Technometrics

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/utch20

A Tweedie Compound Poisson Model in Reproducing Kernel Hilbert Space

Yi Lian, Archer Yi Yang, Boxiang Wang, Peng Shi & Robert William Platt

To cite this article: Yi Lian, Archer Yi Yang, Boxiang Wang, Peng Shi & Robert William Platt (2023) A Tweedie Compound Poisson Model in Reproducing Kernel Hilbert Space, Technometrics, 65:2, 281-295, DOI: 10.1080/00401706.2022.2156615

To link to this article: https://doi.org/10.1080/00401706.2022.2156615

TECHNOMETRICS
2023, VOL. 65, NO. 2, 281–295
https://doi.org/10.1080/00401706.2022.2156615

Taylor & Francis
Taylor & Francis Group

A Tweedie Compound Poisson Model in Reproducing Kernel Hilbert Space

Yi Lian[a], Archer Yi Yang[b], Boxiang Wang[c], Peng Shi[d], and Robert William Platt[a]

[a]Department of Epidemiology, Biostatistics and Occupational Health, McGill University, Montreal, Quebec, Canada; [b]Department of Mathematics and Statistics, McGill University, Montreal, Quebec, Canada; [c]Department of Statistics and Actuarial Science, University of Iowa, Iowa City, IA; [d]Risk and Insurance Department, Wisconsin School of Business, University of Wisconsin-Madison, Madison, WI

ABSTRACT

Tweedie models can be used to analyze nonnegative 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 on today's complex data problems with challenging characteristics such as nonlinear effects, high-order interactions, high-dimensionality and sparsity. In this article, we propose a kernel Tweedie model, Ktweedie, and its sparse variant, SKtweedie, that can simultaneously address the above challenges. Specifically, nonlinear effects and high-order interactions can be flexibly represented through a wide range of kernel functions, which is fully learned from the data; In addition, while the Ktweedie can handle high-dimensional data, the SKtweedie with integrated variable selection can further improve the interpretability. 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. Overall, the Ktweedie and SKtweedie outperform existing Tweedie models when there exist nonlinear effects and high-order interactions, particularly when the dimensionality is high relative to the sample size. The model is implemented in an efficient and user-friendly R package `ktweedie` (*https://cran.r-project.org/package=ktweedie*).

ARTICLE HISTORY
Received January 2022
Accepted November 2022

KEYWORDS
Insurance claims data; Loss-reserving; Ratemaking; Sparse kernel methods; Zero inflated data

1. Introduction

The Tweedie compound Poisson distribution (Tweedie 1984), derived from a Poisson sum of gamma variables, is a member of the exponential dispersion (ED) family (Jorgensen 1997). What distinguishes the distribution from other members of the exponential family is that it incorporates a probability mass at zero into an elsewhere continuous distribution. Because of this unique feature, the Tweedie distribution naturally handles semi-continuous outcomes and 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, Ngare and Odongo 2018), the biomass of a certain species in ecology (El-Shaarawi, Zhu, and Joe 2011; Foster and Bravington 2013), the total catch in fishery (Shono 2008), and the aggregate losses in insurance (Smyth and Jorgensen 2002; Shi, Feng, and Boucher 2016). The medical research community has also found its use in various ways (Moshitch and Nelken 2014; Kurz 2017; Islam et al. 2021).

Various statistical models have been developed based on the Tweedie compound Poisson 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, Feng, and Boucher 2016) and Tweedie mixed models (Zhang 2013; Yang, Luo, and Liu 2019) to more flexible nonparametric approaches such as regression trees (Yang, Qian, and Zou 2018; Lee and Lin 2018; Zhou, Qian, and Yang 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. To improve the flexibility, 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 misspecification, Yang, Qian, and Zou (2018) and Zhou, Qian, and Yang (2020) proposes a tree-based gradient boosting algorithm (Friedman 2001) for the Tweedie model (TDboost). It features the ability to handle nonlinear effects and high-order interactions of the predictors. However, none of the aforementioned methods can directly handle high-dimensional data. As the number of predictors in the data increases and potentially surpasses the number of observations, the prediction performance of those methods will deteriorate rapidly. Although some recent works can work with high-dimensional data (Qian, Yang, and Zou 2016; Fontaine et al. 2020), they are limited within the linear model framework.

In this article, we propose Ktweedie, a fully nonparametric Tweedie model in reproducing kernel Hilbert space (RKHS).

CONTACT Robert William Platt *robert.platt@mcgill.ca* Department of Epidemiology, Biostatistics and Occupational Health, McGill University.
Supplementary materials for this article are available online. Please go to *www.tandfonline.com/r/TECH*.

Table 1. A comparison of characteristics of different Tweedie models.

	TGLM	TGAM	TDboost	Ktweedie	SKtweedie
Nonlinearity		✓	✓	✓	✓
High-order interactions			✓	✓	✓
High-dimensionality	✓*			✓	✓
Sparsity	✓*				✓

NOTE: *requires sparse regularization, see Qian, Yang, and Zou (2016).

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 model structure is fully learned from the data, 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 can handle high-dimensional data and features an automatic variable selection procedure for removing redundant predictors. Under the high-dimensional setting, the error accumulated from noninformative or noisy predictors can more easily undermine the prediction performance (Fan and Fan 2008). This drives us to further incorporate variable selection mechanism into the Ktweedie model. 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 resulting approach is inspired by Allen (2013), Yang, Lv, and Wang (2016), and Chen et al. (2018) and is referred to as SKtweedie in this article. Compared with the other existing methods discussed in the previous paragraph, our work is the only one that can address all the challenges (i.e., nonlinearity, high-order interactions, high-dimensionality and sparsity) simultaneously. Table 1 summarizes the comparison of different Tweedie models.

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 nonzero probability at zero along with a positive continuous support, which makes it a natural choice for insurance applications (Ohlsson and Johansson 2010; Taylor and McGuire 2016). 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, Feng, and Boucher (2016), Halder et al. (2019) for ratemaking applications and Peters, Shevchenko, and Wüthrich (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 article 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 article. Technical details and additional empirical results are provided in the supplementary materials.

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 iid gamma distributed $Z_d \sim \text{Gamma}(\alpha, \gamma)$. Define Y as the conditional sum of N iid gamma random variables standardized by the weight:

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

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 dth 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; Smyth and Jorgensen 2002; Shi 2016). Note that $Y = 0$ if and only if $N = 0$, thus, Y has a probability mass at 0, that is, $\Pr(Y = 0) = \Pr(N = 0) = \exp(-v\lambda)$. 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

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

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

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

where $\dot{\kappa}(\theta)$ and $\ddot{\kappa}(\theta)$ are the first and second order derivative of $\kappa(\theta)$, respectively. For the Tweedie distribution, θ, $\kappa(\theta)$ and

φ have the specific forms:

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

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 $\mathrm{Tw}(\mu, \phi/\nu, \rho)$ can be written as

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

where the exact form of $a(\cdot)$ can be found in Section 3 of Yang, Qian, and Zou (2018). The mean and variance relationship of the Tweedie distribution becomes $\mathrm{var}(Y) = \phi/\nu \cdot \mu^{\rho}$.

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 ith 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 $Y_i/v_i \sim \mathrm{Tw}(\mu_i, \phi/v_i, \rho)$. 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

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

where $\|f\|_{\mathcal{H}_K}^2$ is a generalized Tikhonov regularization defined in the Hilbert space. The optimization problem for f is infinite-dimensional, and f does not belong to some specific parametric family. 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 ith 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. This result allows f to have a "parametric form" with the finite-dimensional representation, the dimension is dependent of sample size n. We consider commonly used kernel functions including the Gaussian radial basis function (RBF) kernel $K(\mathbf{x}_i,\mathbf{x}_{i'}) = \exp(-\sigma\|\mathbf{x}_i - \mathbf{x}_{i'}\|_2^2)$ and the Laplace kernel $K(\mathbf{x}_i,\mathbf{x}_{i'}) = \exp(-\sigma\|\mathbf{x}_i - \mathbf{x}_{i'}\|_2)$, where σ is some kernel parameter. 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 optimizing (10) will be discuss in Section 3.1.

One important issue in kernel-based learning is the choice of the kernel functions based on the data characteristics. For example, the Gaussian RBF kernel can be used to model smooth functions, the periodic kernel can be used to model periodic functions and the string kernel can be used to analyze text data. We direct the interested readers to two comprehensive online tutorials Duvenaud (2014) and Pedregosa et al. (2011) for the use and combined use of different kernel functions.

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, Navot, and Tishby 2004; Li, Yang, and Xing 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 and Zhang 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 objective function (10) in two aspects: First, variable weights are used in the kernel matrix such that

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

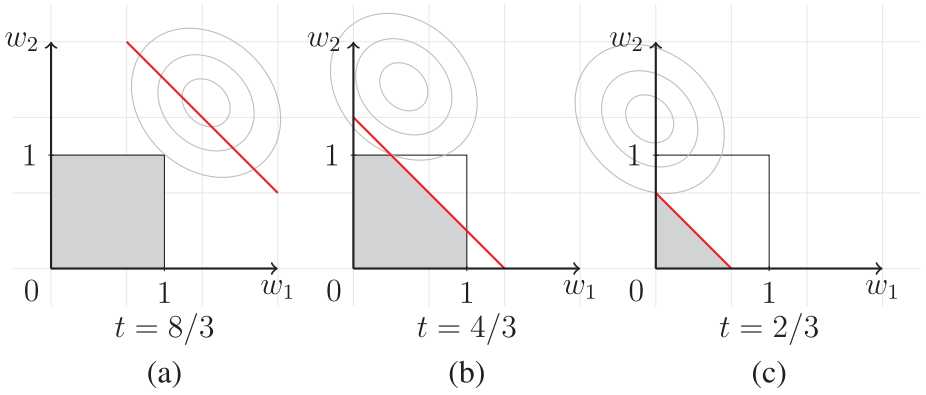


Figure 1. Geometric interpretation of the regularization term $\mathbf{1}^\top\mathbf{w}$. The shaded areas are the constraint regions $w_1, w_2 \in [0,1]$ and $w_1 + w_2 < t$ for different t's. The ellipses are the contours of the loss function as a function of w_1 and w_2. Sparsity is induced in (c).

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

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

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 jth variable is zero, the contribution of the jth 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

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

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 $t \le 1$, the constraint region becomes a right triangle that is similar to a constraint region induced by an ℓ_1 norm.

3. Computation

In this section, we introduce the algorithms for optimizing the Ktweedie and SKtweedie described in Sections 2.1 and 2.2, respectively. For the Ktweedie model with 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 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). The BFGS enjoys the fast convergence rate of the Newton-type algorithms and avoids the exact computation and inverse of the Hessian matrix whose dimension is equal to the sample size. Solving the Ktweedie model requires additional considerations for the intercept α_0, which is discussed in Section B. 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 kth 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:

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

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

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 a bisection line-search in Algorithm S1 (Section A.1) to satisfy the *Wolfe conditions* (Wolfe 1971),

$$\begin{cases} g(\boldsymbol{\alpha}+t\mathbf{p}) \le g(\boldsymbol{\alpha}) + c_1 t \nabla g(\boldsymbol{\alpha})^\top \mathbf{p} & \text{Condition 1} \\ \nabla g(\boldsymbol{\alpha}+t\mathbf{p})^\top \mathbf{p} \ge c_2 \nabla g(\boldsymbol{\alpha})^\top \mathbf{p} & \text{Condition 2} \end{cases}, \quad (17)$$

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.

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. S1 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*;
$\widehat{\boldsymbol{\alpha}} = \boldsymbol{\alpha}^{(k)}$

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 Section C. □

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 S2 (Section A.2) when the bisection line-search in the BFGS fails.

3.2. Fitting the SKtweedie model

We propose an alternating optimization method (Algorithm 2) 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 S3; Section A.3). 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 constraint. Denote the kth update of w_j within the mth outer loop iteration by $w_j^{(m,k)}$. Then the inner $\mathbf{w}$-loop update has the form

$$w_j^{(m,k+1)} = \text{proj}\left(w_j^{(m,k)} - t^{(m,k)} \cdot \nabla_{w_j} g(\boldsymbol{\alpha}^{(m)}, \mathbf{w}^{(m,k)})\right), \quad (18)$$

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 Section D. The details of the alternating optimization are given in Algorithm 2.

Algorithm 2: 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. S3 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*;
$\widehat{\boldsymbol{\alpha}} = \boldsymbol{\alpha}^{(m)}$
$\widehat{\mathbf{w}} = \mathbf{w}^{(m)}$

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 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

$$\widehat{\rho} = \underset{\rho \in \{\rho_1, \rho_2, \ldots, \rho_l\}}{\arg\max} \ \ell(\widehat{\mu}(\rho), \widehat{\phi}(\rho), \rho). \tag{19}$$

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, for example, $\rho = 1.5$ 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 Section E). 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, that is, $w_j^{(m,k)} = 0$ 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 mth 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 RBF kernel (Section D), 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 (18). 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.

We provide an R package `ktweedie` (*https://cran.r-project.org/package=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, Qian, and Zou 2018) in the `R` package `TDboost` (Yang, Qian, and Zou 2016; hereinafter TDboost). The model tuning for the TDboost is replicated from Yang, Qian, and Zou (2018). We consider the RBF kernel and the Laplace kernel in the Ktweedie for demonstration purposes. Throughout this section, we denote the true $f(\cdot)$ used in the simulations by $F(\cdot)$. Results related to the computation times are generated with R version 4.1.2 "Bird Hippie" on a 2021 MacBook Pro with the 10-core M1 Pro CPU and 16GB unified memory that runs macOS Monterey Version 12.4.

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: $\log(\mu) = F(\mathbf{x}) = 0.5\text{I}(x > 0.5)$
Here the true $\log(\mu)$ is a non-smooth function of the one-dimensional predictor x. We assume that $x \sim \text{Unif}(0, 1)$, and $y \sim \text{Tw}(\mu, \phi, \rho)$ 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 $x_1, x_2 \sim \text{Unif}(0, 1)$, and $y \sim \text{Tw}(\mu, \phi, \rho)$ 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 5-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

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

We choose to use the MAD in the article because it is a commonly used measure of prediction error (e.g., Friedman 2001 and Hastie et al. 2009). There is no specific reason why other proper norms such as the RMSE cannot also be used. Since the true F is known in simulation, both the MAD and RMSE are sensible measures that can quantify the difference between the predicted $\widehat{F}$ and the true F and are expected to deliver similar results.

The resulting MADs based on 100 replications are reported in Tables 2 and 3, and some sample predictions are plotted in Figures S1 and 2, for Models 1 and 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 and the Laplace kernels outperforms MGCV, TDboost and TGLM in the smooth function case. We also report the total computation times required for cross-validation, model fitting and prediction in Tables S1 and S2.

In addition, we estimate ϕ and ρ using the profile likelihood approach proposed in Section 3.3. The setups of Model 1 and

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 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)

Table 3. 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)

Model 2 are adopted with true $\rho = 1.5$ and true $\phi = 0.5$ and we only consider the RBF kernel. 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 (19) is used in the subsequent estimation of ϕ and μ. In Figure S2, the estimated likelihoods at different candidate ρ's from a sample run for each model is plotted. On the left, $\widehat{\rho} = 1.52$ for Model 1, and on the right, $\widehat{\rho} = 1.58$ for Model 2. The estimates of ρ and ϕ and the MADs from 20 independent replications are summarized in Table S3. Note that, the MADs are slightly higher than the corresponding entries in Tables 2 and 3 as a price for not knowing the true ϕ and ρ. Overall, the profile likelihood approach coupled with the Ktweedie 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 $\{g_k\}_{k=1}^{20}$:

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

where the b_k's are the coefficients generated from 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,

$$\mathbf{z}_k = \{\mathbf{x}_{\psi_k(j)}\}_{j=1}^{p_k} \tag{21}$$

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 (21) indicates the jth 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, $g_k(\mathbf{z}_k)$ is a p_k-dimensional Gaussian function:

$$g_k(\mathbf{z}_k) = \exp\left\{-\frac{1}{2}(\mathbf{z}_k - \mathbf{u}_k)^\top \mathbf{V}_k (\mathbf{z}_k - \mathbf{u}_k)\right\}, \tag{22}$$

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 $\sqrt{d_k[j]} \overset{\text{iid}}{\sim} \mathrm{Unif}(0.1, 2.0)$. 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 $\mu_i = \exp[F(\mathbf{x}_i)]$.

Table 4. 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)

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 4 and plotted in Figure S3. The results suggest that in the presence of complicated interactions among the predictors, the Ktweedie outperforms both the MGCV and TDboost. We also report the computational times required for the cross-validation, model fitting and prediction in Table S4.

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 S4, the estimation accuracy of μ is almost unaffected by ρ.

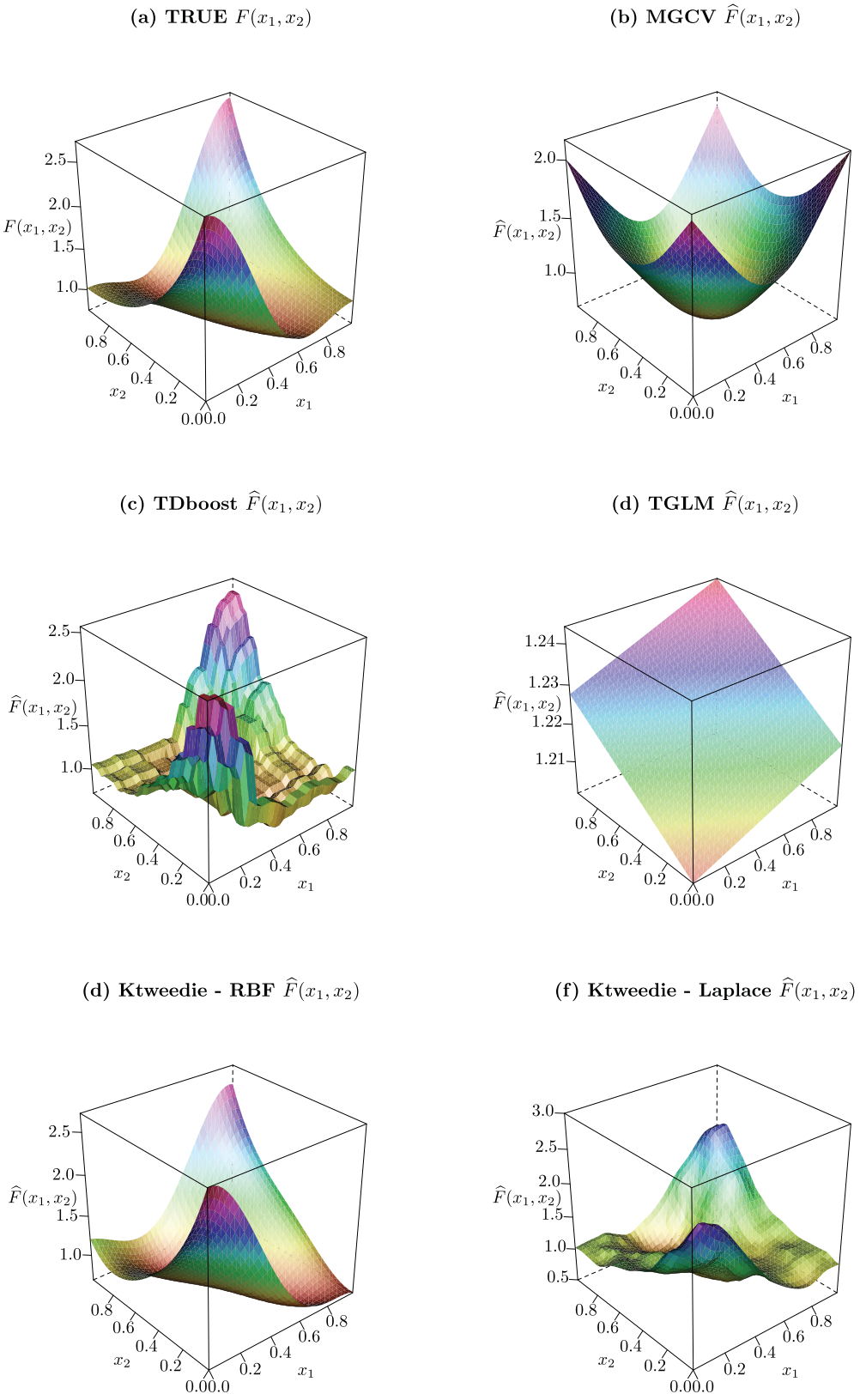


Figure 2. Fitted $\widehat{F}(\mathbf{x})$ versus true $F(\mathbf{x})$ in Model 2 from a sample run ($\phi = 0.5$).

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 (20) 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

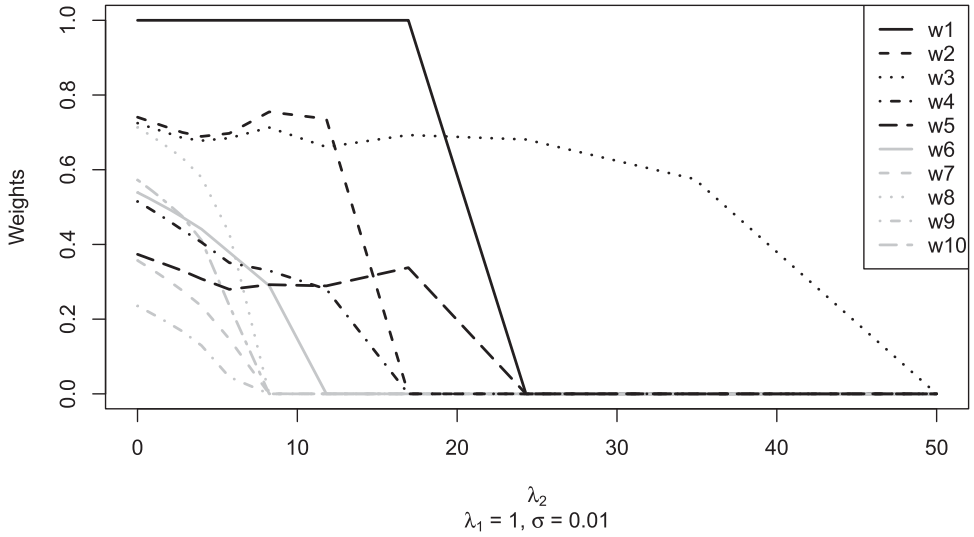


Figure 3. A sample SKtweedie solution path in case IV with arbitrary $\lambda_1 = 1$ and $\sigma = 0.01$.

Table 5. 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)

(22) 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 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 5 and Figure S5. 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.

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 nonzero 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 3 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 S6 shows the estimated weights in the 20 replications for $p = 10$ and 50 ($p = 200$ in Figure S7) 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 6. Overall SKtweedie is able to achieve good variable selection accuracy. In addition, we use this case to demonstrate the effect of sample size on the computation times. Specifically, for each sample size $n = \{50, 100, 200, 400\}$, we record the time needed to fit a Ktweedie model and an SKtweedie model at fixed σ, λ_1 and λ_2. The dimension is fixed at $p = 10$. The timing results averaged over 20 replications are plotted in Figure S8.

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 insurer in United States. The data is obtained from the Schedule P of the National Association of Insurance

Table 6. 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.

	$p = 10$		$p = 50$	
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 7. A typical run-off triangle of incremental paid losses (shaded) and the additional accident years with fully developed claims.

			Development year				
	Accident year		1	2	⋯	9	10
Fully developed	1989	1	$Y_{1,1}$	$Y_{1,2}$	⋯	$Y_{1,9}$	$Y_{1,10}$
	⋮	⋮	⋮	⋮	⋱	⋮	⋮
	1996	8	$Y_{8,1}$	$Y_{8,2}$	⋯	$Y_{8,9}$	$Y_{8,10}$
Run-off triangle	1997	9	$Y_{9,1}$	$Y_{9,2}$	⋯	$Y_{9,9}$	$Y_{9,10}$
	1998	10	$Y_{10,1}$	$Y_{10,2}$	⋯	$Y_{10,9}$	
	⋮	⋮	⋮	⋮	⋰		
	2005	17	$Y_{17,1}$	$Y_{17,2}$			
	2006	18	$Y_{18,1}$				

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 ith AY during the jth 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 7 exhibits the data that are 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 7. Assume Y_t satisfies the Markov property such that the joint distribution of $Y_2, \ldots, Y_T$ can be factorized as

$$g(Y_2, ..., Y_T) = g(Y_2|H_2) \prod_{t=3}^{T} g(Y_t|Y_{t-1}, H_t), \tag{23}$$

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,

$$\begin{aligned} g(Y_t|Y_{t-1}, H_t) &= \prod_{i+j=t} g(Y_{i,j}|Y_{t-1}, H_t) \\ &= \prod_{i+j=t} g(Y_{i,j}|Y_{i-1,j}, Y_{i,j-1}, i, j). \end{aligned} \tag{24}$$

Table 8. 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.

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

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

$$Y_{i,j}|(Y_{i-1,j}, Y_{i,j-1}, i, j) \sim \text{Tw}(\mu_{i,j}, \phi, \rho). \tag{25}$$

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 (Wüthrich 2003; Peters, Shevchenko, and Wüthrich 2008; Taylor and McGuire 2016).

We consider several versions of Tweedie model assumptions in (25)—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 $f \in \mathcal{H}_K$. Due to the limited number of available features that renders variable selection unnecessary, we only consider the prediction problem using Ktweedie. 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 $Y_{i,j}$ 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, that is, 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 8 reports the root mean squared errors (RMSE) for the test data. The Ktweedie model with the RBF kernel outperforms all the other methods and is the only Tweedie model that beats

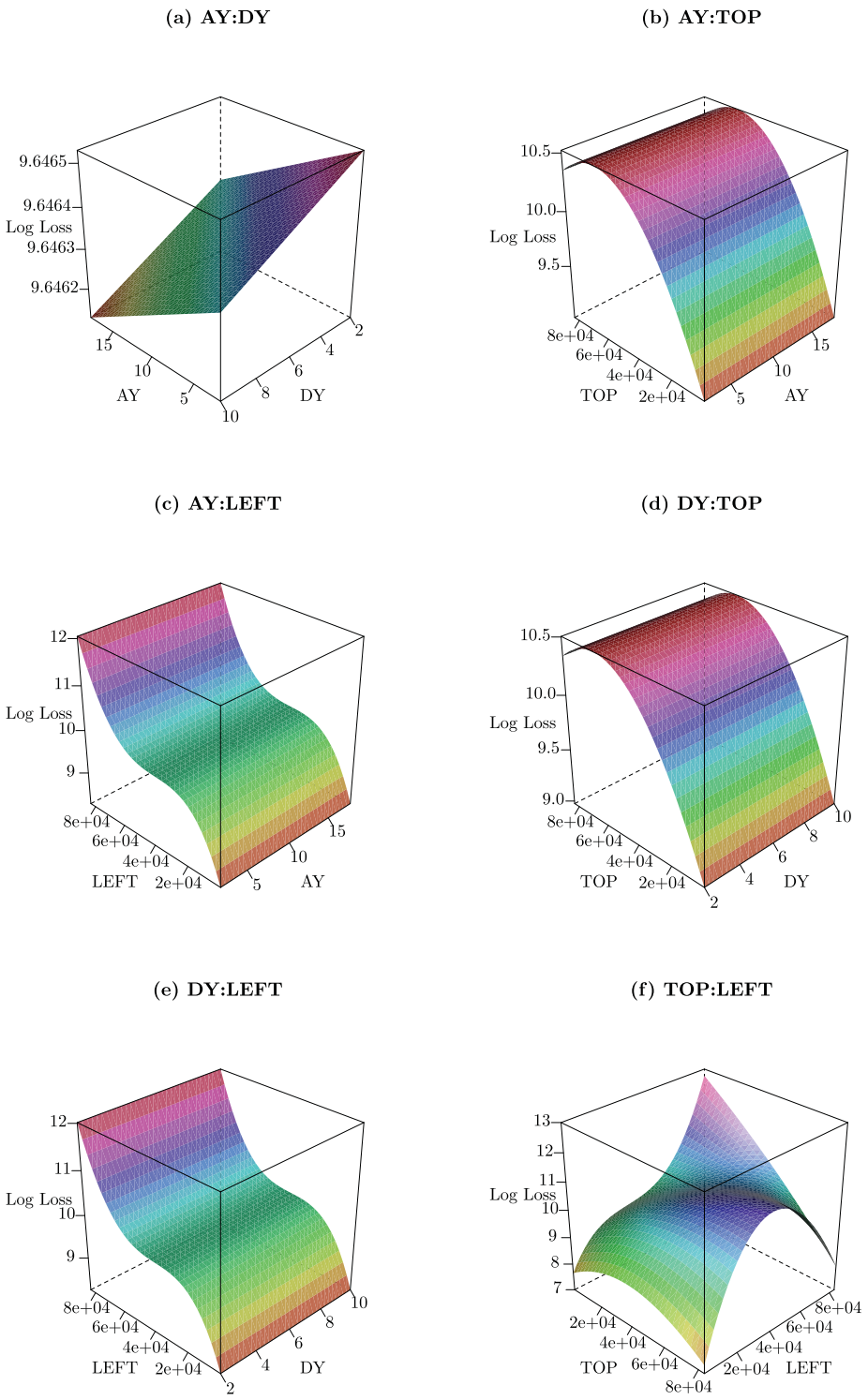


Figure 4. Visualization of pairwise interactions using Ktweedie. With the effects of accident year and development year fixed, the predictions made with the Ktweedie model reveals complex nonlinear interaction between the variables TOP and LEFT.

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 8). 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 S9 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 4(a)–(f), among which, panel (f) emphasizes

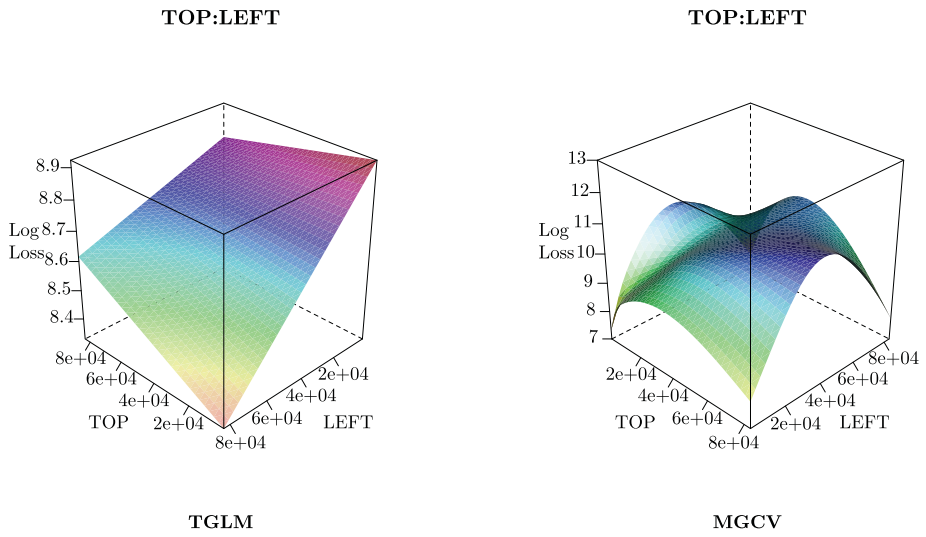


Figure 5. Visualization of the TOP-LEFT interaction using predictions made by the TGLM (left) and MGCV (right) models.

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 5(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 4(f)). The results in Table 8 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.

5.2. Case Study II: Ratemaking

Dataset. For the purpose of pricing insurance contract, we 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 6290 (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 9 and use them as predictors.

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 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 noninformative baseline. In MGCV, 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 5-fold cross-validation as described in Yang, Qian, and Zou (2018). For Ktweedie, the training involves tuning of the kernel parameter σ and the regularization coefficient λ with a 5-fold cross-validation from 10 pairs of candidate values. For all of the methods, we record the total computation times needed for model training, fitting and prediction.

Table 9. Explanatory variables in the claim history dataset.

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

	MAD	RMSE	nGini	Timing
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)	0.026 (0.001)
MGCV	4.258 (0.069)	7.742 (0.179)	0.595 (0.012)	8.370 (0.540)
TDboost	4.119 (0.066)	7.571 (0.180)	0.600 (0.010)	69.75 (0.088)
Ktweedie	4.225 (0.064)	7.598 (0.177)	0.597 (0.011)	441.4 (15.831)

Table 11. The averaged Gini indices (Frees, Meyers, and Cummings 2011) and standard deviations in the auto-insurance claim data example based on 20 replications.

	Competing premium				
Base premium	MEAN	TGLM	MGCV	TDboost	Ktweedie
MEAN	0	48.154(0.926)	48.297(0.976)	48.646(0.819)	48.458(0.876)
TGLM	8.140(2.453)	0	8.115(1.704)	13.527(2.002)	10.476(1.362)
MGCV	6.515(2.303)	2.751(3.573)	0	10.367(2.250)	7.190(2.422)
TDboost	−0.983(1.773)	3.741(2.250)	3.089(1.907)	0	3.911(1.696)
Ktweedie	1.458(1.449)	1.450(2.043)	2.741(2.069)	8.144(1.513)	0

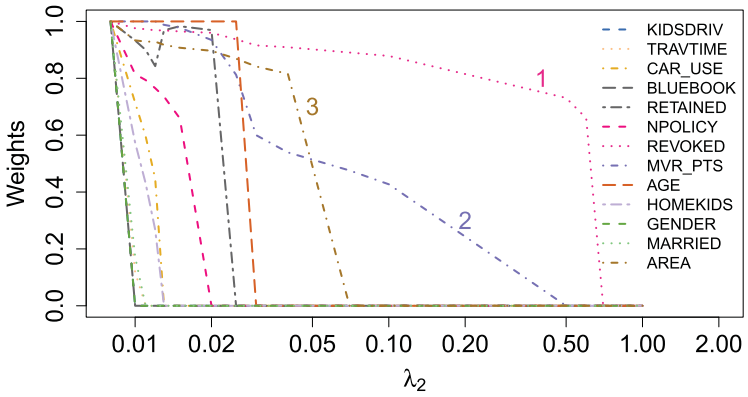


Figure 6. A sample solution path obtained by the SKtweedie model showing the change in variable weights with increasing sparsity-inducing regularization coefficient. The most important predictors of the claim loss are (1) the history of license revocation, (2) the motor vehicle record points, and (3) the area of the policyholder's home/work.

Performance comparison. To compare predictive performance, we first examine the MAD, RMSE, and normalized Gini index (nGini; Ye et al. 2018). Smaller MAD and RMSE, and larger nGini is preferred. Table 10 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 all three criteria slightly supporting the TDboost. The computation times are reported in the rightmost column of the table.

Due to the large proportion of zero outcomes and high skewness, we also use the ordered Lorenz curve and the associated Gini index as performance measures (Frees, Meyers, and Cummings 2011). We consider a pairwise comparison among alternative models and Table 11 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 not take into account the rating variables. Second, both 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 S10 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 11 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 can be used to identify important predictors. We first select parameters λ_1 and σ using cross-validation, and then construct a solution path for the SKtweedie with respect to λ_2. As shown in Figure 6, 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 driver'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, Qian, and Zou (2018).

6. Conclusion

In this article 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, for example, 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, Camoriano, and Rosasco 2015) or random features (Rahimi and Recht 2007). These approximations can also improve prediction performances as they induce implicit regularization.

Supplementary Materials

Supplementary Material includes derivations, proofs, Algorithms S1–S3, Tables S1–S4, and Figures S1–S10 in Sections A–F, as well as R code to reproduce Simulation Case I and links to the development and CRAN versions of the R package `ktweedie`.

Acknowledgments

We sincerely thank the Editor, the Associate Editor, and two anonymous Reviewers for their valuable comments toward improving this work.

Disclosure Statement

The authors report there are no competing interests to declare.

Funding

Yi Lian acknowledges the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) PGSD2-519554-2018. Archer Yi Yang acknowledges the support of the NSERC Discovery grant RGPIN-2016-05174.

ORCID

Yi Lian http://orcid.org/0000-0001-7832-5217

References

Allen, G. I. (2013), "Automatic Feature Selection via Weighted Kernels and Regularization," *Journal of Computational and Graphical Statistics*, 22, 284–299. [282,283,284]

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*, pp. 41–48. [283]

Armijo, L. (1966), "Minimization of Functions Having Lipschitz Continuous First Partial Derivatives," *Pacific Journal of Mathematics*, 16, 1–3. [285]

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. [281]

Brent, R. P. (2013) *Algorithms for Minimization Without Derivatives*, Chelmsford, MA: Courier Corporation. [285]

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. [284]

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*, pp. 121–128. [283]

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, 401–420. [282,283]

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. [281]

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. [281]

Dunn, P. K., and Smyth, G. K. (2005), "Series Evaluation of Tweedie Exponential Dispersion Model Densities," *Statistics and Computing*, 15, 267–280. [285]

Duvenaud, D. (2014), "The Kernel Cookbook: Advice on Covariance Functions," Available at *https://www.cs.toronto.edu/%7Eduvenaud/cookbook/*. [282]

Dzupire, N. C., Ngare, P., and Odongo, L. (2018), "A Poisson-Gamma Model for Zero Inflated Rainfall Data," *Journal of Probability and Statistics*, 2018, 1–12. [281]

El-Shaarawi, A. H., Zhu, R., and Joe, H. (2011), "Modelling Species Abundance using the Poisson–Tweedie Family," *Environmetrics*, 22, 152–164. [281]

Fan, J., and Fan, Y. (2008), "High Dimensional Classification using Features Annealed Independence Rules," *Annals of statistics*, 36, 2605–2637. [282]

Fletcher, R. (1970),"A New Approach to Variable Metric Algorithms," *The Computer Journal*, 13, 317–322. [284]

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. [281]

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. [281]

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. [293]

Friedman, J. H. (2001), "Greedy Function Approximation: A Gradient Boosting Machine," *Annals of Statistics*, 29, 1189–1232. [281,286,287]

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. [283]

Giner, G., and Smyth, G. K. (2016), "statmod: Probability Calculations for the Inverse Gaussian Distribution," *R Journal*, 8, 339–351. [286]

Goldfarb, D. (1970), "A Family of Variable-Metric Methods Derived by Variational Means," *Mathematics of Computation*, 24, 23–26. [284]

Grandvalet, Y., and Canu, S. (2002), "Adaptive Scaling for Feature Selection in svms," in *Advances in Neural Information Processing Systems* (Vol. 15). [283]

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. [282]

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. [281]

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. (2009), *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (Vol. 2), New York: Springer. [286]

Hastie, T. J., and Tibshirani, R. J. (1990), *Generalized Additive Models* (Vol. 43), Boca Raton, FL: CRC Press. [281]

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. [281]

Jørgensen, B. (1987), "Exponential Dispersion Models," *Journal of the Royal Statistical Society*, Series B, 49, 127–162. [282,283]

——— (1997), *The Theory of Dispersion Models*, Boca Raton, FL: CRC Press. [281]

Jørgensen, B., and de Souza, M. C. (1994), "Fitting Tweedie's Compound Poisson Model to Insurance Claims Data," *Scandinavian Actuarial Journal*, 1994, 69–93. [281,282,283,286]

Kurz, C. F. (2017), "Tweedie Distributions for Fitting Semicontinuous Health Care Utilization Cost Data," *BMC Medical Research Methodology*, 17, 1–8. [281]

Lee, S. C., and Lin, S. (2018), "Delta Boosting Machine with Application to General Insurance," *North American Actuarial Journal*, 22, 405–425. [281]

Li, F., Yang, Y., and Xing, E. (2005), "From Lasso Regression to Feature Vector Machine," *Advances in Neural Information Processing Systems*, 18, 779–786. [283]

Lin, Y., and Zhang, H. H. (2006), "Component Selection and Smoothing in Multivariate Nonparametric Regression," *The Annals of Statistics*, 34, 2272–2297. [283]

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. [290]

Meyers, G. G., and Shi, P. (2011), "Loss Reserving Data Pulled from NAIC Schedule P." Available at *https://www.casact.org/publications-research/research/research-resources/loss-reserving-data-pulled-naic-schedule-p*. [290]

Moshitch, D., and Nelken, I. (2014), "Using Tweedie Distributions for Fitting Spike Count Data," *Journal of Neuroscience Methods*, 225, 13–28. [281]

NAIC. (2021), "Data Products – Schedule P," Available at *https://content.naic.org/prod_serv_idp_sched_p.htm*. [290]

Nocedal, J., and Wright, S. (2006), *Numerical Optimization*, New York: Springer. [284,285]

Ohlsson, E., and Johansson, B. (2010), *Non-Life Insurance Pricing with Generalized Linear Models* (Vol. 2), Berlin: Springer. [282]

Peters, G. W., Shevchenko, P. V., and Wüthrich, M. V. (2008), "Model Risk in Claims Reserving within Tweedie's Compound Poisson Models," *ASTIN Bulletin* (to appear). [290]

——— (2009), "Model Uncertainty in Claims Reserving within Tweedie's Compound Poisson Models," *ASTIN Bulletin: The Journal of the IAA*, 39, 1–33. [282]

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. [281,282]

Rahimi, A., and Recht. (2007), "Random Features for Large-Scale Kernel Machines," in *Advances in Neural Information Processing Systems* (Vol. 3), p. 5. [294]

Rasmussen, C. E. (2003), "Gaussian Processes in Machine Learning," in *Summer School on Machine Learning*, eds. O. Bousquet, U. Luxburg, and G. Rätsch, pp. 63–71, Berlin: Springer. [282]

Rudi, A., Camoriano, R., and Rosasco, L. (2015), "Less is More: Nyström Computational Regularization," in *Advances in Neural Information Processing Systems*, pp. 1657–1665. [294]

Shanno, D. F. (1970), "Conditioning of Quasi-Newton Methods for Function Minimization," *Mathematics of Computation*, 24, 647–656. [284]

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. [282]

——— (2016), "Insurance Ratemaking Using a Copula-based Multivariate Tweedie Model," *Scandinavian Actuarial Journal*, 2016, 198–215. [282]

Shi, P., Feng, X., and Boucher, J.-P. (2016), "Multilevel Modeling of Insurance Claims Using Copulas," *The Annals of Applied Statistics*, 10, 834–863. [281,282]

Shono, H. (2008), "Application of the Tweedie Distribution to Zero-Catch Data in CPUE Analysis," *Fisheries Research*, 93, 154–162. [281]

Smyth, G., Hu, Y., Dunn, P., Phipson, B. and shun Chen, Y. (2021), *Statistical Modeling*. R package version 1.4.36. Available at *https://cran.r-project.org/package=statmod*. [286]

Smyth, G., and Jorgensen, B. (2002), "Fitting Tweedie's Compound Poisson Model to Insurance Claims Data: Dispersion Modelling," *ASTIN Bulletin*, 32, 143–157. [281,282]

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*, pp. 572–580. [282]

Sriram, K., and Shi, P. (2020), "Stochastic Loss Reserving: A New Perspective from a Dirichlet Model," *Journal of Risk and Insurance*, 88, 195–230. [290]

Taylor, G. (2019), "Loss Reserving Models: Granular and Machine Learning Forms," *Risks*, 7, 82. [282]

Taylor, G., and McGuire, G. (2016), "Stochastic Loss Reserving Using Generalized Linear Models," *CAS Monograph*, 3, 1–112. [282,290]

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*, pp. 579–604. [281,282,283]

Vapnik, V. (2013), *The Nature of Statistical Learning Theory*, New York: Springer. [282]

Wahba, G. (1990), *Spline Models for Observational Data* (Vol. 59), Philadelphia, PA: SIAM. [283]

Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T., and Vapnik, V. (2000), "Feature Selection for SVMs," in *Advances in Neural Information Processing Systems* (Vol. 13), eds. T. Leen, T. Dietterich and V. Tresp, MIT Press. [283]

Wolfe, P. (1971), "Convergence Conditions for Ascent Methods. II: Some Corrections," *SIAM Review*, 13, 185–188. [285]

Wood, S. (2021), *Mixed GAM Computation Vehicle with Automatic Smoothness Estimation*. R package version 1.8-36. *https://cran.r-project.org/package=mgcv*. [286]

Wood, S. N. (2011), "Fast Stable Restricted Maximum Likelihood and Marginal Likelihood Estimation of Semiparametric Generalized Linear Models," *Journal of the Royal Statistical Society*, Series B, 73, 3–36. [281,286]

Wüthrich, M. V. (2003), "Claims Reserving using Tweedie's Compound Poisson Model," *ASTIN Bulletin: The Journal of the IAA*, 33, 331–346. [290]

Yang, L., Lv, S., and Wang, J. (2016), "Model-Free Variable Selection in Reproducing Kernel Hilbert Space," *The Journal of Machine Learning Research*, 17, 2885–2908. [282]

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)*, pp. 3377–3381. IEEE. [281]

Yang, Y., Qian, W., and Zou, H. (2016), *TDboost: A Boosted Tweedie Compound Poisson Model*. R package version 1.2. Available at *https://CRAN.R-project.org/package=TDboost*. [286]

——— (2018), "Insurance Premium Prediction via Gradient Tree-Boosted Tweedie Compound Poisson Models," *Journal of Business & Economic Statistics*, 36, 456–470. [281,283,286,292,293]

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. [293]

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. [292]

Zhang, Y. (2013), "Likelihood-based and Bayesian Methods for Tweedie Compound Poisson Linear Mixed Models," *Statistics and Computing*, 23, 743–757. [281]

Zhou, H., Qian, W., and Yang, Y. (2020), "Tweedie Gradient Boosting for Extremely Unbalanced Zero-Inflated Data," *Communications in Statistics-Simulation and Computation*, 59, 5507–5529. [281]

Supplementary Material
A Tweedie Compound Poisson Model in Reproducing Kernel Hilbert Space

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

A Algorithms

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 (17).

[*]Department of Epidemiology, Biostatistics and Occupational Health, McGill University
[†]Department of Mathematics and Statistics, McGill University (archer.yang@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 S1: 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

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

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

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 S2: 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*;

A.3 Gradient Descent Update for the Weights in the SKtweedie

Algorithm S3: 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. S2 to find step size $t^{(m,k)}$
  for $j = 1, \ldots, p$ **do**
    Compute $w_j^{(m,k+1)}$ using (18)
  end
  $k := k + 1$
  if $\mathbf{w}^{(m,k+1)} = \mathbf{0}_p$ **then** exit;
until *convergence*;
$\mathbf{w}^{(m+1)} = \mathbf{w}^{(m,k)}$

B 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,

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

C 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 $\boldsymbol{\alpha}$,

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

   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

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

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

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

We can see that

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

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

be lower-bounded on $\mathcal{L}_0$,

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

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

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

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

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

In the sublevel set $\mathcal{L}_0$, the values of third derivatives of g in (1) can be upper-bounded

$$\left| \frac{\partial^3 g(\widetilde{\boldsymbol{\alpha}})}{\partial\alpha_1 \partial\alpha_i \partial\alpha_j} \right| \leq D, \tag{2}$$

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. □

D The Derivative of the SKtweedie Objective Function

The objective function is

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

where

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

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

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

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 $\frac{\partial g(\boldsymbol{\alpha},\mathbf{w})}{\partial w_j} = \lambda_2$, for all $w_j = 0$.

E 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 $\hat{\rho}$.

F Additional Tables and Figures

Table S1: The mean computation times for Case I Model 1 based on 20 replications for different values of ϕ.

ϕ	MGCV	TDboost	TGLM	RBF	Laplace
0.1	0.020	0.971	0.001	0.638	1.273
0.5	0.055	0.994	0.001	0.672	1.340
1.0	0.019	0.970	0.001	0.687	1.457
2.0	0.022	0.982	0.001	0.682	1.611

Table S2: The mean computation times for Case I Model 2 based on 20 replications for different values of ϕ.

ϕ	MGCV	TDboost	TGLM	RBF	Laplace
0.1	0.130	4.211	0.002	0.915	2.095
0.5	0.037	4.369	0.001	0.958	3.444
1.0	0.064	4.230	0.001	0.976	4.469
2.0	0.130	4.132	0.001	1.027	5.822

Table S3: 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)

Table S4: The mean computation times for Case II based on 20 replications for different values of ϕ.

ϕ	MGCV	TDboost	RBF	Laplace
0.1	0.703	0.088	0.417	0.436
0.5	0.686	0.088	0.672	0.706
1.0	0.679	0.088	0.202	0.276
2.0	0.756	0.088	0.236	0.274

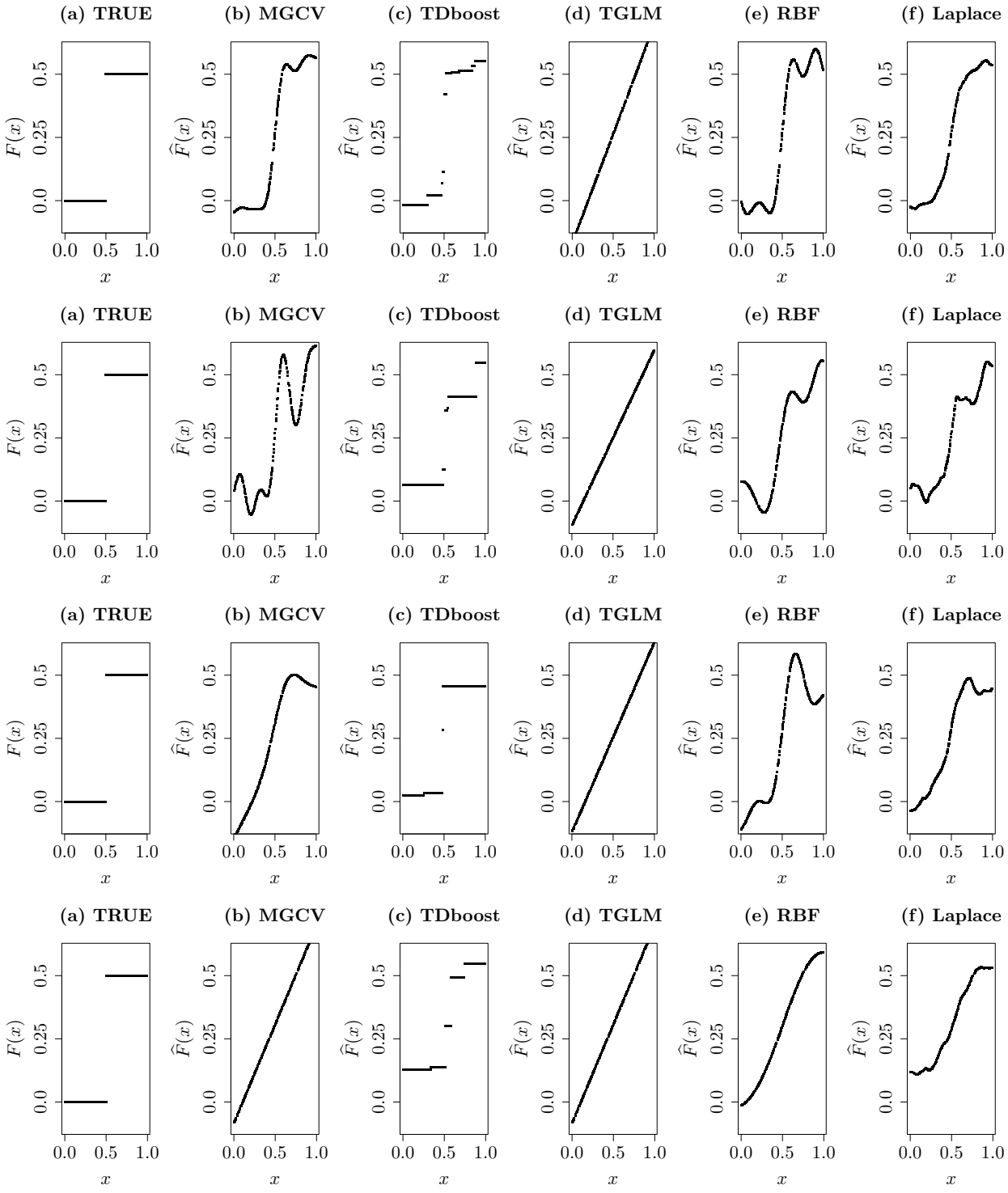
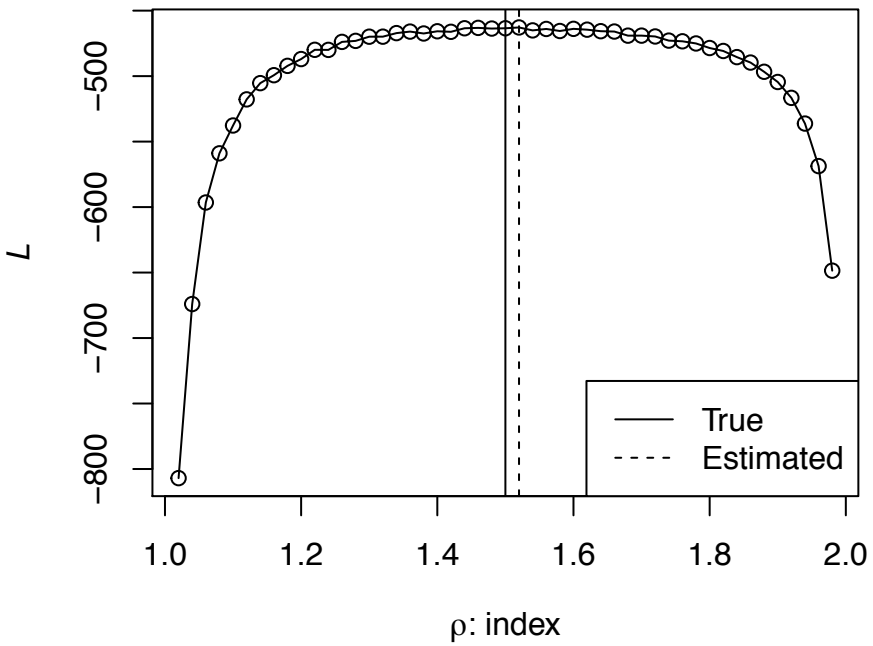


Figure S1: Fitted $\widehat{F}(\mathbf{x})$ vs. true $F(\mathbf{x})$ in Model 1 from a sample run (top to bottom $\phi = 0.1, 0.5, 1.0, 2.0$).



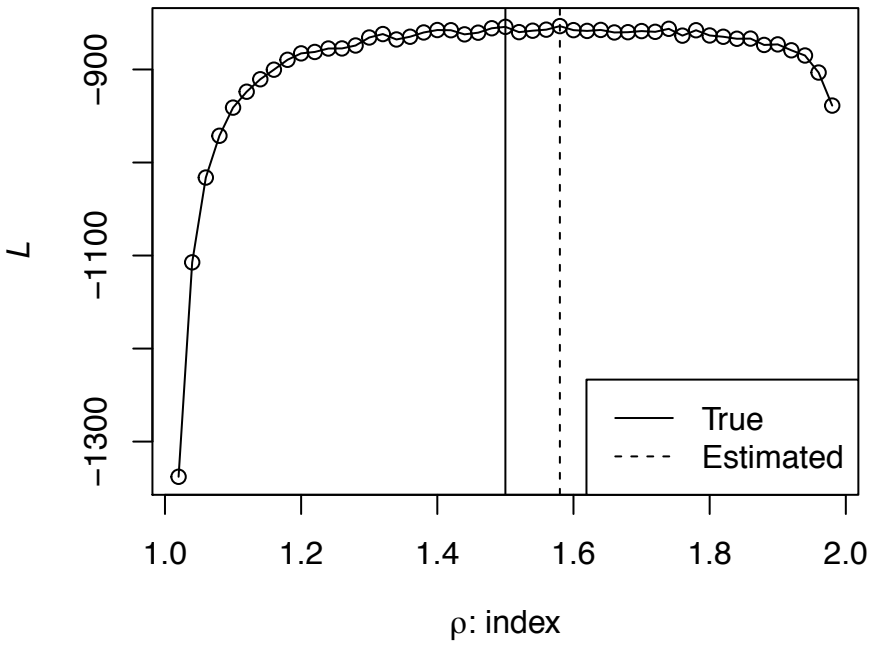


Figure S2: The profile likelihood of ρ from a sample run. Model 1 (left): true $\rho = 1.5$, $\widehat{\rho} = 1.52$; Model 2 (right): true $\rho = 1.5$, $\widehat{\rho} = 1.58$.

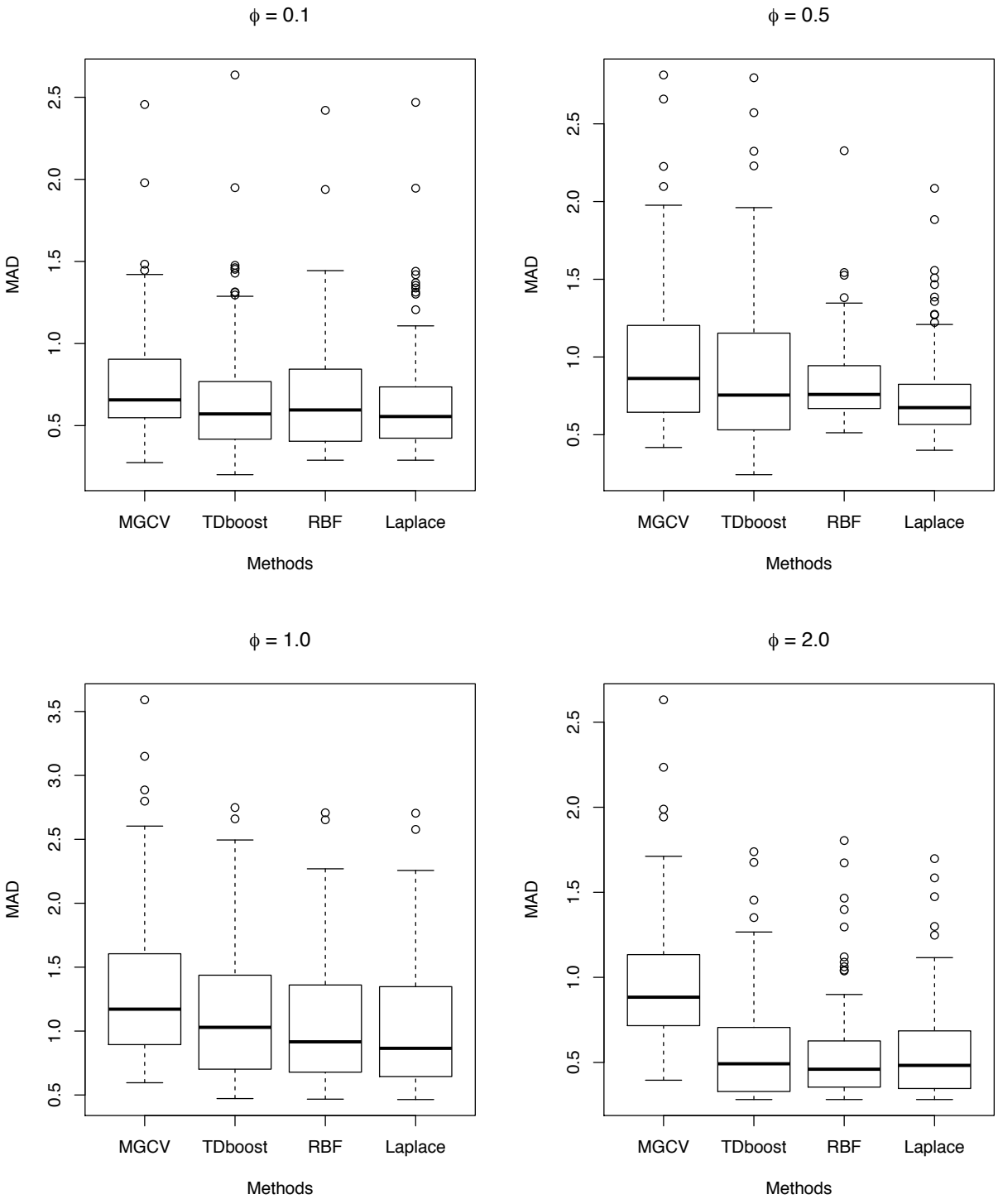


Figure S3: Distribution of the mean absolute deviations from the MGCV, TDboost, and Ktweedie (RBF and Laplace kernel) in Case II based on 100 independent replications.

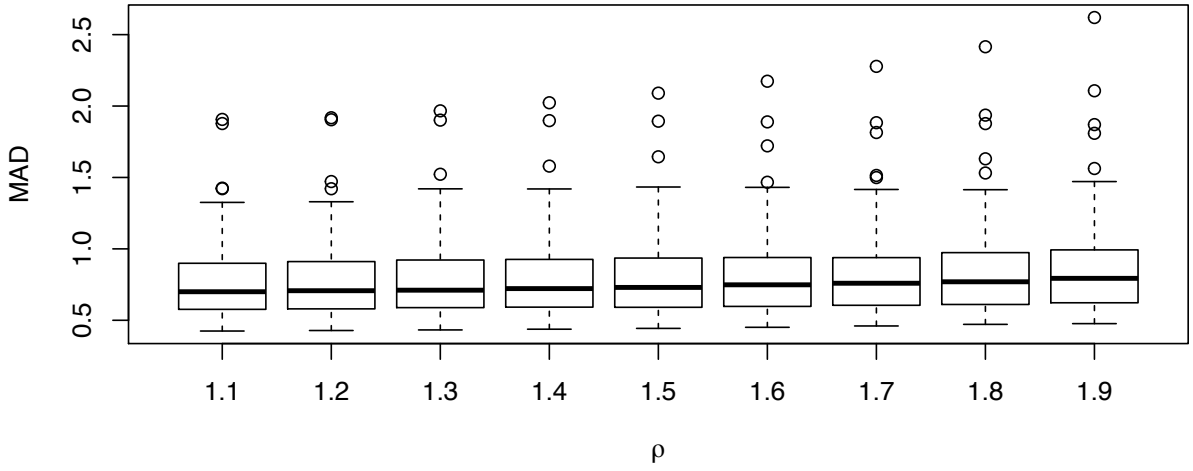


Figure S4: Boxplot of the mean absolute deviations for different values of the index parameter $\rho \in \{1.1, 1.2, \ldots, 1.9\}$ used during model fitting when the true value ($\rho = 1.5$) is unknown. The estimation accuracy is almost unaffected by ρ.

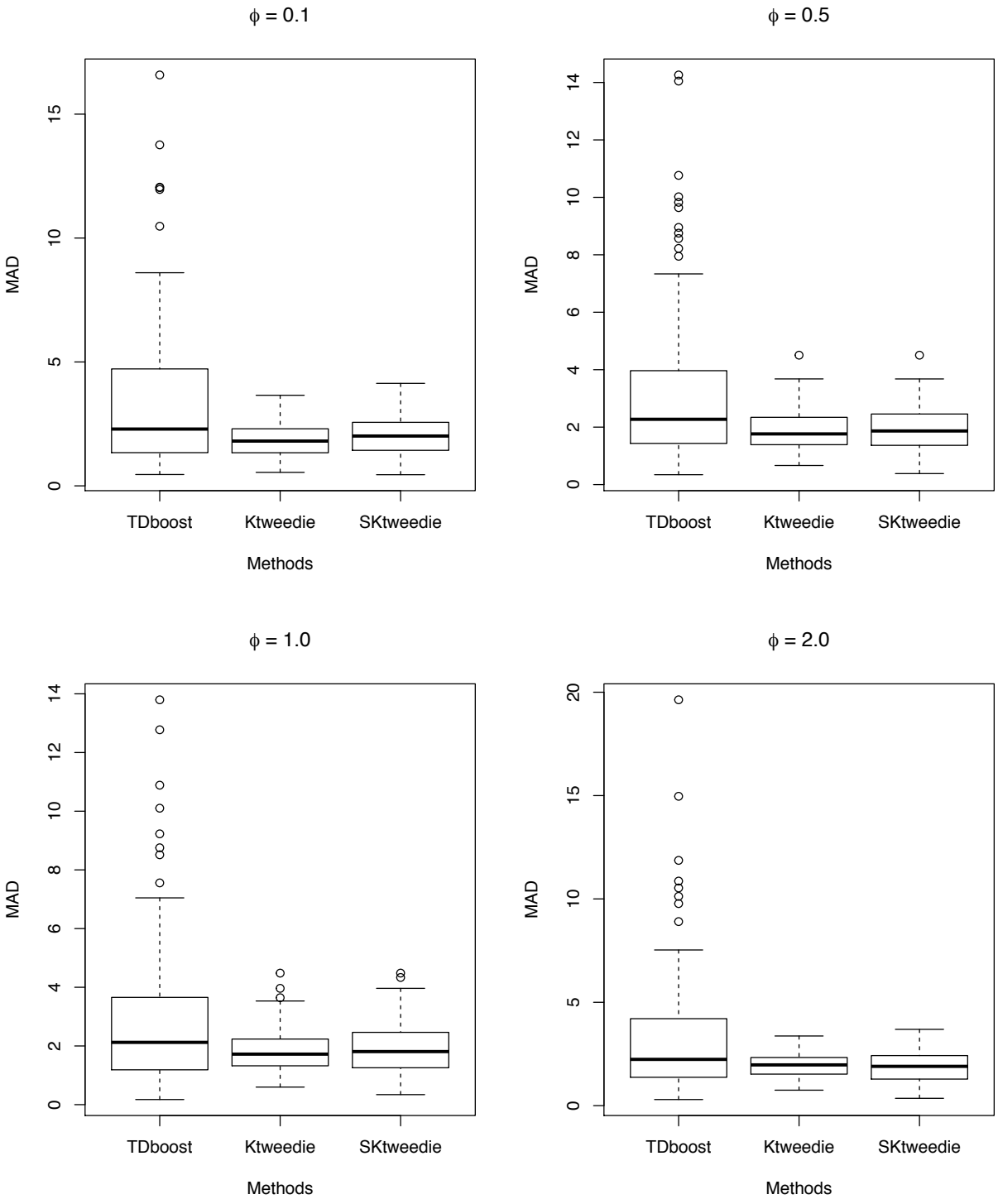


Figure S5: Distribution of the mean absolute deviations from the TDboost, Ktweedie, and SKtweedie in Case III based on 100 independent replications.

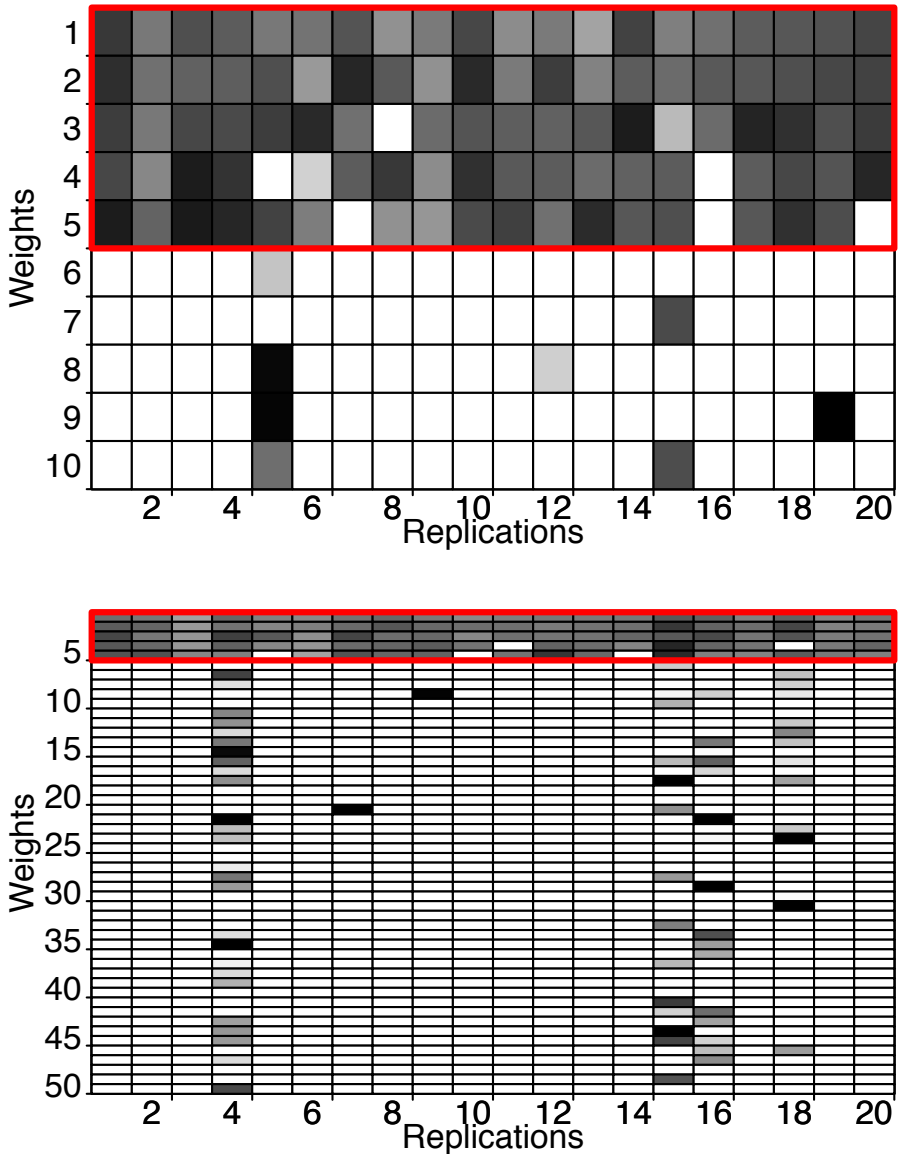


Figure S6: Variable selection results using SKtweedie with the Gaussian RBF kernel (left: $p = 10$, right: $p = 50$). Each column corresponds to a replication and each row corresponds to a variable, thus within the red rectangles are the true signal variables. The grayscale represents the magnitude of the estimated weights with a value between 0 and 1.

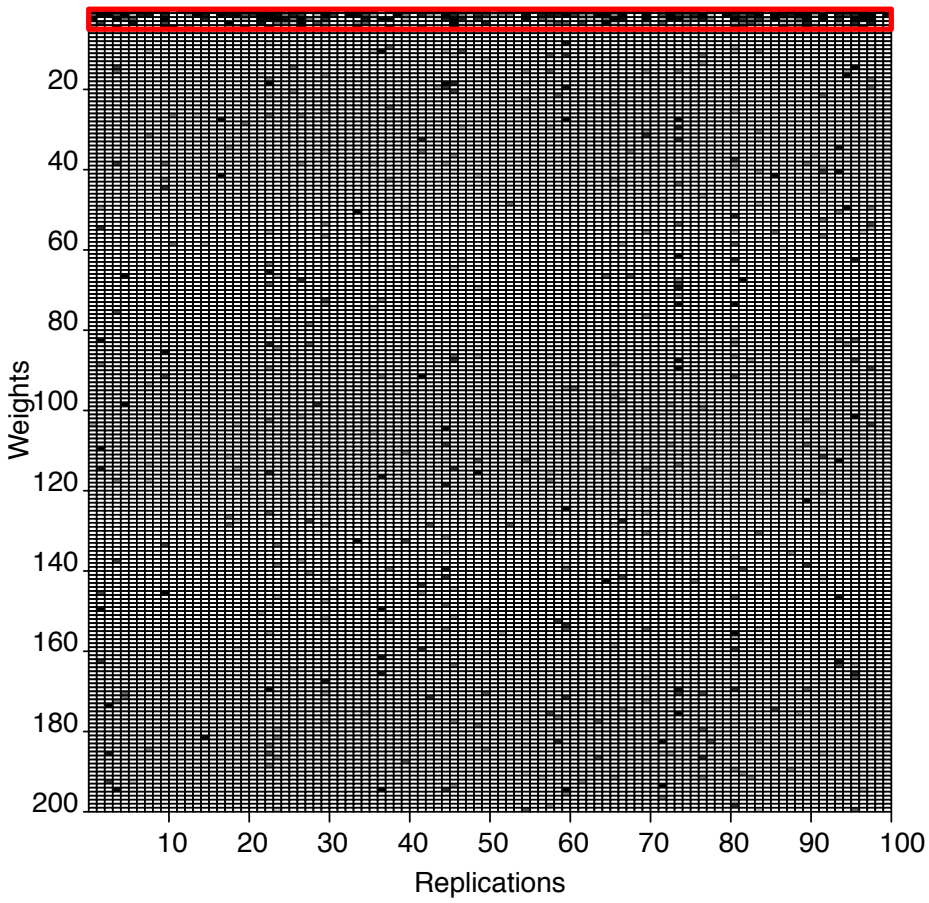


Figure S7: Variable selection results using the SKtweedie with Gaussian RBF kernel ($p = 200$)

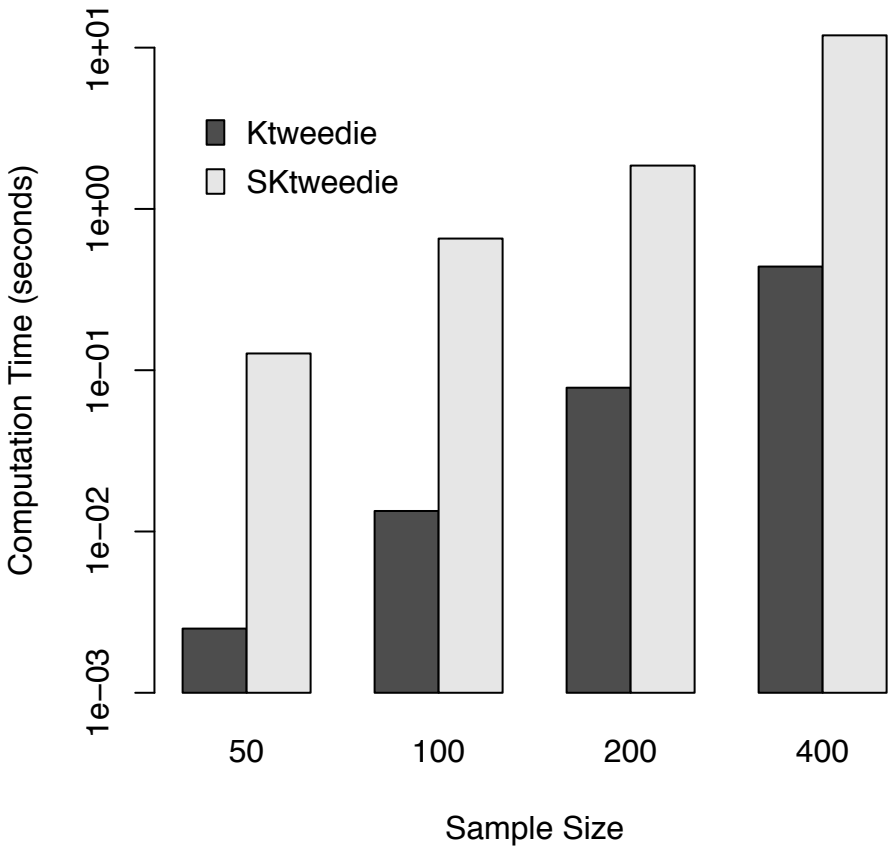


Figure S8: Computation times needed to fit a Ktweedie model and an SKtweedie model for sample size $n = 50, 100, 200, 400$ and $p = 10$ in simulation Case IV.

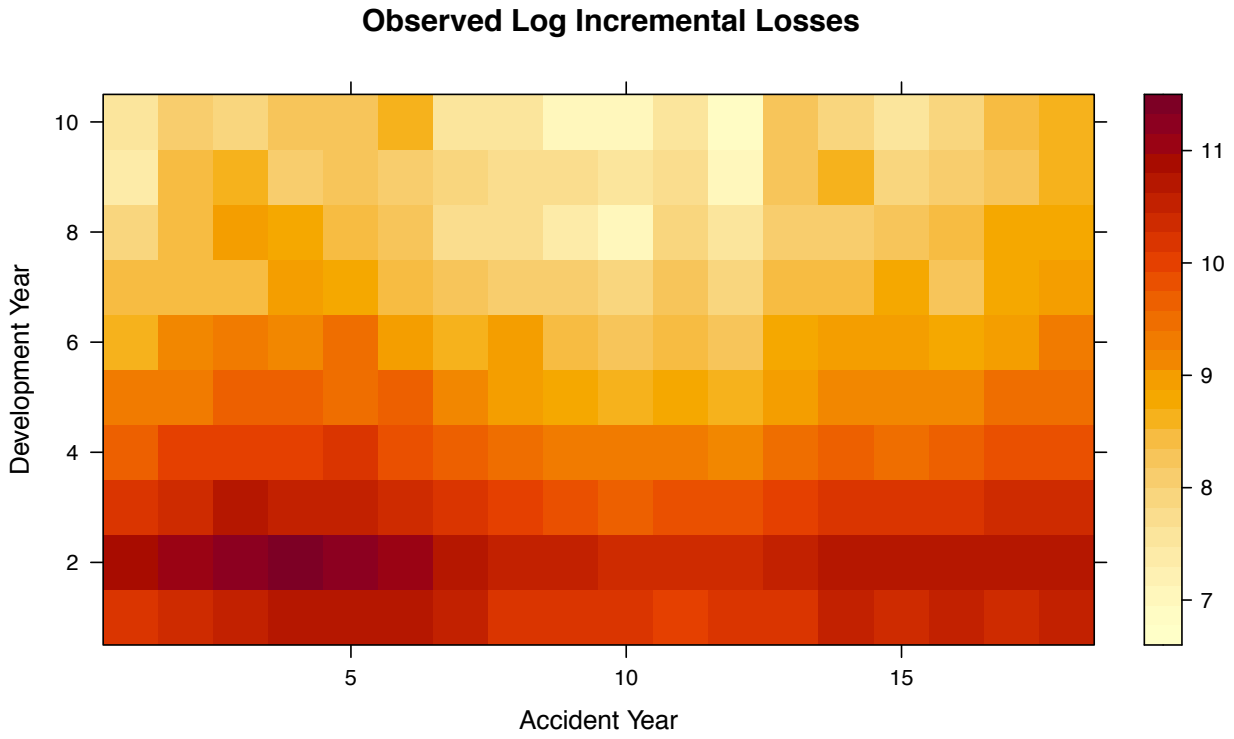


Figure S9: A heatmap of the log incremental losses by accident year and development year.

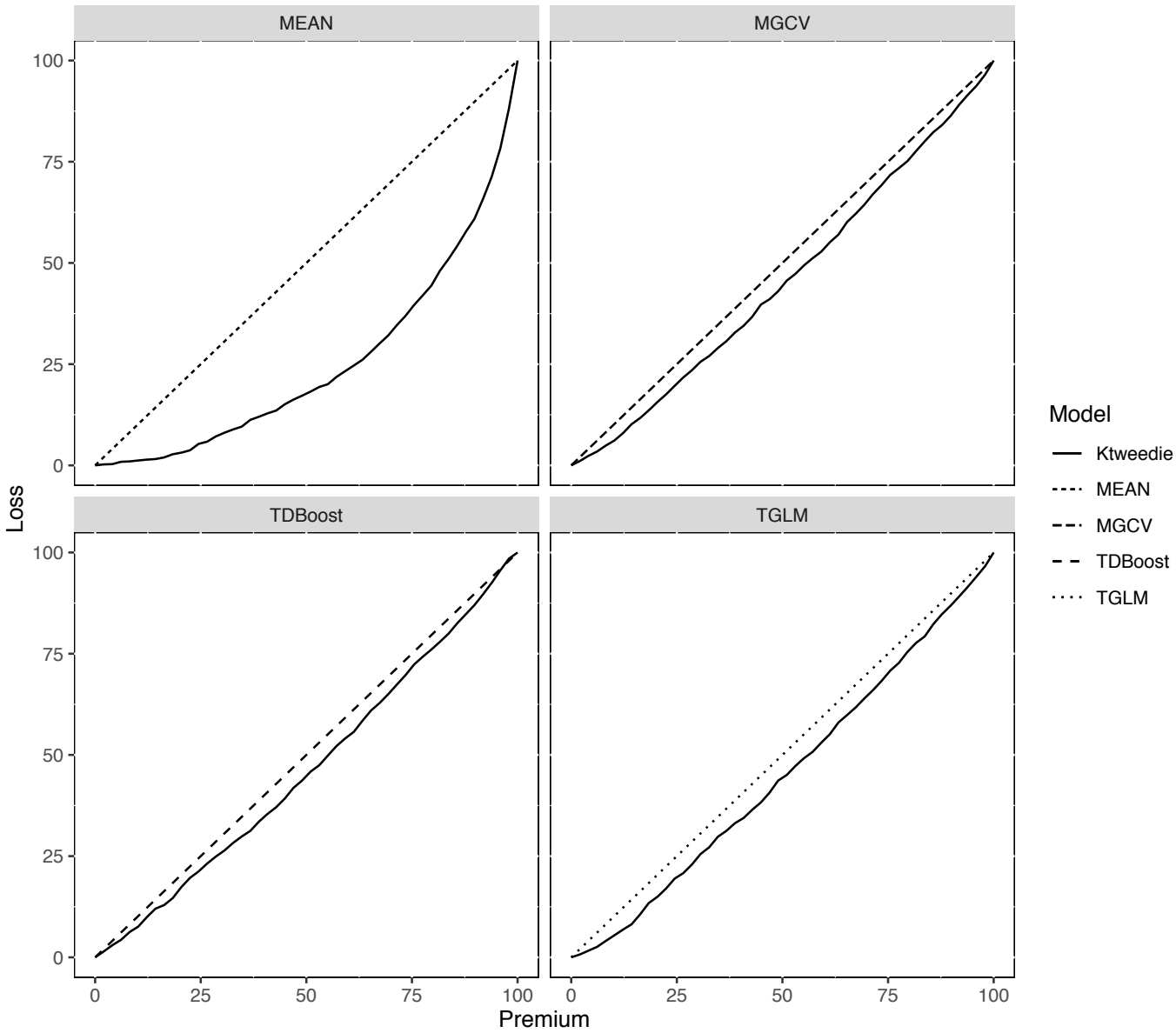


Figure S10: The ordered Lorenz curves for the auto-insurance claim data. In all four plots, the Ktweedie serves as the competing model.

References

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

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. E

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

Nocedal, J. and Wright, S. (2006) *Numerical optimization*. Springer Science & Business Media. C, C, C