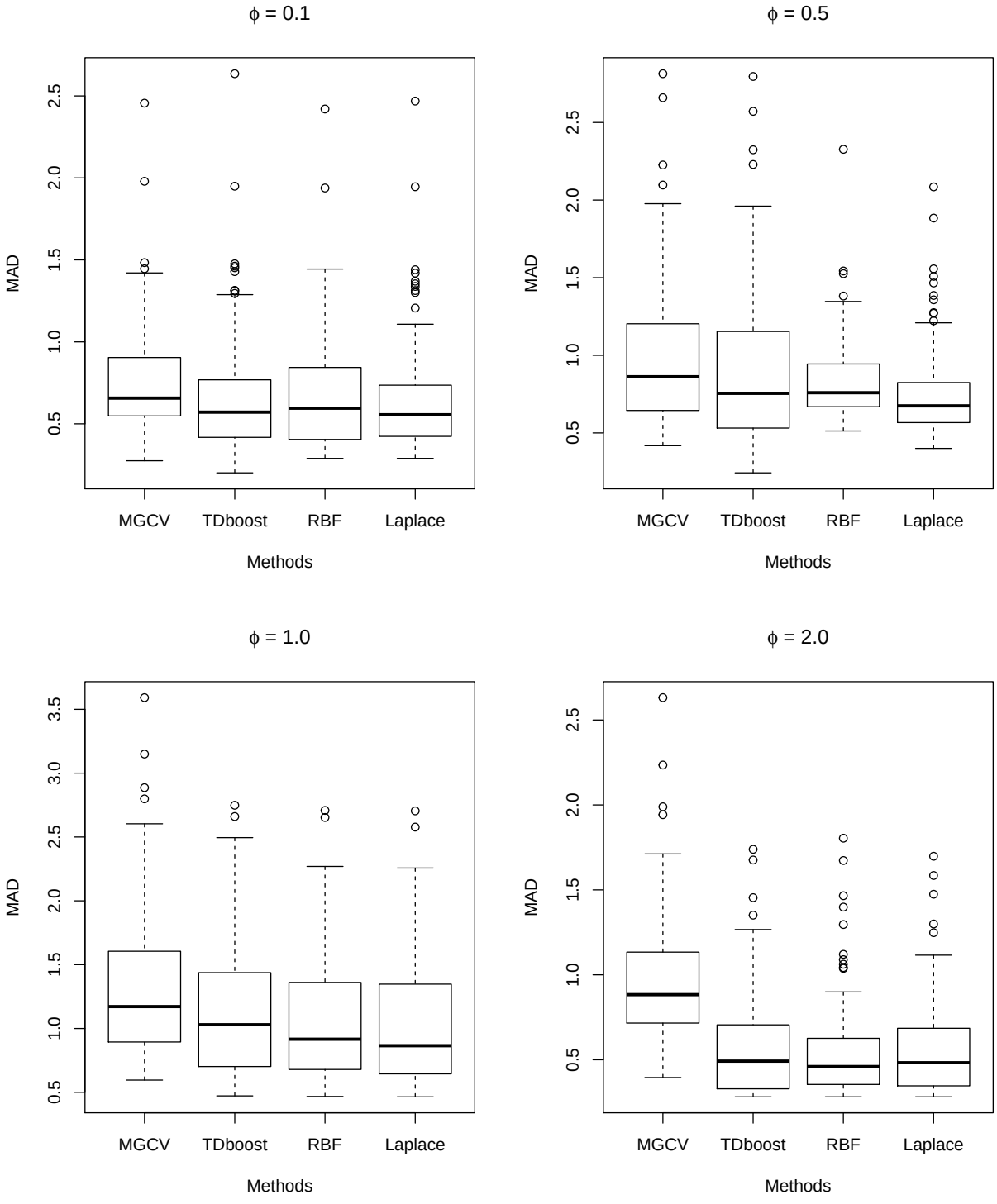

Figure 4: Distribution of the mean absolute deviations from the MGCV, TDboost, and Ktweedie (RBF and Laplace kernel) in Case II based on 100 independent replications.

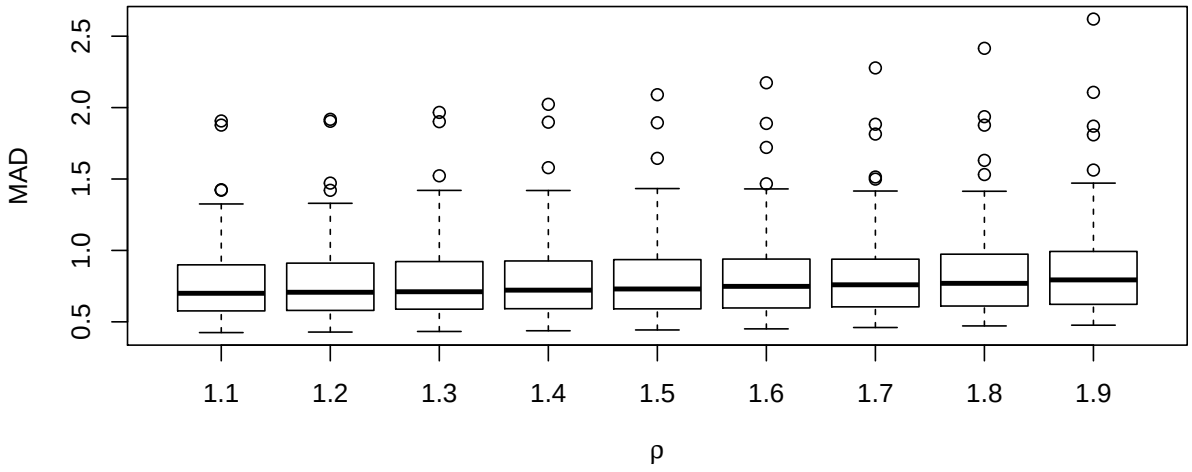

Figure 5: Boxplot of the mean absolute deviations for different values of the index parameter $\rho \in \{1.1, 1.2, \ldots, 1.9\}$ used during model fitting when the true value ($\rho = 1.5$) is unknown. The estimation accuracy is almost unaffected by ρ.

worth explaining the hyperparameter tuning strategy of the SKtweedie. As mentioned in Section 2.2, the weight regularization coefficient λ_2 controls the sparsity in the variables. Similar to the LASSO, there exists a $\lambda_2^{\max}(\sigma, \lambda_1)$ such that for all $\lambda_2 \geq \lambda_2^{\max}(\sigma, \lambda_1)$, all weights are zero for the given σ and λ_1. In our experiments, grid search or random search for the three hyperparameters regularly encounters combinations that lead to such a mean model. This is because $\lambda_2^{\max}(\sigma, \lambda_1)$ can be very sensitive to σ, λ_1 and the data and it is difficult to set proper search ranges that can control the relative magnitude of the three hyperparameters. We therefore use a tuning strategy that combines the random search and a solution path. Specifically, the tuning of σ and λ_1 is performed with a random search with cross-validation (which is the tuning strategy for the Ktweedie), then a solution path for λ_2 is constructed with the chosen σ and λ_1. The MADs for the TDboost, Ktweedie and SKtweedie with the RBF kernel are compared and summarized in Table 4 Figure 6. Under high dimensional setting, our kernel method performs better than the TDboost. In addition, SKtweedie is able to achieve LASSO-type variable selection, which is tested more purposefully in Section 4.4.

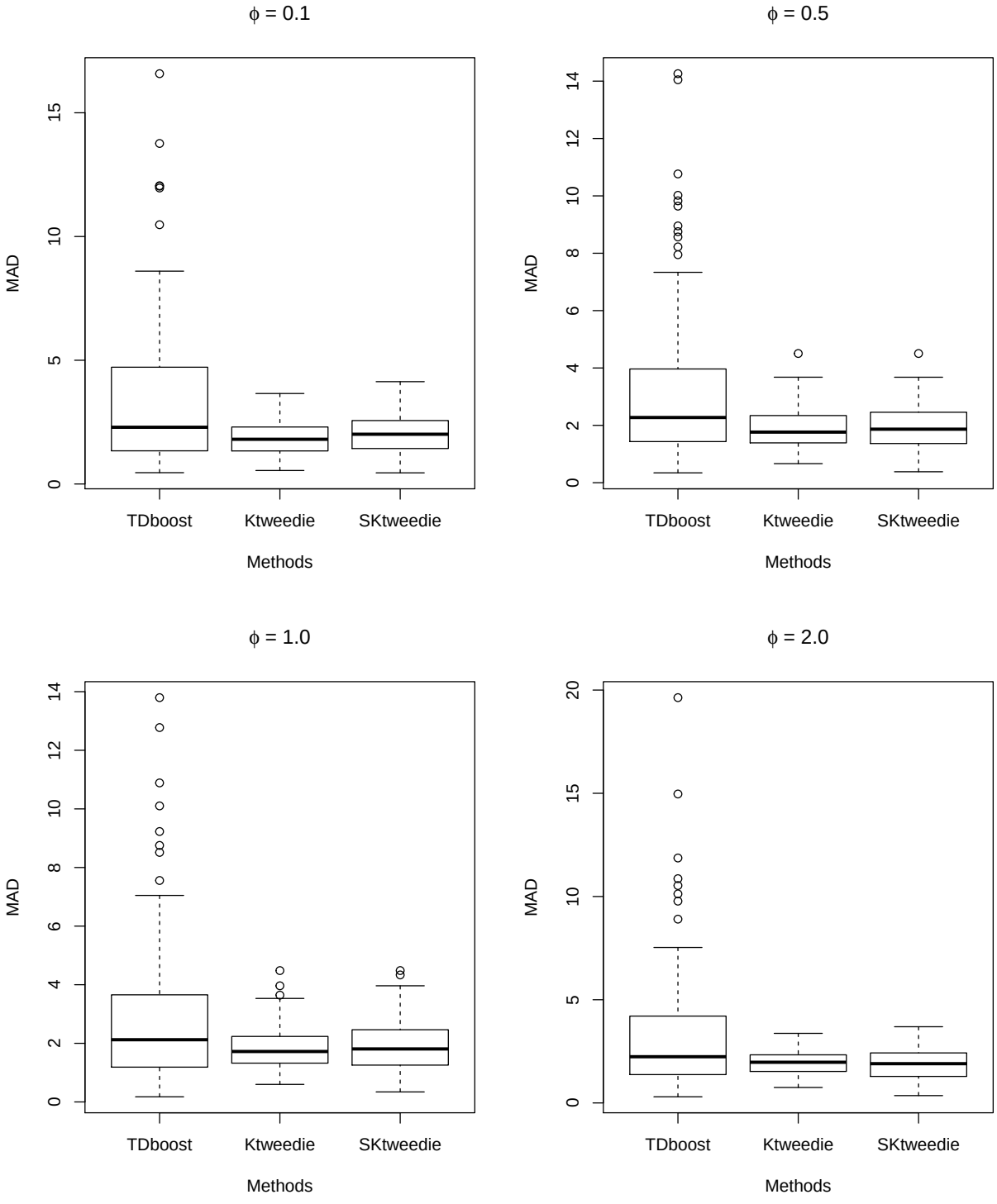

Figure 6: Distribution of the mean absolute deviations from the TDboost, Ktweedie, and SKtweedie in Case III based on 100 independent replications.

Table 4: The mean and standard errors of the mean absolute deviations in Case III for different values of ϕ based on 100 replications.

ϕ	TDboost	Ktweedie	SKtweedie
0.1	3.496 (0.3205)	1.844 (0.0731)	1.997 (0.0863)
0.5	3.341(0.2986)	1.932 (0.0785)	1.975 (0.0856)
1.0	3.006 (0.2706)	1.846 (0.0706)	1.914 (0.0839)
2.0	3.346 (0.3262)	1.933 (0.0608)	1.859 (0.0790)

4.4 Case IV

The purpose of this simulation is to test the variable selection performance of the SKtweedie. Consider the model:

$$\log(\mu) = F(\mathbf{x}) = \sin(\mathbf{x})^\top \boldsymbol{\beta}^{\text{true}}$$

where $\mathbf{x} \in \mathbb{R}^p$ is randomly generated from a standard normal distribution and $\boldsymbol{\beta}^{\text{true}} \in \mathbb{R}^p$ are the true coefficients.

We generate training datasets $\{y_i, \mathbf{x}_i\}_{i=1}^n$ with $n = 100, 200, 500$ and three different dimensions $p = 10, 50, 200$. The true coefficients are $\boldsymbol{\beta}^{\text{true}} = [6, -4, 3, 2, -2, 0, \ldots, 0]^\top$, whose first five non-zero entries are the coefficients for the signal variables and the remaining zero entries corresponds to the noise variables. The same strategy used in Case III is used to tune the hyperparameters. For illustration, we show in Figure 7 a visual demonstration of a sample solution path for λ_2 at an arbitrary combination of σ and λ_1. To formally test the variable selection performance, we fit SKtweedie with the tuning strategy mentioned in Section 4.3. Figure 8 shows the estimated weights in the 20 replications for $p = 10$ and 50 ($p = 200$ in Figure A2, Appendix E) with $n = 500$. Each column corresponds to a replication and each row corresponds to a variable. The red rectangle indicates the true signal variables, and the grayscale represents the magnitude of the estimated weights with a value between 0 and 1. The average precision and recall of the twenty replications are summarized in Table 5. Overall SKtweedie is able to achieve good variable selection accuracy.

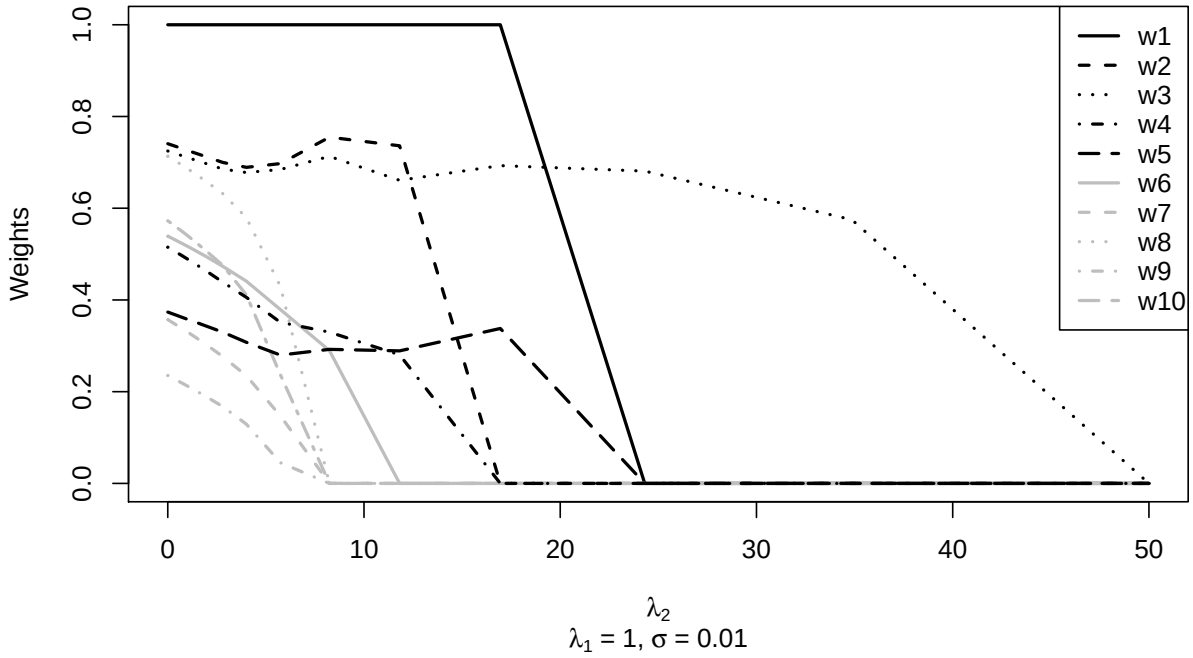

Figure 7: A sample solution path for λ_2 in SKtweedie with the Gaussian RBF kernel in case IV with arbitrary λ_1 and σ.

	$p = 10$		$p = 50$	
n	Precision	Recall	Precision	Recall
100	87.4% (3.3%)	84.0% (4.3%)	73.3% (6.1%)	60.0% (6.5%)
200	91.5% (2.9%)	85.0% (4.1%)	81.9% (4.7%)	73.0% (5.1%)
500	94.4% (2.9%)	94.0% (2.6%)	86.3% (6.5%)	94.7% (2.1%)

Table 5: Mean precision and recall of the variable selection accuracy with standard errors under different sample sizes and dimensions in Case IV based on 20 replications.

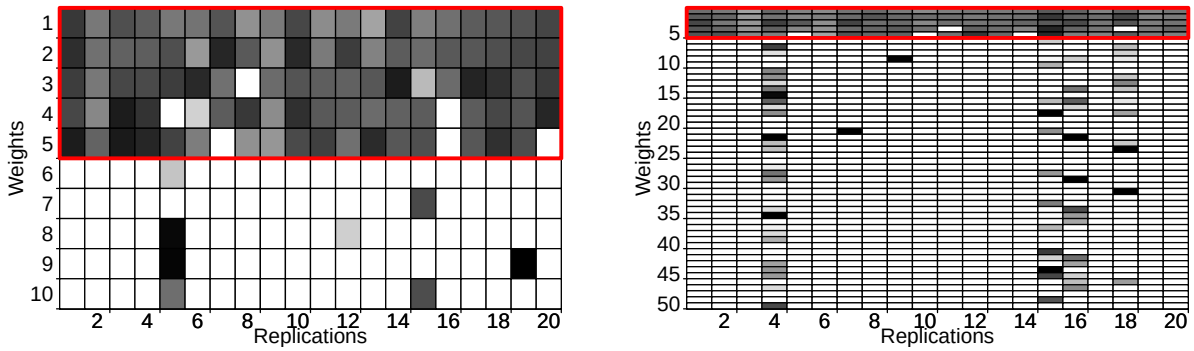

Figure 8: Variable selection results using SKtweedie with the Gaussian RBF kernel (left: $p = 10$, right: $p = 50$). Each column corresponds to a replication and each row corresponds to a variable, thus within the red rectangles are the true signal variables. The grayscale represents the magnitude of the estimated weights with a value between 0 and 1.

5 Real data analysis

We demonstrate the application of the proposed Tweedie models in two insurance operations.

5.1 Case study I: Claims reserving

Dataset. In claims reserving, data are often organized in a triangular format, known as the "run-off triangle". We consider the triangle of paid losses for workers compensation insurance of a large

			Development Year				
	Accident Year		1	2	$\cdots$	9	10
Fully Developed	1989	1	$Y_{1,1}$	$Y_{1,2}$	$\cdots$	$Y_{1,9}$	$Y_{1,10}$
	1990	2	$Y_{2,1}$	$Y_{2,2}$	$\cdots$	$Y_{2,9}$	$Y_{2,10}$
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
	1996	8	$Y_{8,1}$	$Y_{8,2}$	$\cdots$	$Y_{8,9}$	$Y_{8,10}$
Run-off Triangle	1997	9	$Y_{9,1}$	$Y_{9,2}$	$\cdots$	$Y_{9,9}$	$Y_{9,10}$
	1998	10	$Y_{10,1}$	$Y_{10,2}$	$\cdots$	$Y_{10,9}$	
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\iddots$		
	2005	17	$Y_{17,1}$	$Y_{17,2}$			
	2006	18	$Y_{18,1}$				

Table 6: A typical run-off triangle of incremental paid losses (shaded) and the additional accident years with fully developed claims.

insurer in US. The data is obtained from the Schedule P of the National Association of Insurance Commissioners (NAIC) database (Meyers and Shi, 2011; NAIC, 2021). Let i and j be the *accident year* (AY) and the *development year* (DY) respectively, and $t = i + j$ be the *calendar year*. Assume $i = 1, \ldots, I$, $j = 1, \ldots, J$, $J \leq I$. Let $Y_{i,j}$ denote the amount losses paid by the insurer for claims occurred in the i-th AY during the j-th DY. We collect the incremental losses $Y_{i,j}$ for $I = 18$ AYs (from 1989 to 2006) where each accident year has a development period of $J = 10$ years, following Sriram and Shi (2020). Table 6 exhibits the data that is available to an analyst by the end of 2006. The shaded portion of the dataset with AY $\in \{10, \ldots, 18\}$ represents the standard run-off triangle which is the most widely used data format for loss reserving. Our dataset contains 8 additional AYs that are fully developed. The goal of loss reserving is to predict the future payments in the lower triangle $\{Y_{i,j} : 10 \leq i \leq 18, 2 \leq j \leq 10, 20 \leq i + j \leq 28\}$ based on the observed payments in the upper trapezoid $\{Y_{i,j} : i + j \leq 19\}$.

Models. Define $Y_t = \{Y_{i,j} : i + j = t\}$ to be the vector of paid losses in calendar year t, which is located on the anti-diagonal (lower left to upper right) in Table 6. Assume Y_t satisfies the Markov

property such that the joint distribution of $Y_2, \ldots, Y_T$ can be factorized as

$$g(Y_2, ..., Y_T) = g(Y_2|H_2) \prod_{t=3}^{T} g(Y_t|Y_{t-1}, H_t), \tag{25}$$

where $T = 20$ is the latest calendar year in the training data,, and H_t is a set of additional predictors. In this analysis, we set $H_t = \{(i, j) : i+j = t\}$ where i represents the AY effect and j represents the DY effect. Note that $g(Y_2|H_2)$ is the initial condition that can be omitted under some assumptions. Then we have the conditional probability in the following form,

$$\begin{aligned} g(Y_t|Y_{t-1}, H_t) &= \prod_{i+j=t} g(Y_{i,j}|Y_{t-1}, H_t) \\ &= \prod_{i+j=t} g(Y_{i,j}|Y_{i-1,j}, Y_{i,j-1}, i, j). \end{aligned} \tag{26}$$

We further assume that for each $i = 1, \ldots, I$, $j = 1, \ldots, J$, the incremental loss $Y_{i,j}$ follows a Tweedie distribution

$$Y_{i,j}|(Y_{i-1,j}, Y_{i,j-1}, i, j) \sim \text{Tw}(\mu_{i,j}, \phi, \rho). \tag{27}$$

The fact that each $Y_{i,j}$ represents the aggregate claims from a large number of policyholders in the portfolio makes the Tweedie model a natural choice.

We consider several versions of Tweedie model assumptions in (27) – each of them corresponds to a specific model under comparison: Model 1 – the Tweedie linear model (TGLM) with $\log \mu_{i,j} = \beta_1 Y_{i-1,j} + \beta_2 Y_{i,j-1} + \boldsymbol{\beta}_3^\top \mathbf{d}(i) + \boldsymbol{\beta}_4^\top \mathbf{d}(j)$ where $\mathbf{d}(i)$ and $\mathbf{d}(j)$ are dummy variables for the factor predictors i and j, respectively; Model 2 – the Tweedie additive model (MGCV) with $\log \mu_{i,j} = h_1(Y_{i-1,j}) + h_2(Y_{i,j-1}) + h_3(i) + h_4(j)$; Model 3 – the tree-based gradient boosting model (TDboost) with $\log \mu_{i,j} = \sum_{m=1}^{M} T_m(Y_{i-1,j}, Y_{i,j-1}, i, j)$, and Model 4 – the kernel Tweedie model (Ktweedie) with $\log \mu_{i,j} = f(Y_{i-1,j}, Y_{i,j-1}, i, j)$ where $f \in \mathcal{H}_K$. We refer to the four predictors $\{Y_{i-1,j}, Y_{i,j-1}, i, j\}$ in the above Tweedie models as *TOP*, *LEFT*, *AY,* and *DY*, respectively. In addition, we consider Model 0 – the classic Mack Chain-Ladder (MCL) algorithm (Mack, 1993) as a baseline. The MCL is an industry benchmark and is mathematically equivalent to modeling $Y_{i,j}$

No.	Model	Triangle	Diagonal	IBNR
	True	0	0	245768.0
0	MCL	1919.7	1297.2	197416.0
1	TGLM	2036.0	1456.2	192938.4
2	MGCV	2139.8	1763.5	194288.2
3	TDboost	4460.2	7832.4	318237.9
4	Ktweedie	**1679.1**	**922.1**	215499.1
5	TGLM-2	1902.9	1219.7	197596.9
6	TGLM-x	2053.7	1547.1	192182.5
7	MGCV-x	2114.4	1107.5	194257.5

Table 7: The root mean square errors (on the entire run-off triangle and on the diagonal immediately next to the observed data) and the incurred but not reported (IBNR) amounts of different loss reserving models.

using the Poisson GLM with the factor predictors i and j.

Performance Comparison. We train the models using the trapezoid-shaped training data $\{Y_{i,j} : i \geq 2, j \geq 2, i + j \leq 20\}$, and compare their prediction performance on two test data: **(a)** the anti-diagonal immediately next to the observed data $\{Y_{i,j} : i + j = 21\}$; **(b)** the entire lower triangle $\{Y_{i,j} : i \geq 11, j \geq 3, i + j \geq 21\}$. The first test corresponds to one-year prediction and the second one predicts the ultimate losses at the valuation date. For the lower triangle case, predictions are made sequentially, i.e. predicted $\widehat{Y}_{i,j}$ is plugged into the subsequent predictions for $\widehat{Y}_{i+1,j}$ and $\widehat{Y}_{i,j+1}$. Note that extra data $\{Y_{ij} : i + j = 20\}$ is added to the training set to avoid extrapolation in prediction due to our model specification. For fair comparison, all candidate models are trained using the same data.

Table 7 reports the RMSEs for test data. The Ktweedie model with the RBF kernel outperforms all the other methods and is the only Tweedie model that beats the MCL. An in-depth analysis reveals that the advantage of the Ktweedie mainly comes from: **(1)** its appropriate use of the Tweedie distribution; **(2)** its flexible functional structure that can capture the complex interactions in the data.

To demonstrate **(1)**, we fit Model 5 – a Tweedie GLM with only two factor predictors AY and

DY (TGLM-2) and observe that the resulting RMSEs are lower than those of the MCL (Table 7). Given the connection between the MCL and the Poisson GLM, we conclude that the Tweedie-based models offers superior prediction to the Poisson-based models for claim reserving.

To demonstrate **(2)**, we first show the interaction effects in the data. As an illustration, Figure 9 visualizes the relationship between $Y_{i,j}$ and the pair (AY, DY) using a heat map. Next we examine the two-way interaction effects implied by the fitted function of the Ktweedie. The log predicted losses are plotted against $\binom{4}{2} = 6$ pairs of the predictors among TOP, LEFT, AY and DY in Figure 10 (a)–(f), among which, panel (f) emphasizes the nonlinear interaction between TOP and LEFT. Finally, we stress that as a fully nonparametric method, the Ktweedie accommodates the complex interactions better than the more restricted models. To this end, we compare the Ktweedie with Model 6 – a TGLM model with a two-way interaction between TOP and LEFT (TGLM-x), and Model 7 – a modified MGCV model with a full tensor product smooth term for the TOP-LEFT interaction (MGCV-x). The interaction effects in the TGLM-x and the MGCV-x are plotted in Figure 11 (a) and (b), respectively. In particular, the interaction effect in the TGLM-x, which is represented by the product of TOP and LEFT, is much more restricted than in the Ktweedie (Figure 10.f). The results in Table 7 shows that the TGLM-x does not provide improvement over the original TGLM, while the MGCV-x significantly lowers the RMSE than the original additive model for one-year prediction. But neither models delivers a better performance than the Ktweedie. It is also worth noting that despite its potential in capturing the complex interactions in the data, the TDboost cannot be trained effectively on such fairly small dataset.

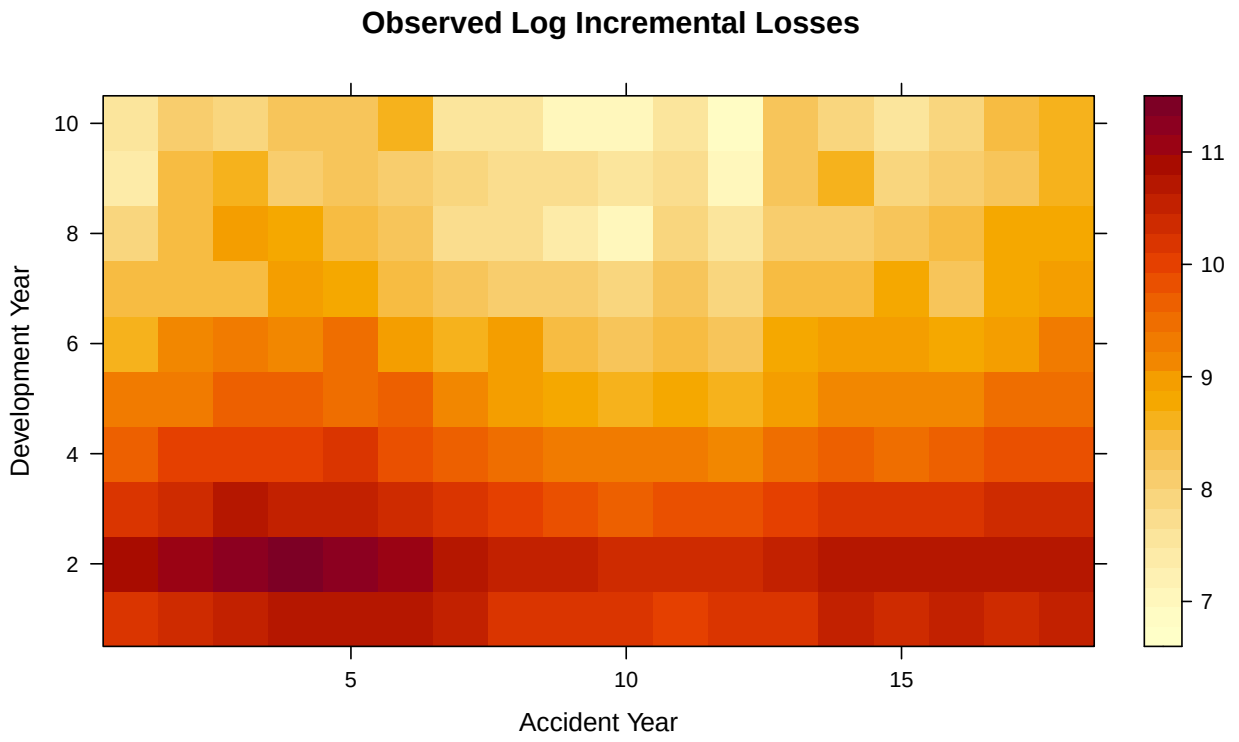

Figure 9: A heatmap of the log incremental losses by accident year and development year.

5.2 Case study II: Ratemaking

Dataset. For the purpose of pricing insurance contract, we are to predict the loss cost of individual policyholders. We analyze an automobile insurance claims dataset in Yip and Yau (2005). The dataset contains the total loss amount (y_i) for 10,296 observations over a five-year period ($v_i = 5$), among which 6,290 (61%) observations have zero losses, and 961 (9%) observations have losses over $10,000. In addition, the dataset contains basic rating variables that the insurer uses for risk classification. We summarize the set of rating variables in Table 8 and use them as predictors in the Tweedie models.

Models. Five different models are compared in terms of their prediction accuracy, including a mean model (MEAN), TGLM, MGCV, TDboost and Ktweedie (with the RBF kernel). The real data analysis is conducted in the following way: in each replication, the dataset is split into a training

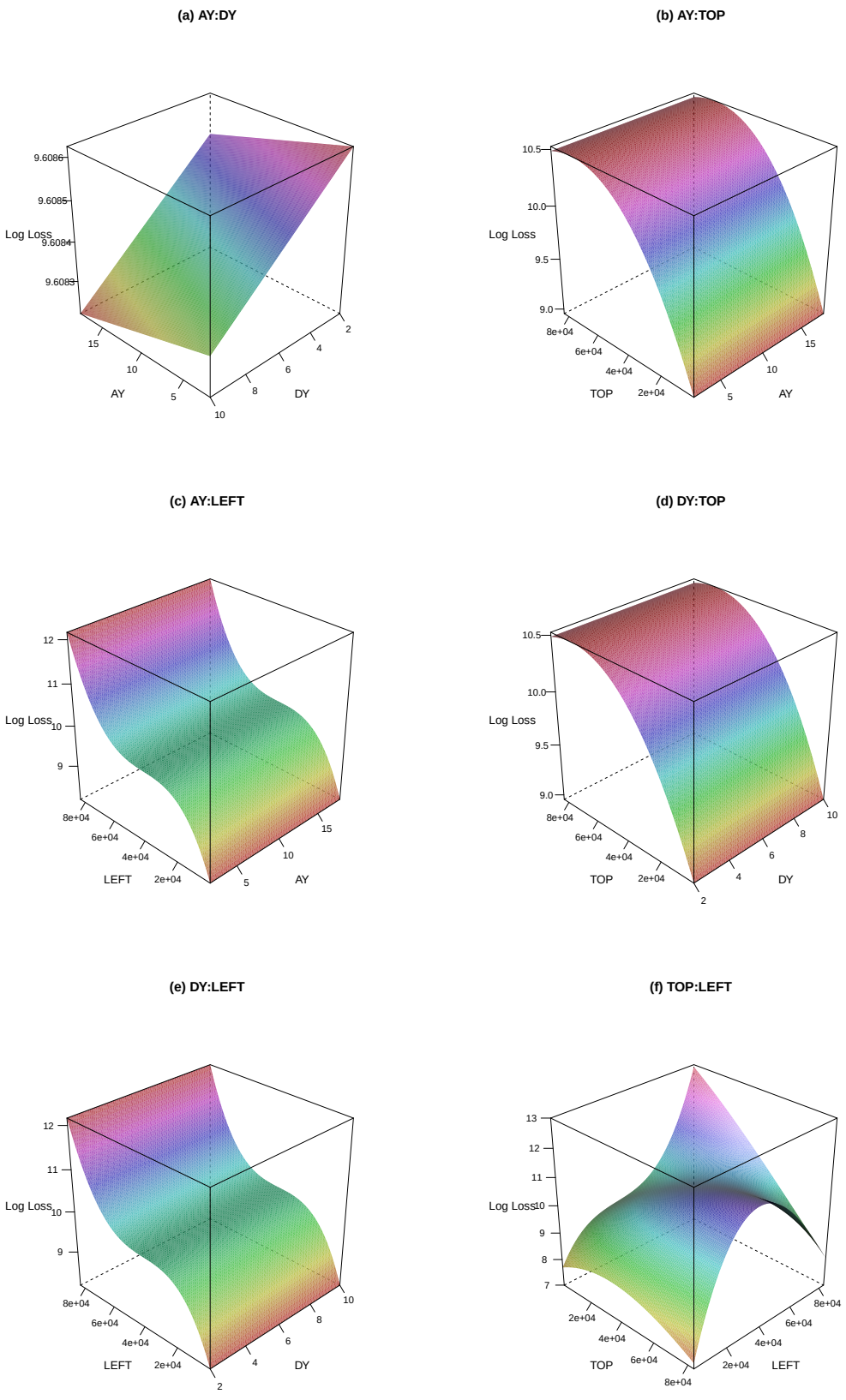

Figure 10: Visualization of pairwise interactions using Ktweedie. With the effects of accident year and development year fixed, the predictions made with the Ktweedie model reveals complex nonlinear interaction between the variables TOP and LEFT.

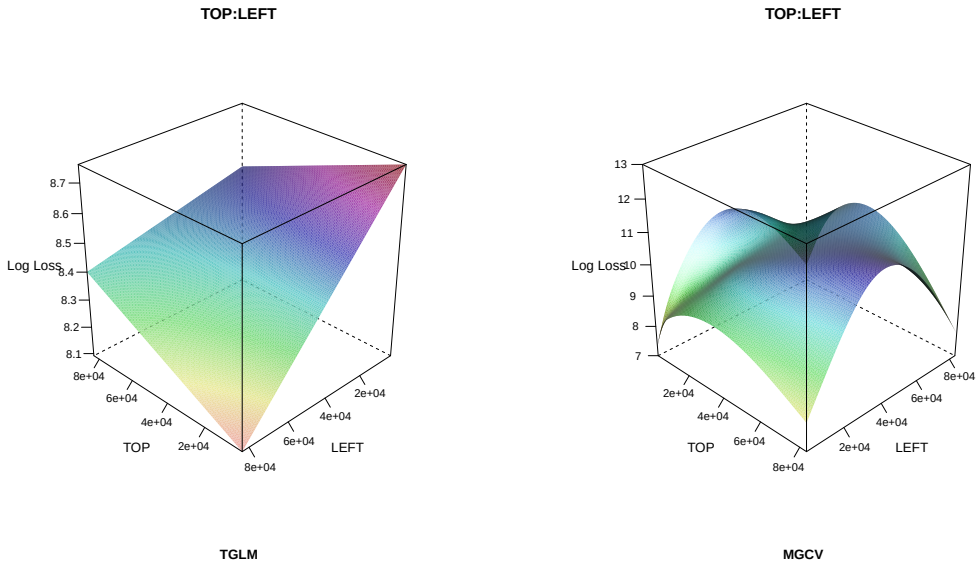

Figure 11: Visualization of the TOP-LEFT interaction using predictions made by the TGLM (left) and MGCV (right) models.

ID	Variable	Description
1	AGE	Driver's age
2	BLUEBOOK	Value of vehicle
3	HOMEKIDS	Number of children
4	KIDSDRIV	Number of driving children
5	MVR_PTS	Motor vehicle record points
6	NPOLICY	Number of policies
7	RETAINED	Number of years as a customer
8	TRAVTIME	Distance to work
9	AREA	Home/work area: Rural, Urban
10	CAR_USE	Vehicle use: Commercial, Private
11	GENDER	Driver's gender: F, M
12	MARRIED	Married or not: Yes, No
13	REVOKED	Whether license revoked in past 7 years: Yes, No

Table 8: Explanatory variables in the claim history data set.

	MAD	RMSE	nGini
MEAN	5.316 (0.121)	8.759 (0.186)	0.005 (0.017)
TGLM	4.420 (0.078)	8.167 (0.245)	0.594 (0.012)
MGCV	4.258 (0.069)	7.742 (0.179)	0.595 (0.012)
TDboost	4.119 (0.066)	7.571 (0.180)	0.600 (0.010)
Ktweedie	4.212 (0.069)	7.621 (0.186)	0.604 (0.011)

Table 9: The mean absolute deviations, root mean square errors and the normalized Gini indices with standard deviations in the parentheses of the prediction made with different models based on 20 replications.

set and a test set of equal size. All five models are trained using the training set, then the trained models are used to make predictions on the test set. We replicate the above procedure for 20 times and calculate the average prediction accuracy over the 20 replications.

The mean model predicts using an intercept-only linear Tweedie model, which serves as a non-informative baseline. In the MGCV model, the effect of the numerical variables (AGE, BLUEBOOK, HOMEKIDS, KIDSDRIV, MVR PTS, NPOLICY, RETAINED, and TRAVTIME) are modeled by splines. The TDboost model is tuned with the five-fold cross-validation as described in Yang et al. (2018). For the Ktweedie, the training involves the tuning of the kernel parameter σ and the regularization coefficient λ with a five-fold cross-validation.

Performance Comparison. To compare predictive performance, we first examine the MAD, the root mean squared error (RMSE), and the normalized Gini index (nGini) proposed by Ye et al. (2018). A model with a smaller value of MAD or RMSE, or a larger value of nGini is preferred. Table 9 reports the average accuracy measures with the associated standard deviations. The results suggest that the TDboost and Ktweedie outperform all the other models. The performance of TDboost and Ktweedie are comparable, with the nGini favoring the Ktweedie while the MAD and RMSE slightly supporting the TDboost.

Due to the large proportion of zero outcomes and the high skewness, we also use the ordered Lorenz curve and the associated Gini index as performance measures (Frees et al., 2011). We

Base premium	Competing premium				
	MEAN	TGLM	MGCV	TDboost	Ktweedie
MEAN	0	48.154 (0.926)	48.297 (0.976)	48.646 (0.819)	48.997 (0.883)
TGLM	8.140 (2.453)	0	8.115 (1.704)	13.527 (2.002)	12.634 (1.278)
MGCV	6.515 (2.303)	2.751 (3.573)	0	10.367 (2.250)	9.274 (2.079)
TDboost	-0.983 (1.773)	3.741 (2.250)	3.089 (1.907)	0	5.963 (1.880)
Ktweedie	2.511 (2.353)	2.106 (1.361)	3.354 (1.651)	8.952 (1.731)	0

Table 10: The averaged Gini indices (Frees et al., 2011) and standard deviations in the auto-insurance claim data example based on 20 replications.

consider a pairwise comparison among alternative models, and Table 10 summarizes the average Gini indices and standard deviations from 20 replications. Each row uses one candidate model as the base, and evaluates the "relative improvement" made by the competing models. A large and significant Gini index indicates by switching from the base to the alternative, the insurer could better separate low and high risks. First, the MEAN model is the least favorable one as it does take into account the rating variables. Second, both the TDboost and Ktweedie show superior performance over other Tweedie models. Third, the selection between the TDboost and Ktweedie is not obvious, neither showing substantial advantage over the other. Figure A3 exhibits the ordered Lorenz curves from one replication, which shows the superior performance of the Ktweedie to other Tweedie models. The Gini indices in Table 10 are calculated as twice the area between the curve and the equity line.

Overall, we conclude that the Ktweedie clearly outperforms the TGLM and MGCV and its prediction accuracy is on par with the TDboost. It is within our expectation that the TDboost also has a good performance on this dataset since the sample size is sufficiently large while the dimension is relatively low – a setting that generally favors the tree-based gradient boosting methods. In addition, a number of binary predictors in the dataset puts the tree-based methods in advantage due to its natural handling of the partition of the input space.

Last, we show that the SKtweedie model can be used to identify important predictors in the model. We first select parameters λ_1 and σ using cross validation, and then construct a solution

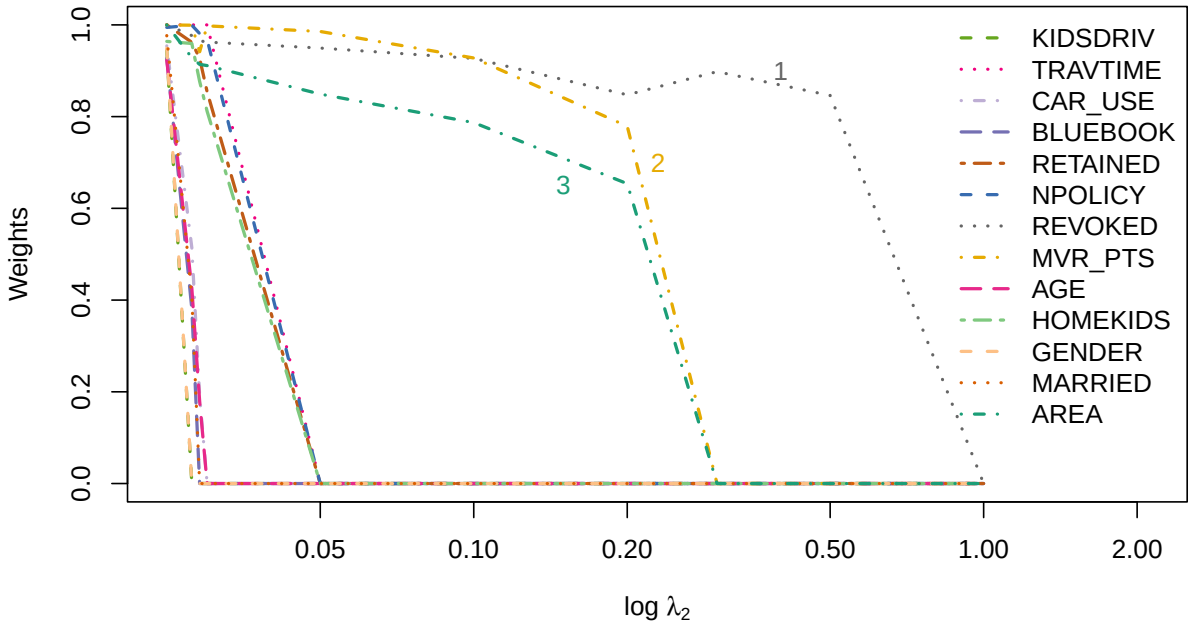

Figure 12: A sample solution path obtained by the SKtweedie model showing the change in variable weights with increasing sparsity-inducing regularization coefficient. The most important predictors of the claim loss are (1) the history of license revocation, (2) the motor vehicle record points, and (3) the area of the policyholder's home/work.

path for the SKtweedie with respect to λ_2. As shown in Figure 12, the weights for most variables shrink to zero quickly, except for REVOKED, AREA and MVR_PTS. The results indicate that the history of license revocation, the diver's motor vehicle record points, and whether the policyholder lives/works in rural or urban areas are the most important predictors for the insurance losses. The findings are highly consistent with those by the TDboost, where the most significant predictors are REVOKED, AREA, BLUEBOOK and MVR_PTS as shown in Figure 9 and Section 6.4 in Yang et al. (2018).

6 Conclusion

In this paper we have derived a kernel Tweedie model in RKHS and also proposed a sparse variant which integrated variable selection via sparsity-inducing regularization. We have demonstrated the favorable prediction performance of the proposed methods through comprehensive simulation and two case studies using real data. The proposed Ktweedie and SKtweedie are implemented in `Fortran` with an `R` interface for improved speed. We apply several computational tricks including the warm start and the active set.

One major issue with kernel learning is the computational limitation. e.g. the computation and storage of the Gram matrix of a kernel problem can be very expensive when the sample size is large. To avoid the problem of calculating the whole Gram matrix (it costs $O(n^2p)$), it remains interesting to develop low-cost approximations of the kernel matrix through subsampling methods (Rudi et al., 2015) or random features (Rahimi et al., 2007). These approximations can also improve prediction performances as they induce implicit regularization.

References

Allen, G. I. (2013) Automatic feature selection via weighted kernels and regularization. *Journal of Computational and Graphical Statistics*, **22**, 284–299. 1, 2.2, 2.2

Argyriou, A., Hauser, R., Micchelli, C. A. and Pontil, M. (2006) A DC-programming algorithm for kernel selection. In *Proceedings of the 23rd international conference on Machine learning*, 41–48. 2.2

Armijo, L. (1966) Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific J. Math.*, **16**, 1–3. 3.1

Blier-Wong, C., Cossette, H., Lamontagne, L. and Marceau, E. (2021) Machine learning in P&C insurance: A review for pricing and reserving. *Risks*, **9**, 4. 1

Brent, R. P. (2013) *Algorithms for minimization without derivatives*. Courier Corporation. 3.3

Broyden, C. G. (1970) The convergence of a class of double-rank minimization algorithms 1. general considerations. *IMA Journal of Applied Mathematics*, **6**, 76–90. 3

Cao, B., Shen, D., Sun, J.-T., Yang, Q. and Chen, Z. (2007) Feature selection in a kernel space. In *Proceedings of the 24th International Conference on Machine learning*, 121–128. 2.2

Chen, J., Zhang, C., Kosorok, M. R. and Liu, Y. (2018) Double sparsity kernel learning with automatic variable selection and data extraction. *Statistics and its interface*, **11**, 401. 1, 2.2

Cox, D. and Reid, N. (1989) On the stability of maximum-likelihood estimators of orthogonal parameters. *Canadian Journal of Statistics*, **17**, 229–233. Appendix D

Cox, D. R. and Reid, N. (1987) Parameter orthogonality and approximate conditional inference. *Journal of the Royal Statistical Society: Series B (Methodological)*, **49**, 1–18. Appendix D

Dons, K., Bhattarai, S., Meilby, H., Smith-Hall, C. and Panduro, T. E. (2016) Indirect approach for estimation of forest degradation in non-intact dry forest: modelling biomass loss with Tweedie distributions. *Carbon balance and management*, **11**, 1–10. 1

Dunn, P. K. (2004) Occurrence and quantity of precipitation can be modelled simultaneously. *International Journal of Climatology: A Journal of the Royal Meteorological Society*, **24**, 1231–1239. 1

Dunn, P. K. and Smyth, G. K. (2005) Series evaluation of Tweedie exponential dispersion model densities. *Statistics and Computing*, **15**, 267–280. 3.3

Duvenaud, D. (2014) The kernel cookbook: Advice on covariance functions. *URL https://www.cs.toronto.edu/%7Eduvenaud/cookbook/*. 1

Dzupire, N. C., Ngare, P. and Odongo, L. (2018) A Poisson-gamma model for zero inflated rainfall data. *Journal of Probability and Statistics*, **2018**. 1

El-Shaarawi, A. H., Zhu, R. and Joe, H. (2011) Modelling species abundance using the Poisson–Tweedie family. *Environmetrics*, **22**, 152–164. 1

Fan, J. and Fan, Y. (2008) High dimensional classification using features annealed independence rules. *Annals of statistics*, **36**, 2605. 1

Fletcher, R. (1970) A new approach to variable metric algorithms. *The computer journal*, **13**, 317–322. 3

Fontaine, S., Yang, Y., Qian, W., Gu, Y. and Fan, B. (2020) A unified approach to sparse Tweedie modeling of multisource insurance claim data. *Technometrics*, **62**, 339–356. 1

Foster, S. D. and Bravington, M. V. (2013) A Poisson–gamma model for analysis of ecological non-negative continuous data. *Environmental and ecological statistics*, **20**, 533–552. 1

Frees, E. W., Meyers, G. and Cummings, A. D. (2011) Summarizing insurance scores using a Gini index. *Journal of the American Statistical Association*, **106**, 1085–1098. 5.2, 10

Friedman, J. H. (2001) Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232. 1, 4.2

Gilad-Bachrach, R., Navot, A. and Tishby, N. (2004) Margin based feature selection – theory and algorithms. In *Proceedings of the twenty-first international conference on Machine learning*, 43. 2.2

Giner, G. and Smyth, G. K. (2016) statmod: probability calculations for the inverse Gaussian distribution. *R Journal*, **8**, 339–351. 4

Goldfarb, D. (1970) A family of variable-metric methods derived by variational means. *Mathematics of computation*, **24**, 23–26. 3

Grandvalet, Y. and Canu, S. (2002) Adaptive scaling for feature selection in svms. In *Advances in Neural Information Processing Systems*, vol. 15, 2002. 2.2

Halder, A., Mohammed, S., Chen, K. and Dey, D. (2019) Spatial risk estimation in Tweedie compound Poisson double generalized linear models. *arXiv preprint arXiv:1912.12356*. 1

Hasan, M. M. and Dunn, P. K. (2011) Two Tweedie distributions that are near-optimal for modelling monthly rainfall in Australia. *International Journal of Climatology*, **31**, 1389–1397. 1

Hastie, T. J. and Tibshirani, R. J. (1990) *Generalized additive models*, vol. 43. CRC press. 1

Islam, A. R. M. T., Hasanuzzaman, M., Shammi, M., Salam, R., Bodrud-Doza, M., Rahman, M. M., Mannan, M. A. and Huq, S. (2021) Are meteorological factors enhancing COVID-19 transmission in Bangladesh? Novel findings from a compound Poisson generalized linear modeling approach. *Environmental Science and Pollution Research*, **28**, 11245–11258. 1

Jørgensen, B. (1987) Exponential dispersion models. *Journal of the Royal Statistical Society. Series B (Methodological)*, **49**, 127–162. 2, 2

Jorgensen, B. (1997) *The theory of dispersion models*. CRC Press. 1

Jørgensen, B. and Knudsen, S. J. (2004) Parameter orthogonality and bias adjustment for estimating functions. *Scandinavian Journal of Statistics*, **31**, 93–114. Appendix D

Jørgensen, B. and de Souza, M. C. (1994) Fitting Tweedie's compound Poisson model to insurance claims data. *Scandinavian Actuarial Journal*, **1994**, 69–93. 1, 2, 2.1, 4

Kurz, C. F. (2017) Tweedie distributions for fitting semicontinuous health care utilization cost data. *BMC Medical Research Methodology*, **17**, 171. 1

Lee, S. C. and Lin, S. (2018) Delta boosting machine with application to general insurance. *North American Actuarial Journal*, **22**, 405–425. 1

Li, F., Yang, Y. and Xing, E. (2005) From lasso regression to feature vector machine. *Advances in Neural Information Processing Systems*, **18**, 779–786. 2.2

Lin, Y., Zhang, H. H. et al. (2006) Component selection and smoothing in multivariate nonparametric regression. *The Annals of Statistics*, **34**, 2272–2297. 2.2

Mack, T. (1993) Distribution-free calculation of the standard error of chain ladder reserve estimates.

ASTIN Bulletin: The Journal of the IAA, **23**, 213–225. 5.1

Meyers, G. G. and Shi, P. (2011) Loss reserving data pulled from NAIC Schedule P. URL: https://www.casact.org/publications-research/research/research-resources/loss-reserving-data-pulled-naic-schedule-p. 5.1

Moshitch, D. and Nelken, I. (2014) Using Tweedie distributions for fitting spike count data. *Journal of neuroscience methods*, **225**, 13–28. 1

NAIC (2021) DATA PRODUCTS – SCHEDULE P. URL: https://content.naic.org/prod_serv_idp_sched_p.htm. 5.1

Nocedal, J. and Wright, S. (2006) *Numerical optimization*. Springer Science & Business Media. 3, 3.1, Appendix B, Appendix B, Appendix B

Peters, G. W., Shevchenko, P. V. and Wüthrich, M. V. (2009) Model uncertainty in claims reserving within Tweedie's compound Poisson models. *ASTIN Bulletin: The Journal of the IAA*, **39**, 1–33. 1

Qian, W., Yang, Y. and Zou, H. (2016) Tweedie's compound Poisson model with grouped elastic net. *Journal of Computational and Graphical Statistics*, **25**, 606–625. 1

Rahimi, A., Recht, B. et al. (2007) Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, vol. 3, 5. 6

Rasmussen, C. E. (2003) Gaussian processes in machine learning. In *Summer school on machine learning*, 63–71. Springer. 1

Rudi, A., Camoriano, R. and Rosasco, L. (2015) Less is more: Nyström computational regularization. In *Advances in Neural Information Processing Systems*, 1657–1665. 6

Shanno, D. F. (1970) Conditioning of quasi-Newton methods for function minimization. *Mathematics of computation*, **24**, 647–656. 3

Shi, P. (2014) A copula regression for modeling multivariate loss triangles and quantifying reserving variability. *ASTIN Bulletin: The Journal of the IAA*, **44**, 85–102. 1

— (2016) Insurance ratemaking using a copula-based multivariate Tweedie model. *Scandinavian Actuarial Journal*, **2016**, 198–215. 2

Shi, P., Feng, X. and Boucher, J.-P. (2016) Multilevel modeling of insurance claims using copulas. *The Annals of Applied Statistics*, **10**, 834–863. 1

Shono, H. (2008) Application of the Tweedie distribution to zero-catch data in CPUE analysis. *Fisheries Research*, **93**, 154–162. 1

Smyth, G., Hu, Y., Dunn, P., Phipson, B. and shun Chen, Y. (2021) *Statistical Modeling*. URL: https://cran.r-project.org/package=statmod. R package version 1.4.36. 4

Smyth, G. and Jorgensen, B. (2002) Fitting Tweedie's compound Poisson model to insurance claims data: dispersion modelling. *ASTIN Bulletin*, **32**. 1, 2

Smyth, G. K. (1996) Regression analysis of quantity data with exact zeros. In *Proceedings of the second Australia–Japan workshop on stochastic models in engineering, technology and management*, 572–580. 2

Sriram, K. and Shi, P. (2020) Stochastic loss reserving: A new perspective from a Dirichlet model. *Journal of Risk and Insurance*. 5.1

Taylor, G. (2019) Loss reserving models: Granular and machine learning forms. *Risks*, **7**, 82. 1

Tweedie, M. (1984) An index which distinguishes between some important exponential families. In *Statistics: Applications and New Directions: Proc. Indian Statistical Institute Golden Jubilee International Conference*, 579–604. 1, 2, 2

Vapnik, V. (2013) *The nature of statistical learning theory*. Springer science & business media. 1

Wahba, G. (1990) *Spline models for observational data*, vol. 59. SIAM. 2.1

Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T. and Vapnik, V. (2000) Feature selection for SVMs. 2.2

Wolfe, P. (1971) Convergence conditions for ascent methods. II: Some corrections. *SIAM review*, **13**, 185–188. 3.1

Wood, S. (2021) *Mixed GAM Computation Vehicle with Automatic Smoothness Estimation*. URL: https://cran.r-project.org/package=mgcv. R package version 1.8-36. 4

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

1, 4

Yang, L., Lv, S. and Wang, J. (2016a) Model-free variable selection in reproducing kernel Hilbert space. *The Journal of Machine Learning Research*, **17**, 2885–2908. 1

Yang, Y., Luo, R. and Liu, Y. (2019) Adversarial variational Bayes methods for Tweedie compound Poisson mixed models. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3377–3381. IEEE. 1

Yang, Y., Qian, W. and Zou, H. (2016b) *TDboost: A Boosted Tweedie Compound Poisson Model*. URL: https://CRAN.R-project.org/package=TDboost. R package version 1.2. 4

— (2018) Insurance premium prediction via gradient tree-boosted tweedie compound poisson models. *Journal of Business & Economic Statistics*, **36**, 456–470. 1, 2, 4, 5.2, 5.2

Ye, C., Zhang, L., Han, M., Yu, Y., Zhao, B. and Yang, Y. (2018) Combining predictions of auto insurance claims. *arXiv preprint arXiv:1808.08982*. 5.2

Yip, K. C. and Yau, K. K. (2005) On modeling claim frequency data in general insurance with extra zeros. *Insurance: Mathematics and Economics*, **36**, 153–163. 5.2

Zhang, Y. (2013) Likelihood-based and Bayesian methods for Tweedie compound Poisson linear mixed models. *Statistics and Computing*, **23**, 743–757. 1

Zhou, H., Qian, W. and Yang, Y. (2020) Tweedie gradient boosting for extremely unbalanced zero-inflated data. *Communications in Statistics-Simulation and Computation*, 1–23. 1

Supplemental Materials for "A Kernel Tweedie Compound Poisson Model"

Yi Lian [*], Yi Yang [†], Boxiang Wang [‡], Peng Shi [§], Robert William Platt [¶]

August 30, 2021

Appendix

Appendix A Line-search algorithms

Appendix A.1 Bisection line-search for BFGS

This line-search is performed in each (inverse) BFGS update iteration. It aims to find an appropriate positive step size t that satisfies the Wolfe conditions in (18).

Appendix A.2 Backtracking line-search for gradient descent

This line-search is performed in each gradient descent update iteration. It aims to find an appropriate positive step size t that satisfies the Armijo-Goldstein condition

[*]Department of Epidemiology, Biostatistics and Occupational Health, McGill University
[†]Department of Mathematics and Statistics, McGill University (yi.yang6@mcgill.ca)
[‡]Department of Statistics and Actuarial Science, University of Iowa
[§]Risk and Insurance Department, Wisconsin School of Business, University of Wisconsin-Madison
[¶]Corresponding author, Department of Epidemiology, Biostatistics and Occupational Health, McGill University

Algorithm 4: Bisection line-search for the (inverse) BFGS

Input: $\boldsymbol{\alpha}, \mathbf{p}$
Output: t
Constants: $c_1 = 10^{-4}, c_2 = 0.9, a = 0$

Initialization: $t = 1$, phase = A, accept = False;
repeat phase A
  if *Condition 1 holds* **then**
    if *Condition 2 holds* **then**
      accept = True
    else
      $t = 2t$
    end
  else
    phase = B
    exit;
  end
until *accept*;
if *phase = B* **then**
  $b = t$
  repeat phase B
    $t_{old} = t$
    $t = (a + b)/2$
    if $t_{old} = t$ **then**
      cannot find proper t
      exit;
      /* exit BFGS */
      /* switch to GD */
    end
    if *Condition 1 holds* **then**
      if *Condition 2 holds* **then**
        accept = True
      else
        $a = t$
      end
    else
      $b = t$
    end
  until *accept*;
end

$$g(\boldsymbol{\xi} - t\nabla g(\boldsymbol{\xi})) \leq g(\boldsymbol{\xi}) - ct\|\nabla g(\boldsymbol{\xi})\|_2^2$$

where $\boldsymbol{\xi}$ is the parameter of interest ($\boldsymbol{\alpha}$ or $\mathbf{w}$ in our case) and $c \in (0, 1/2]$ is some constant.

Algorithm 5: Backtracking line-search for gradient descent

Input: $\boldsymbol{\xi}$
Output: t
Constants: $c = 0.5$
Initialization: $t = 1$, accept = False;
repeat
  if $g(\boldsymbol{\xi} - t\nabla g(\boldsymbol{\xi})) \leq g(\boldsymbol{\xi}) - ct\|\nabla g(\boldsymbol{\xi})\|_2^2$ **then**
    accept = True
  else
    $t = 0.9t$
  end
until *accept*;

Appendix B Proof of Theorem 1

Proof. According to Theorem 6.5 (Nocedal and Wright, 2006), in order to show **the global convergence of BFGS** in our algorithm, we only need to check the following two conditions (Assumption 6.1 Nocedal and Wright, 2006) are satisfied:

1. The objective function g is twice continuously differentiable.

2. There exist positive constants m and M such that, for all $\boldsymbol{\alpha}$,

$$m\mathbf{I}_n \preceq \nabla^2 g(\boldsymbol{\alpha}) \preceq M\mathbf{I}_n.$$

   where $\mathbf{I}_n$ is an $n \times n$ identity matrix.

Since Algorithm 1 is descending along its iterations thus we can restrict the domain of $\boldsymbol{\alpha}$ to the sublevel set $\mathcal{L}_0 = \left\{\boldsymbol{\alpha} \in \mathbb{R}^n : g(\boldsymbol{\alpha}) \leq g(\boldsymbol{\alpha}^{(0)})\right\}$. Since g is a convex function, set $\mathcal{L}_0$ is convex compact. Without loss of generality, assume not all y_i's are zero. Define $\tau_i = \mathbf{K}_i^\top \boldsymbol{\alpha}$ for $i = 1, \ldots, n$.

It follows that the set

$$\mathcal{C}_0 = \left\{ \boldsymbol{\tau} = (\tau_1, \ldots, \tau_n)^\top : \boldsymbol{\alpha} \in \mathcal{L}_0 \right\}$$

is convex compact. Therefore for all $\boldsymbol{\alpha} \in \mathcal{L}_0$, η_i is bounded by $\eta_{\max}$, where

$$\eta_{\max} = \max_{1 \le i \le n} \sup_{\boldsymbol{\alpha} \in \mathcal{L}_0} |\eta_i| < \infty.$$

Also y_i's are bounded by $v_{\max} = \max_{1 \le i \le n} v_i$ and $y_{\max} = \max_{1 \le i \le n} y_i$. Let

$$\bar{w}_i = v_i \left((\rho - 1) y_i e^{(1-\rho)\tau_i} + (2 - \rho) e^{(2-\rho)\tau_i} \right)$$

Note that $\bar{w}_i$ is bounded by

$$\max_{1 \le i \le n} \sup_{\boldsymbol{\alpha} \in \mathcal{L}_0} |\bar{w}_i| \le v_{\max} \left(y_{\max} (\rho - 1) e^{(\rho-1)\tau_{\max}} + (2 - \rho) e^{(2-\rho)\tau_{\max}} \right) \equiv w_{\max}.$$

We can see that

$$\begin{aligned} \nabla^2 g(\boldsymbol{\alpha}) &= \mathbf{K} \mathrm{diag}\left[\bar{w}_1, \bar{w}_2, \ldots, \bar{w}_n \right] \mathbf{K} + \lambda \mathbf{K} \\ &\preceq (w_{\max} \Lambda_{\max}(\mathbf{K}\mathbf{K}) + \Lambda_{\max}(\mathbf{K})) \mathbf{I}_n, \qquad \forall \boldsymbol{\alpha} \in \mathcal{L}_0. \end{aligned}$$

where $\Lambda_{\max}(\mathbf{A})$ represents the largest eigenvalue of matrix $\mathbf{A}$. Thus $g(\boldsymbol{\alpha})$ is strongly smooth on the sublevel set $\mathcal{L}_0$. We can also show that $g(\boldsymbol{\alpha})$ is strongly convex on $\mathcal{L}_0$. It can be shown that $\bar{w}_i$ can be lower-bounded on $\mathcal{L}_0$,

$$\bar{w}_i \ge \left(\frac{\rho - 1}{2 - \rho} \right)^{3-2\rho} v_i \left(y_i \right)^{2-\rho} I\left(y_i > 0 \right) + (2 - \rho) e^{-(2-\rho)\eta_{\max}} I\left(y_i = 0 \right) > 0$$

for all $\boldsymbol{\alpha} \in \mathcal{L}_0$ and $i = 1, \ldots, n$. Let

$$w_{\min} = \min \left\{ \left(\frac{\rho - 1}{2 - \rho} \right)^{3-2\rho} \min_{i : y_i > 0} w_i \left(y_i \right)^{2-\rho}, (2 - \rho) e^{-(2-\rho)\eta_{\max}} \right\}.$$

We see that $\bar{w}_i \geq w_{\min} > 0$. Therefore

$$\begin{aligned}\nabla^2 g(\boldsymbol{\alpha}) &= \mathbf{K}\text{diag}\left[\bar{w}_1, \bar{w}_2, \ldots, \bar{w}_n\right]\mathbf{K} + \lambda\mathbf{K} \\ &\succeq (w_{\min}\Lambda_{\min}(\mathbf{KK}) + \Lambda_{\min}(\mathbf{K}))\mathbf{I}_n, \qquad \forall \boldsymbol{\alpha} \in \mathbb{R}^n.\end{aligned}$$

This shows that $g(\boldsymbol{\alpha})$ is strongly convex. We have proved that Assumption 6.1 in Theorem 6.5 (Nocedal and Wright, 2006) holds so that Algorithm 1 has global convergence.

By Theorem 6.6 (Nocedal and Wright, 2006), in order to show that the update $\boldsymbol{\alpha}^{(k)}$ generated by Algorithm 1 converges to $\boldsymbol{\alpha}^*$ at a superlinear rate, we only need to show that g is twice continuously differentiable and that the Hessian matrix $\nabla^2 g$ is Lipschitz continuous (Assumption 6.2 Nocedal and Wright, 2006), i.e. for all $\boldsymbol{\alpha}, \boldsymbol{\alpha}' \in \text{dom} g$, there exists a positive constant L such that,

$$\left\|\nabla^2 g(\boldsymbol{\alpha}) - \nabla^2 g(\boldsymbol{\alpha}')\right\|_2 \leq L\left\|\boldsymbol{\alpha} - \boldsymbol{\alpha}'\right\|_2,$$

where the norm applied to the matrix is the spectral norm.

We consider a vector-valued function $h(t) : \mathbb{R} \to \mathbb{R}^n$ satisfying $h_{\mathbf{b}}(t) = \mathbf{b}^\top \nabla^2 f(\boldsymbol{\alpha} + t(\boldsymbol{\alpha}' - \boldsymbol{\alpha}))$, then by the mean value theorem

$$\begin{aligned}\mathbf{b}^\top[\nabla^2 g(\boldsymbol{\alpha}) - \nabla^2 g(\boldsymbol{\alpha}')] &= \frac{h_{\mathbf{b}}(1) - h_{\mathbf{b}}(0)}{1 - 0} \\ &= h'_{\mathbf{b}}(\tilde{t}) \qquad \text{(mean value theorem, } \tilde{t} \in (0,1)) \\ &= \begin{bmatrix} \sum_i \sum_j \frac{\partial^3 g(\widetilde{\boldsymbol{\alpha}})}{\partial\alpha_1\partial\alpha_i\partial\alpha_j} b_i(\alpha'_j - \alpha_j) \\ \vdots \\ \sum_i \sum_j \frac{\partial^3 g(\widetilde{\boldsymbol{\alpha}})}{\partial\alpha_n\partial\alpha_i\partial\alpha_j} b_i(\alpha'_j - \alpha_j) \end{bmatrix}. \qquad (\widetilde{\boldsymbol{\alpha}} = \boldsymbol{\alpha} + \tilde{t}(\boldsymbol{\alpha}' - \boldsymbol{\alpha})) \quad (28)\end{aligned}$$

In the sublevel set $\mathcal{L}_0$, the values of third derivatives of g in (28) can be upper-bounded

$$\left|\frac{\partial^3 g(\widetilde{\boldsymbol{\alpha}})}{\partial\alpha_1\partial\alpha_i\partial\alpha_j}\right| \leq D, \qquad (29)$$

where $D > 0$ is a constant. Therefore the L_2 norm of the vector $\mathbf{b}^\top[\nabla^2 g(\boldsymbol{\alpha}) - \nabla^2 g(\boldsymbol{\alpha}')]$ can also be upper-bounded

$$\begin{aligned}\|\mathbf{b}^\top[\nabla^2 g(\boldsymbol{\alpha}) - \nabla^2 g(\boldsymbol{\alpha}')]\|_2 &\leq D\sqrt{n}\Big|\sum_i \sum_j b_i(\alpha'_j - \alpha_j)\Big| \\ &\leq D\sqrt{n}\cdot n\|\mathbf{b}\|_2\|\boldsymbol{\alpha}' - \boldsymbol{\alpha}\|_2.\end{aligned}$$

The above inequality indicates that $\nabla^2 g$ is Lipschitz continuous, since that

$$\begin{aligned}\left\|\nabla^2 g(\boldsymbol{\alpha}) - \nabla^2 g(\boldsymbol{\alpha}')\right\|_2 &= \max_{\|\mathbf{b}\|_2=1} \|\mathbf{b}^\top[\nabla^2 g(\boldsymbol{\alpha}) - \nabla^2 g(\boldsymbol{\alpha}')]\|_2 \\ &\leq \max_{\|\mathbf{b}\|_2=1} D\sqrt{n}\cdot n\|\mathbf{b}\|_2\|\boldsymbol{\alpha}' - \boldsymbol{\alpha}\|_2 \\ &= D\sqrt{n}\cdot n\|\boldsymbol{\alpha}' - \boldsymbol{\alpha}\|_2,\end{aligned}$$

where the first line follows by the definition of the spectral norm. Therefore Assumption 6.1 in Theorem 6.5 (Nocedal and Wright, 2006) holds. □

Appendix C The derivative of the SKtweedie objective function

The objective function is

$$\begin{aligned}g(\boldsymbol{\alpha}, \mathbf{w}) &= l_1 + l_2 + p_1 + p_2 \\ &= \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i \exp\left[-(\rho-1)\mathbf{K}(\mathbf{w})_i^\top \boldsymbol{\alpha}\right]}{\rho - 1}\right) \quad \ldots\ldots\ldots\ldots\ldots\ldots (l_1) \\ &+ \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\exp\left[(2-\rho)\mathbf{K}(\mathbf{w})_i^\top \boldsymbol{\alpha}\right]}{2 - \rho}\right) \quad \ldots\ldots\ldots\ldots\ldots\ldots (l_2) \\ &+ \lambda_1 \boldsymbol{\alpha}^\top \mathbf{K}(\mathbf{w})\boldsymbol{\alpha} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (p_1) \\ &+ \lambda_2 \mathbf{1}^\top \mathbf{w} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (p_2) \\ \text{s.t. } & w_j \in [0, 1],\ j = 1, \ldots, p,\end{aligned}$$

where

$$\mathbf{K}(\mathbf{w}) = \begin{bmatrix} \mathbf{K}(\mathbf{w})_1 \\ \mathbf{K}(\mathbf{w})_2 \\ \vdots \\ \mathbf{K}(\mathbf{w})_n \end{bmatrix} = \begin{bmatrix} \mathbf{K}(\mathbf{w})_{11} & \mathbf{K}(\mathbf{w})_{12} & \cdots & \mathbf{K}(\mathbf{w})_{1n} \\ \mathbf{K}(\mathbf{w})_{21} & \mathbf{K}(\mathbf{w})_{22} & \cdots & \mathbf{K}(\mathbf{w})_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{K}(\mathbf{w})_{n1} & \mathbf{K}(\mathbf{w})_{n2} & \cdots & \mathbf{K}(\mathbf{w})_{nn} \end{bmatrix}$$

$$= \begin{bmatrix} K(\mathbf{w} \odot \mathbf{x}_1, \mathbf{w} \odot \mathbf{x}_1) & K(\mathbf{w} \odot \mathbf{x}_1, \mathbf{w} \odot \mathbf{x}_2) & \cdots & K(\mathbf{w} \odot \mathbf{x}_1, \mathbf{w} \odot \mathbf{x}_n) \\ K(\mathbf{w} \odot \mathbf{x}_2, \mathbf{w} \odot \mathbf{x}_1) & K(\mathbf{w} \odot \mathbf{x}_2, \mathbf{w} \odot \mathbf{x}_2) & \cdots & K(\mathbf{w} \odot \mathbf{x}_2, \mathbf{w} \odot \mathbf{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ K(\mathbf{w} \odot \mathbf{x}_n, \mathbf{w} \odot \mathbf{x}_1) & K(\mathbf{w} \odot \mathbf{x}_n, \mathbf{w} \odot \mathbf{x}_2) & \cdots & K(\mathbf{w} \odot \mathbf{x}_n, \mathbf{w} \odot \mathbf{x}_n) \end{bmatrix},$$

and $K(\cdot, \cdot)$ is the RBF kernel function with tuning parameter σ. For $i, j = 1, 2, \ldots, n$,

$$\mathbf{K}(\mathbf{w})_{ij} = k(\mathbf{w} \odot \mathbf{x}_i, \mathbf{w} \odot \mathbf{x}_j) = \exp(-\sigma \cdot \|\mathbf{w} \odot \mathbf{x}_i - \mathbf{w} \odot \mathbf{x}_j\|_2^2).$$

For clarity, divide the objective function into four parts $g(\boldsymbol{\alpha}, \mathbf{w}) = l_1 + l_2 + p_1 + p_2$ and derive individually. First, we take derivative of l_1 with respect to $\mathbf{w}$,

$$\frac{\partial l_1}{\partial \mathbf{w}} = \frac{1}{n} \sum_{i=1}^{n} \frac{\partial l_1}{\partial \mathbf{K}(\mathbf{w})_i} \cdot \frac{\partial \mathbf{K}(\mathbf{w})_i}{\partial \mathbf{w}},$$

where

$$\frac{\partial l_1}{\partial \mathbf{K}(\mathbf{w})_i} = -y_i \exp\left[-(\rho - 1)\mathbf{K}(\mathbf{w})_i^\top \boldsymbol{\alpha}\right] \cdot \boldsymbol{\alpha} = \eta_i \cdot \boldsymbol{\alpha} \in \mathbb{R}^n,$$

with $\eta_i = -y_i \exp\left[-(\rho - 1)\mathbf{K}(\mathbf{w})_i^\top \boldsymbol{\alpha}\right]$ is a scalar, and

$$\frac{\partial \mathbf{K}(\mathbf{w})_i}{\partial \mathbf{w}} = \frac{\partial \left[\mathbf{K}(\mathbf{w})_{i1}, \mathbf{K}(\mathbf{w})_{i2}, \ldots, \mathbf{K}(\mathbf{w})_{in}\right]}{\partial \mathbf{w}} \in \mathbb{R}^{n \times p},$$

with

$$\begin{aligned}\frac{\partial \mathbf{K}(\mathbf{w})_{ij}}{\partial \mathbf{w}} &= \frac{\partial k(\mathbf{w}\odot\mathbf{x}_i, \mathbf{w}\odot\mathbf{x}_j)}{\partial \mathbf{w}} \\ &= \frac{\partial \exp(-\sigma\cdot\|\mathbf{w}\odot\mathbf{x}_i - \mathbf{w}\odot\mathbf{x}_j\|_2^2)}{\partial \mathbf{w}} \\ &= \exp(-\sigma\cdot\|\mathbf{w}\odot\mathbf{x}_i - \mathbf{w}\odot\mathbf{x}_j\|_2^2)\cdot(-2\sigma)\cdot(\mathbf{x}_i-\mathbf{x}_j)\odot(\mathbf{x}_i-\mathbf{x}_j)\odot\mathbf{w} \\ &= c_{ij}\cdot(\mathbf{x}_i-\mathbf{x}_j)\odot(\mathbf{x}_i-\mathbf{x}_j)\odot\mathbf{w},\end{aligned}$$

for the scalar $c_{ij} = -2\sigma\cdot\exp(-\sigma\cdot\|\mathbf{w}\odot\mathbf{x}_i - \mathbf{w}\odot\mathbf{x}_j\|_2^2)$. Therefore,

$$\frac{\partial \mathbf{K}(\mathbf{w})_i}{\partial \mathbf{w}} = \begin{bmatrix} c_{i1}\cdot(\mathbf{x}_i-\mathbf{x}_1)\odot(\mathbf{x}_i-\mathbf{x}_1)\odot\mathbf{w} \\ c_{i2}\cdot(\mathbf{x}_i-\mathbf{x}_2)\odot(\mathbf{x}_i-\mathbf{x}_2)\odot\mathbf{w} \\ \vdots \\ c_{in}\cdot(\mathbf{x}_i-\mathbf{x}_n)\odot(\mathbf{x}_i-\mathbf{x}_n)\odot\mathbf{w} \end{bmatrix}.$$

Put it together,

$$\frac{\partial \ell_1}{\partial \mathbf{w}} = \frac{1}{n}\sum_{i=1}^{n}\eta_i\cdot\boldsymbol{\alpha}^\top\cdot\begin{bmatrix} c_{i1}\cdot(\mathbf{x}_i-\mathbf{x}_1)\odot(\mathbf{x}_i-\mathbf{x}_1)\odot\mathbf{w} \\ c_{i2}\cdot(\mathbf{x}_i-\mathbf{x}_2)\odot(\mathbf{x}_i-\mathbf{x}_2)\odot\mathbf{w} \\ \vdots \\ c_{in}\cdot(\mathbf{x}_i-\mathbf{x}_n)\odot(\mathbf{x}_i-\mathbf{x}_n)\odot\mathbf{w} \end{bmatrix} \in \mathbb{R}^p.$$

Next, we derive l_2. Similar to the above,

$$\begin{aligned}\frac{\partial l_2}{\partial \mathrm{w}} &= \frac{1}{n}\sum_{i=1}^{n}\frac{\partial l_2}{\partial \mathbf{K}(\mathbf{w})_i}\cdot\frac{\partial \mathbf{K}(\mathbf{w})_i}{\partial \mathbf{w}} \\ &= \frac{1}{n}\sum_{i=1}^{n}\zeta_i\cdot\boldsymbol{\alpha}^\top\cdot\begin{bmatrix} c_{i1}\cdot(\mathbf{x}_i-\mathbf{x}_1)\odot(\mathbf{x}_i-\mathbf{x}_1)\odot\mathbf{w} \\ c_{i2}\cdot(\mathbf{x}_i-\mathbf{x}_2)\odot(\mathbf{x}_i-\mathbf{x}_2)\odot\mathbf{w} \\ \vdots \\ c_{in}\cdot(\mathbf{x}_i-\mathbf{x}_n)\odot(\mathbf{x}_i-\mathbf{x}_n)\odot\mathbf{w} \end{bmatrix},\end{aligned}$$

where $\zeta_i = \exp\left[(2-\rho)\mathbf{K}(\mathbf{w})_i^\top \boldsymbol{\alpha}\right], i = 1, 2, \ldots, n$.

Next, take the derivative of the first penalty p_1 w.r.t. $\mathbf{w}$,

$$\begin{aligned}
\frac{\partial p_1}{\partial \mathbf{w}} &= \lambda_1 \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\partial p_1}{\partial \mathbf{K}(\mathbf{w})_{ij}} \cdot \frac{\partial \mathbf{K}(\mathbf{w})_{ij}}{\partial \mathbf{w}} \\
&= \lambda_1 \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j \frac{\partial \mathbf{K}(\mathbf{w})_{ij}}{\partial \mathbf{w}} \\
&= \lambda_1 \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j c_{ij} \cdot (\mathbf{x}_i - \mathbf{x}_j) \odot (\mathbf{x}_i - \mathbf{x}_j) \odot \mathbf{w}.
\end{aligned}$$

Finally, $\partial p_2 / \partial \mathbf{w}$ has the following form,

$$\frac{\partial p_2}{\partial \mathbf{w}} = \lambda_2.$$

Note that the gradient is scaled by the weights except for the last term, thus $\frac{\partial g(\boldsymbol{\alpha}, \mathbf{w})}{\partial w_j} = \lambda_2, \forall w_j = 0$.

Appendix D Parameter orthogonality

Following (5), $g(y|\mu, \phi, \rho)$ is the density function, for y, we have $\int g(y|\mu, \phi, \rho) dy = 1$. Therefore

$$\begin{aligned}
0 &= \frac{\partial}{\partial \mu} \int g(y|\mu, \phi, \rho) dy \\
&= \int \frac{g(y|\mu, \phi, \rho)}{g(y|\mu, \phi, \rho)} \frac{\partial g(y|\mu, \phi, \rho)}{\partial \mu} dy \\
&= \int g(y|\mu, \phi, \rho) \frac{\partial \log g(y|\mu, \phi, \rho)}{\partial \mu} dy \\
&= \mathbb{E}_Y \left[\frac{\partial \log g(y|\mu, \phi, \rho)}{\partial \mu} \right].
\end{aligned}$$

Since

$$g(y|\mu, \phi, \rho) = a(y, \phi, \rho) \exp\left\{ \frac{1}{\phi} \left(\frac{y\mu^{1-\rho}}{1-\rho} - \frac{\mu^{2-\rho}}{2-\rho} \right) \right\},$$

the density satisfies

$$\frac{\partial \log g\left(y|\mu,\rho,\phi\right)}{\partial \mu} = \frac{y-\mu}{\phi\mu^{\rho}}.$$

Therefore

$$\begin{aligned}
\mathbb{E}\left[\frac{\partial^2 \log g(y|\mu,\phi,\rho)}{\partial\mu\partial\phi}\right] &= \mathbb{E}\left[\frac{\partial}{\partial\phi}\left(\frac{y-\mu}{\phi\mu^{\rho}}\right)\right] \\
&= \mathbb{E}\left[-\frac{1}{\phi^2}\cdot\frac{y-\mu}{\mu^{\rho}}\right] \\
&= -\frac{1}{\phi}\mathbb{E}\left[\frac{y-\mu}{\phi\mu^{\rho}}\right] \\
&= -\frac{1}{\phi}\mathbb{E}\left[\frac{\partial \log g(y|\mu,\phi,\rho)}{\partial\mu}\right] \\
&= 0,
\end{aligned}$$

also

$$\begin{aligned}
\mathbb{E}\left[\frac{\partial^2 \log g(y|\mu,\phi,\rho)}{\partial\mu\partial\rho}\right] &= \mathbb{E}\left[\frac{\partial}{\partial\rho}\left(\frac{y-\mu}{\phi\mu^{\rho}}\right)\right] \\
&= \mathbb{E}\left[\log\mu\cdot\frac{y-\mu}{\phi\mu^{\rho}}\right] \\
&= \log\mu\cdot\mathbb{E}\left[\frac{y-\mu}{\phi\mu^{\rho}}\right] \\
&= \log\mu\cdot\mathbb{E}\left[\frac{\partial \log g(y|\mu,\phi,\rho)}{\partial\mu}\right] \\
&= 0.
\end{aligned}$$

Therefore μ is orthogonal to both ϕ and ρ (Cox and Reid, 1987, 1989; Jørgensen and Knudsen, 2004). The statistical consequences of this orthogonality is that the maximum likelihood estimates $\hat{\mu}$ is asymptotically independent to $\hat{\phi}$ and $\hat{\rho}$.

Appendix E Additional tables and figures

Table A1: The mean and standard errors of MADs, $\widehat{\rho}$ and $\widehat{\phi}$ based on 20 independent replications. True $\rho = 1.5$ and true $\phi = 0.5$

Model	MAD	$\widehat{\rho}$	$\widehat{\phi}$
1	0.096 (0.004)	1.503 (0.0126)	0.497 (0.008)
2	0.088 (0.003)	1.441 (0.024)	0.505 (0.013)

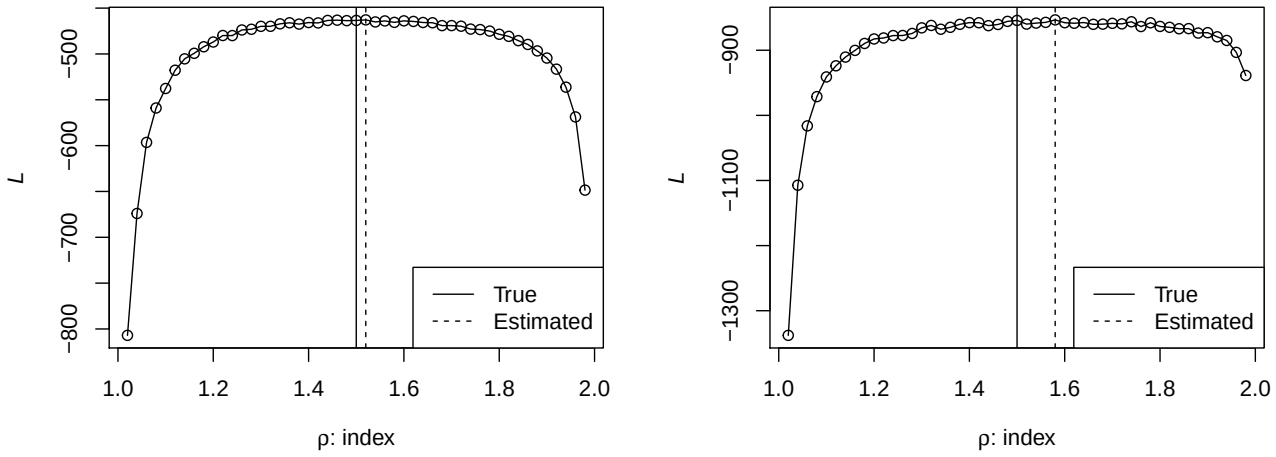

Figure A1: The profile likelihood of ρ from a sample run. Model 1 (left): true $\rho = 1.5$, $\widehat{\rho} = 1.52$; Model 2 (right): true $\rho = 1.5$, $\widehat{\rho} = 1.58$.

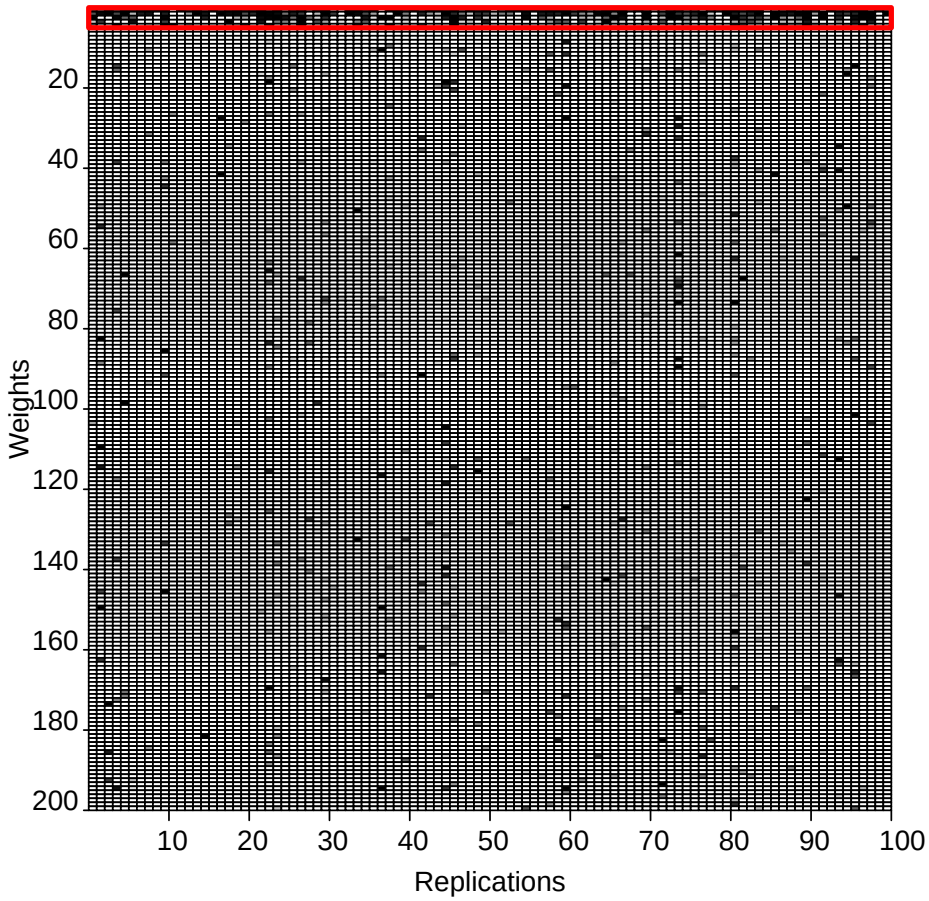

Figure A2: Variable selection results using the SKtweedie with Gaussian RBF kernel ($p = 200$)

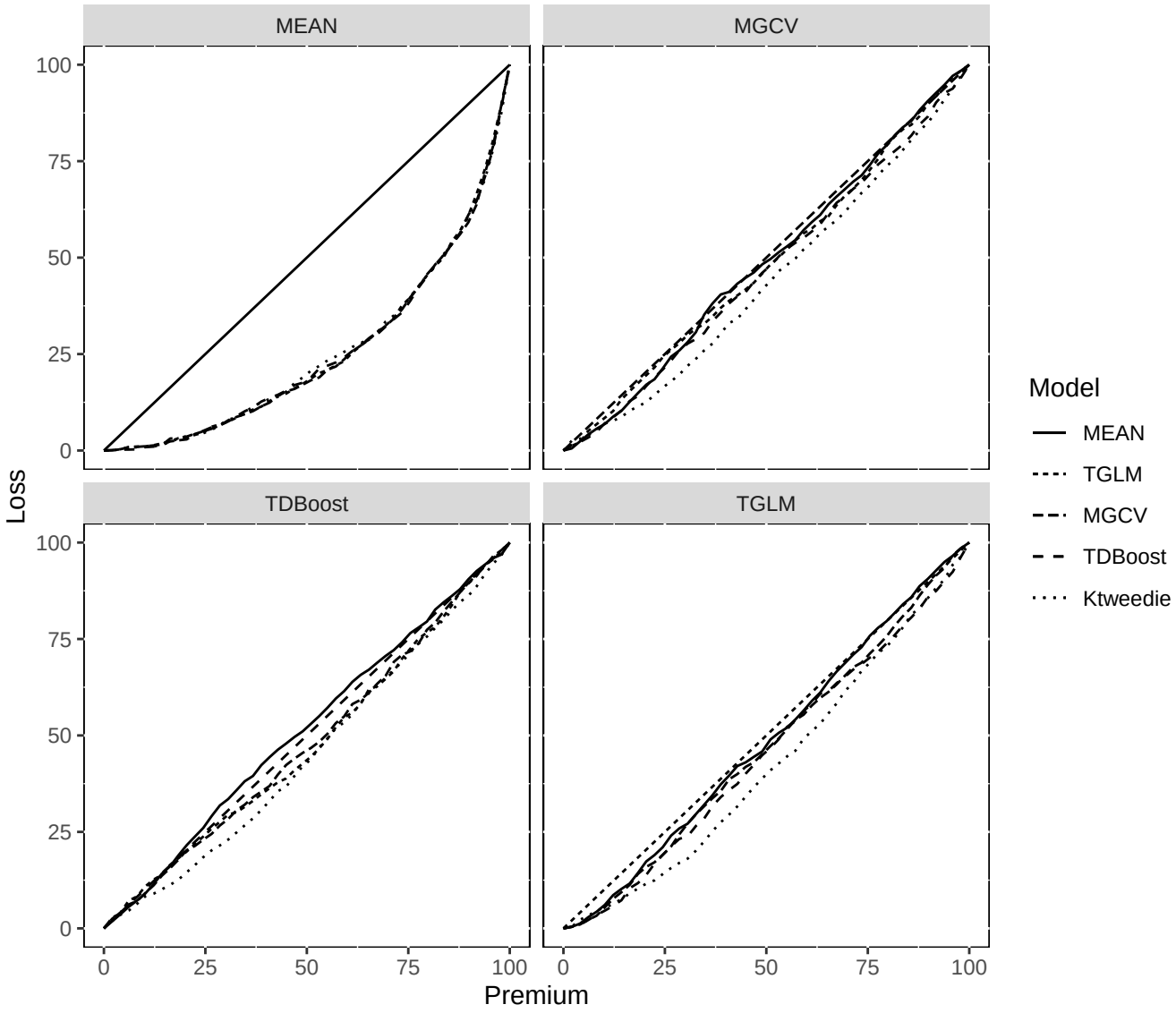

Figure A3: The ordered Lorenz curves for the auto-insurance claim data.