Technometrics

ISSN: 0040-1706 (Print) 1537-2723 (Online) Journal homepage: https://www.tandfonline.com/loi/utch20

A Unified Approach to Sparse Tweedie Modeling of Multisource Insurance Claim Data

Simon Fontaine, Yi Yang, Wei Qian, Yuwen Gu & Bo Fan

To cite this article: Simon Fontaine, Yi Yang, Wei Qian, Yuwen Gu & Bo Fan (2019): A Unified Approach to Sparse Tweedie Modeling of Multisource Insurance Claim Data, Technometrics, DOI: 10.1080/00401706.2019.1647881

To link to this article: https://doi.org/10.1080/00401706.2019.1647881

TECHNOMETRICS
2019, VOL. 00, NO. 0, 1–18
https://doi.org/10.1080/00401706.2019.1647881

Taylor & Francis
Taylor & Francis Group

A Unified Approach to Sparse Tweedie Modeling of Multisource Insurance Claim Data

Simon Fontaine[a], Yi Yang[b], Wei Qian[c], Yuwen Gu[d], and Bo Fan[e]

[a]Department of Statistics, University of Michigan, Ann Arbor, MI; [b]Department of Mathematics and Statistics, McGill University, Montreal, Quebec, Canada; [c]Department of Applied Economics and Statistics, University of Delaware, Newark, DE; [d]Department of Statistics, University of Connecticut, Storrs, CT; [e]Department of Statistics, University of Oxford, Oxford, UK

ABSTRACT

Actuarial practitioners now have access to multiple sources of insurance data corresponding to various situations: multiple business lines, umbrella coverage, multiple hazards, and so on. Despite the wide use and simple nature of single-target approaches, modeling these types of data may benefit from an approach performing variable selection jointly across the sources. We propose a unified algorithm to perform sparse learning of such fused insurance data under the Tweedie (compound Poisson) model. By integrating ideas from multitask sparse learning and sparse Tweedie modeling, our algorithm produces flexible regularization that balances predictor sparsity and between-sources sparsity. When applied to simulated and real data, our approach clearly outperforms single-target modeling in both prediction and selection accuracy, notably when the sources do not have exactly the same set of predictors. An efficient implementation of the proposed algorithm is provided in our R package `MStweedie`, which is available at *https://github.com/fontaine618/MStweedie*. Supplementary materials for this article are available online.

ARTICLE HISTORY
Received June 2018
Accepted June 2019

KEYWORDS
Backtracking line search; Groupwise proximal gradient descent; Multisource insurance data; Multitask learning; Regularization; Tweedie model

1. Introduction

Insurance claim data are characterized by excess zeros, corresponding to insurance policies without any claims, and highly right-skewed positive values associated with nonzero claim amounts, typically in monetary value. The modeling of insurance claim data helps to predict the expected loss associated with a portfolio of policies and is widely used for premium pricing. As claim data reflect a unique mixed nature of distributions with both discrete and continuous components, there are generally two popular modeling approaches. The first type considers a frequency-severity approach where claim frequency (i.e., whether a claim exists or not) and claim amount are modeled separately (Yip and Yau 2005; Frees, Gao, and Rosenberg 2011; Shi, Feng, and Ivantsova 2015), so that the two models need to be used together for claim loss prediction. The second type uses Tweedie's compound Poisson model (or Tweedie model for short; Tweedie 1984) that considers an inherent Poisson process and models both components simultaneously. Our study will focus on the second approach that draws upon Tweedie distribution's natural structure for claim data modeling (Smyth and Jørgensen 2002; Frees, Meyers, and Cummings 2011; Zhang 2013; Shi, Feng, and Boucher 2016). It is also common practice that insurers collect and maintain external information associated with insurance policies either directly from policy holders or from third-party databases. Covariates generated from the external information can be associated with the claim loss and help improve the modeling process. Depending on the type of insurance, this information may consist of policyholder's characteristics (demographics, employment status, experience, etc.) of the insured object characteristics (car type and usage, property value, etc.) or of any other characteristic deemed relevant.

Traditionally, actuarial practitioners adopt a single-target approach that, for a given insurance product, assumes one population to be homogeneously characterized by some covariates and aims to build a single Tweedie model solely from the product's sample data. Despite the wide use and simple nature of this approach, practitioners now have access to multiple sources of insurance data. For instance, many insurers have multiple business lines such as the auto insurance and the property insurance; in umbrella coverage, claim amounts are available for multiple types of coverage and for different hazard causes of the same coverage; multiple datasets can be accumulated for a long period of time, during which business environment may have changed significantly so that earlier-year and later-year data sources may not be treated as one homogeneous population. As a result, the modern multisource insurance data may not be characterized well by a homogeneous model. With these emerging multisource insurance data problems, much attention has been drawn to addressing their modeling issues in statistics and actuarial science. Both the frequency-severity and Tweedie model approaches have been investigated in the context of multivariate regressions to model the multiple responses simultaneously (see Frees, Lee, and Yang 2016; Shi 2016 and references therein).

Variable selection is one of the most important tasks in building transparent and interpretable models for claim loss prediction. Large-scale high-dimensional sparse modeling is

CONTACT Yi Yang *yi.yang6@mcgill.ca* Department of Mathematics and Statistics, McGill University, Montreal, Quebec H3A 0G4, Canada.
Color versions of one or more of the figures in the article can be found online at *www.tandfonline.com/r/TECH*.
Supplementary materials for this article are available online. Please go to *www.tandfonline.com/r/TECH*.

commonly encountered as hundreds of covariates are often considered as candidate variables while only a few of them are believed to be associated with the claim loss or can be used in the final model production. Under the single population setting, efficient variable selection approaches designed for the Tweedie model have been developed via a shrinkage-type approach (see Qian, Yang, and Zou 2016 and references therein). The increasingly prevalent multisource data scenarios coupled with high dimensionality and large data scale pose new challenges to actuarial practitioners. To our knowledge, the corresponding variable selection issues for multisource Tweedie models have not been studied in the literature. On the one hand, simply treating all different data sources as if they were from one population is problematic due to severe model misspecification. On the other hand, it may not be ideal either to perform variable selection separately on each individual data source because it often results in a loss of estimation efficiency. In the aforementioned multiline, multitype, or multiyear scenarios, the different data sources often contain similar types of covariates and some (or all) of them can be relevant across some (or all) data sources, even if different data come from totally different sets of customers. For example, both auto and property insurance contain geographical, credit, and experience variables that may be important in both lines of business. Therefore, a proper variable selection process should ideally take advantage of the potential connections among data sources as opposed to simply treating each data source separately.

In this article, we augment the multisource claim data analysis through an integrated shrinkage-based Tweedie modeling approach that fuses different data sources to find commonly shared relevant covariates. To insure our method is plausible, we will assume that the different sources have some (if not all) covariates in common. At the same time, our method retains the ability to recover model structures and covariates unique to individual data sources. In particular, we impose a composite adaptive lasso-type penalty (Tibshirani 1996; Zou 2006; Simon et al. 2013) in the composite Tweedie model to obtain both common and source-specific variables simultaneously. We study several different candidate penalty terms for our multisource data setting and devise a new algorithm (named MStweedie) to efficiently solve the corresponding optimization problems in a unified fashion. Our proposal is closely related to the celebrated multitask lasso in Lounici et al. (2011) that intends to uncover shared information across different tasks while achieving improved estimation, selection and prediction efficiency compared to independent variable selection (Obozinski, Wainwright, and Jordan 2008; Lounici et al. 2009; Huang and Zhang 2010; Lounici et al. 2011) under least square settings. Different from the existing multitask learning and related studies that mainly focused on the least squares (see, e.g., Jenatton et al. 2010; Morales, Micchelli, and Pontil 2010; Kim and Xing 2012) or classification (see, e.g., Zhang et al. 2008; Friedman, Hastie, and Tibshirani 2010; Obozinski, Taskar, and Jordan 2010; Vincent and Hansen 2014; Qian et al. 2019) setting, our proposal solves the important challenges posed by the semi-continuous, highly right-skewed claim data with excess zeros which cannot be efficiently modeled by a Gaussian or logit distribution. In particular, we show that the MStweedie algorithm is theoretically guaranteed to converge to the optimization target with at least linear rate, and is practically flexible to handle source-specific missing covariates. In addition, we implement our proposal in an efficient and user-friendly R package called `MStweedie` (standing for multisource Tweedie modeling), which is available at *https://github.com/fontaine618/MStweedie*.

The article is organized as follows. In Section 2, we introduce the sparse Tweedie model for multisource claim data and derive a general objective function. Section 3 develops a unified algorithm to efficiently optimize that objective. Section 4 provides the details of implementation and tuning parameter selection for the proposed algorithm. In Section 5, we compare the performance of our proposal to other existing methods in a series of numerical experiments on both simulated and real data. Section 6 concludes the article. The technical proofs are relegated to the appendix (supplementary materials).

2. Methodology

2.1. Tweedie's Compound Poisson Model

The Tweedie model is closely related to the exponential dispersion models (EDM; Jørgensen 1987)

$$f_Y(y|\theta,\phi) = a(y,\phi)\exp\left\{\frac{y\theta-\kappa(\theta)}{\phi}\right\},$$

parameterized by the natural parameter θ and dispersion parameter ϕ, where $\kappa(\cdot)$ is the cumulant function and $a(\cdot)$ is the normalizing function. Both $a(\cdot)$ and $\kappa(\cdot)$ are known functions. It can be shown that Y has mean $\mu \equiv E(Y) = \dot\kappa(\theta)$ and variance $\text{var}(Y) = \phi\ddot\kappa(\theta)$, where $\dot\kappa(\theta)$ and $\ddot\kappa(\theta)$ denote the first and second derivatives of $\kappa(\theta)$, respectively. In this article, we are primarily interested in the Tweedie EDMs, a class of EDMs that have the mean-variance relationship $\text{var}(Y) = \phi\mu^\rho$, where ρ is the power parameter. Such mean-variance relation gives

$$\theta = \begin{cases} \frac{\mu^{1-\rho}}{1-\rho}, & \rho \neq 1 \\ \log\mu, & \rho = 1 \end{cases} \quad \text{and} \quad \kappa(\theta) = \begin{cases} \frac{\mu^{2-\rho}}{2-\rho}, & \rho \neq 2 \\ \log\mu, & \rho = 2 \end{cases}. \quad (1)$$

In particular, when $\rho \in (1,2)$, the Tweedie EDMs correspond to a family of distributions called the compound Poisson distributions. In the sequel, we briefly discuss the compound Poisson distributions and their connection to the Tweedie EDMs. A compound Poisson random variable can be written as the sum of a (random) Poisson number of Gamma random variables. Specifically, let $Z_1, Z_2, \ldots, Z_N$ be N iid random variables from Gamma(α,γ), where N follows Poisson(λ). We assume that the Z_i's are independent of N. Then the sum of the Z_i's

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

follows the compound Poisson distribution

$$f_Y(y|\lambda,\alpha,\gamma) = P(N=0)\delta_0(y) + \sum_{j=1}^{\infty} P(N=j) f_{Y|N=j}(y)$$
$$= e^{-\lambda}\delta_0(y) + \sum_{j=1}^{\infty} \frac{\lambda^j y^{j\alpha-1} e^{-\lambda-y/\gamma}}{j!\gamma^{j\alpha}\Gamma(j\alpha)},$$

where δ_0 is the Dirac delta mass at zero, $f_{Y|N=j}(\cdot)$ is the conditional density of Y given $N = j$, and $\Gamma(\cdot)$ is the gamma function. The compound Poisson distributions fit into a special class of Tweedie EDMs with $\rho \in (1, 2)$. To see this, we reparameterize $(\lambda, \gamma, \alpha)$ by

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

The compound Poisson model will then have the form

$$\log f_Y(y|\mu,\phi,\rho) = \frac{1}{\phi}\left(y\frac{\mu^{1-\rho}}{1-\rho} - \frac{\mu^{2-\rho}}{2-\rho}\right) + \log a_\rho(y,\phi), \quad (3)$$

where

$$a_\rho(y,\phi) = \begin{cases} \frac{1}{y}\sum_{j=1}^{\infty} \frac{y^{j\alpha}}{j!(\rho-1)^{j\alpha}\phi^{j(\alpha+1)}\Gamma(j\alpha)} & \text{if } y > 0, \\ 1 & \text{if } y = 0. \end{cases}$$

It can be directly seen that (3) belongs to the Tweedie EDMs. As a result, for the rest of this article, we simply refer to (3) as Tweedie's compound Poisson model (or the Tweedie model), and denote it by $\text{Tw}(\mu, \phi, \rho)$, where $1 < \rho < 2$.

This equivalence provides a very intuitive justification for the use of the Tweedie distribution in modeling insurance claim data: the random variable N corresponds to the number of claims during the exposure period, $Z_1, \ldots, Z_N$ correspond to the claim amounts, and $Y = \sum_{j=1}^{N} Z_j$ then corresponds to the aggregate claim amount. The case $Y = 0$ represents the absence of claims during the exposure period and is a frequent situation for this type of data.

2.2. A Sparse Tweedie Modeling Framework for Multisource Claim Data

Suppose the claim data consist of K data sources (possibly from different policy products), and each data source k $(1 \le k \le K)$ has n_k policies. Given any policy i in data source k, assume exposure is $w_i^{(k)}$. For data source k, denote by $\tilde{Y}_i^{(k)} = \sum_{j=1}^{N_i^{(k)}} Z_{i,j}^{(k)}$ the corresponding claim loss, where $N_i^{(k)}$ is the claim frequency and the $Z_{i,j}^{(k)}$'s are the claim severity. The goal is to model the pure premium $Y_i^{(k)} = \tilde{Y}_i^{(k)}/w_i^{(k)}$. Here, the exposure is a known measure of certain risk in force (e.g., the exposure of a personal auto insurance can be the policy duration) so that in the Tweedie model, we assume $N_i^{(k)} \sim \text{Poisson}(\lambda_i^{(k)} w_i^{(k)})$ and $\tilde{Y}_{i,j}^{(k)} \mid N_i^{(k)} \sim \text{Gamma}(\alpha, \gamma_i^{(k)})$, where $\lambda_i^{(k)}$ represents a policy-specific parameter for the expected claim frequency under unit exposure, $\gamma_i^{(k)}$ is a policy-specific parameter for claim severity, and α is a known scalar (Dunn and Smyth 2005). Further assume a mean-variance relation $\text{var}(Y_i^{*(k)}) = \phi^{(k)}\{E(Y_i^{*(k)})\}^\rho$, where $Y_i^{*(k)}$ is the pure premium under unit exposure (that is, $w_i^{(k)} = 1$) and $\phi^{(k)}$ is the source-specific dispersion. Then we have $Y_i^{(k)} \sim \text{Tw}(\mu_i^{(k)}, \phi^{(k)}/w_i^{(k)}, \rho)$ with $\mu_i^{(k)} = E(Y_i^{(k)})$ (Smyth and Jørgensen 2002; Yang, Qian, and Zou 2018).

Suppose that each policy i in data source k has p covariates $\mathbf{x}_i^{(k)} = (x_{i1}^{(k)}, \ldots, x_{ip}^{(k)})^\top$. For brevity, we assume these covariates are of the same type with equal dimension across different data sources, but as will be discussed in our numerical studies, we can generalize this setting to handle possibly unequal dimension scenarios. We adopt the commonly used multiplicative logarithmic link

$$\log \mu_i^{(k)} = \eta_i^{(k)} = \beta_0^{(k)} + \mathbf{x}_i^{(k)\top}\boldsymbol{\beta}^{(k)},$$

where $\boldsymbol{\beta}^{(k)} = (\beta_1^{(k)}, \ldots, \beta_p^{(k)})^\top$ with $\beta_j^{(k)}$ being the jth element of $\boldsymbol{\beta}^{(k)}$, $j = 1, \ldots, p$. Let $\boldsymbol{\beta}_0 = (\beta_0^{(1)}, \ldots, \beta_0^{(K)})^\top$, $\boldsymbol{\beta}_j = (\beta_j^{(1)}, \ldots, \beta_j^{(K)})^\top$, and $\boldsymbol{\beta} = (\boldsymbol{\beta}_1^\top, \ldots, \boldsymbol{\beta}_p^\top)^\top \in \mathbb{R}^{pK}$ be the target coefficient parameters. Assume that only a small fraction of the covariates in $\mathbf{x}_i^{(k)}$ are relevant to $y_i^{(k)}$ so that many elements in $\boldsymbol{\beta}^{(k)}$ are zero. Given K sources of polices $\mathbf{D}^{(k)} = \{(y_i^{(k)}, \mathbf{x}_i^{(k)}, w_i^{(k)})\}_{i=1}^{n_k}$ for $k = 1, \ldots, K$, the multisource data setting naturally leads to a composite objective function

$$L(\boldsymbol{\beta}_0, \boldsymbol{\beta}) = \prod_{k=1}^{K}\prod_{i=1}^{n_k} f_Y(y_i^{(k)} \mid \mu_i^{(k)}, \phi^{(k)}/w_i^{(k)}, \rho), \quad (4)$$

which, assuming independence across different data sources, becomes the likelihood function. When the independence assumption is violated, (4) can still be viewed as a composite marginal likelihood (Varin, Reid, and Firth 2011), the study of which plays an important role in allowing feasible estimation of marginal parameters (see, e.g., Chandler and Bate 2007; Shi 2016). Without loss of generality, we assume same dispersion $\phi = \phi^{(1)} = \cdots = \phi^{(K)}$ across all data sources (otherwise, we can simply adjust $w_i^{(k)}$'s in (5) accordingly). Hence, we assume common dispersion parameters between subjects; it does not much affect the modeling of the mean since the two parameters, μ and ϕ, are orthogonal to each other from the compound Poisson-Gamma distribution (Shi 2016). Taking negative logarithm and omitting constant terms, we obtain the following objective function (up to a dispersion scalar)

$$\ell(\boldsymbol{\beta}_0, \boldsymbol{\beta}) = \sum_{k=1}^{K}\sum_{i=1}^{n_k} w_i^{(k)} \left\{ -\frac{y_i^{(k)} e^{(1-\rho)\eta_i^{(k)}}}{1-\rho} + \frac{e^{(2-\rho)\eta_i^{(k)}}}{2-\rho} \right\}, \quad (5)$$

which is the negative log-likelihood under the independence assumption and is a convex objective.

To take advantage of the commonly shared relevant covariates while recovering source-specific model structures, we consider the composite penalty (Zhao, Rocha, and Yu 2009)

$$P_\alpha(\boldsymbol{\beta}) = \sum_{j=1}^{p} \nu_j \left[(1-\alpha)||\boldsymbol{\beta}_j||_q + \alpha||\boldsymbol{\beta}_j||_1 \right]$$

for some $0 \le \alpha \le 1$ and $q \in \{2, \infty\}$, where the ν_j's are the penalty weights. The first component in $P_\alpha(\boldsymbol{\beta})$ is aimed to find common relevant covariates across data sources and the second component is intended to deal with potential between-source differences in sparsity and to find source-specific relevant covariates. When $\alpha = 0$, $P_\alpha(\boldsymbol{\beta})$ simplifies to the group lasso if $q = 2$ (Yuan and Lin 2006), while it gives a different "group discount" if $q = \infty$ as only the largest coefficient is penalized (Obozinski, Taskar, and Jordan 2006). In both cases, when the jth covariate is selected by the model, then it is selected for all sources, which means that the coefficient $\beta_j^{(k)}$ will be nonzero

for all sources $k = 1, \ldots, K$. When $0 < \alpha < 1$ and $q = 2$, $P_\alpha(\boldsymbol{\beta})$ becomes the sparse group lasso (Simon et al. 2013). In that case, when the jth feature is included in the model, the q-norm of $\boldsymbol{\beta}_j$ is still freed from zero, but each individual coefficient are individually penalized so that some components of $\boldsymbol{\beta}_j$ may still remain zero even though other components will be nonzero; selecting the jth feature only ensure that the coefficient for at least one (but not necessarily all) source is nonzero. The use of the penalty weights is motivated from the adaptive Lasso (Zou 2006) for improved variable selection performance. Our integral approach to sparse Tweedie modeling for multisource data aims to solve the regularized objective

$$f^* = \min_{\boldsymbol{\beta}_0, \boldsymbol{\beta}} f(\boldsymbol{\beta}_0, \boldsymbol{\beta}), \quad f(\boldsymbol{\beta}_0, \boldsymbol{\beta}) = \ell(\boldsymbol{\beta}_0, \boldsymbol{\beta}) + \lambda P_\alpha(\boldsymbol{\beta}), \tag{6}$$

where $\lambda > 0$ is the tuning parameter. We call (6) the $L_1/L_q(\alpha)$ regularization, and when $\alpha = 0$, we simply call it L_1/L_q regularization ($q = 2$ or ∞).

As in most multitask learning algorithms, our proposed approach has a single-task reinterpretation. Indeed, similar to Turlach, Venables, and Wright (2005), we can model the response $y_i^{(k)}$ through the equivalent log-link for the mean

$$\log \mu_i^{(k)} = \sum_{l=1}^{K} \mathbb{I}(l = k)\, \beta_0^{(k)} + \left[\mathbb{I}(l = k)\, \mathbf{x}_i^{(k)}\right]^\top \boldsymbol{\beta}^{(k)} = \tilde{\mathbf{x}}_i^{(k)\top} \boldsymbol{\beta},$$

where

$$\tilde{\mathbf{x}}_i^{(k)\top} = \left(0, \mathbf{0}_K^\top, \ldots, 1, \underbrace{\mathbf{x}_i^{(k)\top}}_{\text{position } k+1}, \ldots, 0, \mathbf{0}_K^\top\right),$$
$$\boldsymbol{\beta}^\top = \left(\beta_0^{(1)}, \boldsymbol{\beta}^{(1)\top}, \ldots, \beta_0^{(k)}, \boldsymbol{\beta}^{(k)\top}, \ldots, \beta_0^{(K)}, \boldsymbol{\beta}^{(K)\top}\right).$$

Hence, our model can be seen has a single-source Tweedie model with $K \cdot p$ features (plus the K intercepts) where a response $y_i^{(k)}$ has nonzero features only for the group of feature related to the kth source. The penalty $P_\alpha(\boldsymbol{\beta})$ then correspond to a sparse group Lasso penalty where the groups are defined by grouping a feature across sources. The novelty of our algorithm is actually to profit from the specific structure of the groups and dummy variables to improve the efficiency of the algorithm. Fitting a single Tweedie with such a description of the groups would be inefficient computationally since a large proportion of the entries of the design matrix would artificially be zeroes.

3. Algorithm

In this section, we propose an efficient algorithm to solve the penalized composite Tweedie model (6). We decompose the description of our algorithm into four parts: Section 3.1 gives a general idea on how to solve our optimization problem via the cyclic groupwise proximal gradient (GPG) descent; Section 3.2 discusses an acceleration scheme for the proposed algorithm; and Section 3.3 provides detailed solutions to the L_1/L_q regularization, which gives necessary information to introduce our complete algorithm to solve the more general $L_1/L_q(\alpha)$ regularization in Section 3.4.

For data source k, denote the response vector by $Y^{(k)} = (y_1^{(k)}, \ldots, y_{n_k}^{(k)})^\top$ and the $n_k \times p$ design matrix by $\mathbf{X}^{(k)} = (\mathbf{x}_1^{(k)}, \ldots, \mathbf{x}_{n_k}^{(k)})^\top = (X_1^{(k)}, \ldots, X_p^{(k)})$. For consistency of notation, we also let $X_0^{(k)} = \mathbf{1}_{n_k}$.

3.1. A Groupwise Proximal Gradient Algorithm for MStweedie

Note that the penalty term $P_\alpha(\boldsymbol{\beta})$ in (6) is separable with respect to the indices of the feature sets $j = 1, \ldots, p$. We exploit this property and propose to iteratively update and cycle through the $\boldsymbol{\beta}_j$'s ($j = 0, 1, \ldots, p$) via the proximal gradient (Beck and Teboulle 2009) scheme which gives rise to a cyclic GPG algorithm designed for MStweedie. Specifically, let $\tilde{\mathbf{b}}$ be the current iterate

$$\tilde{\mathbf{b}} \equiv (\tilde{\boldsymbol{\beta}}_0, \ldots, \tilde{\boldsymbol{\beta}}_{j-1}, \tilde{\boldsymbol{\beta}}_j, \tilde{\boldsymbol{\beta}}_{j+1}, \ldots, \tilde{\boldsymbol{\beta}}_p)^\top,$$

and $\tilde{\mathbf{b}}_{-j}$ be the current iterate with the jth group excluded

$$\tilde{\mathbf{b}}_{-j} \equiv (\tilde{\boldsymbol{\beta}}_0, \ldots, \tilde{\boldsymbol{\beta}}_{j-1}, \tilde{\boldsymbol{\beta}}_{j+1}, \ldots, \tilde{\boldsymbol{\beta}}_p)^\top, \quad j = 0, \ldots, p.$$

Suppose we are about to update the group $\boldsymbol{\beta}_j = (\beta_j^{(1)}, \ldots, \beta_j^{(K)})^\top$ for some $j \in \{0, 1, \ldots, p\}$. View the negative log-likelihood function $\ell(\boldsymbol{\beta}_0, \boldsymbol{\beta})$ in (5) as a function of the jth group $\boldsymbol{\beta}_j$, while keeping all the other groups fixed at $\tilde{\mathbf{b}}_{-j}$, that is, $\ell(\boldsymbol{\beta}_j; \tilde{\mathbf{b}}_{-j}) = \ell(\boldsymbol{\beta}_0, \boldsymbol{\beta})|_{\boldsymbol{\beta}_m = \tilde{\boldsymbol{\beta}}_m, 0 \le m \le p, m \ne j}$. For group j, note that a quadratic approximation to $\ell(\boldsymbol{\beta}_j; \tilde{\mathbf{b}}_{-j})$ around $\tilde{\boldsymbol{\beta}}_j$ is given by

$$\begin{aligned} &\ell(\boldsymbol{\beta}_j; \tilde{\mathbf{b}}_{-j}) \approx \ell_{Q_j}(\boldsymbol{\beta}_j; \tilde{\mathbf{b}}, t_j) \\ &\equiv \ell(\tilde{\mathbf{b}}) + \nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})^\top (\boldsymbol{\beta}_j - \tilde{\boldsymbol{\beta}}_j) + \frac{1}{2t_j} \|\boldsymbol{\beta}_j - \tilde{\boldsymbol{\beta}}_j\|_2^2, \; t_j > 0. \end{aligned} \tag{7}$$

It can be seen that $\ell_{Q_j}(\boldsymbol{\beta}_j; \tilde{\mathbf{b}}, t_j) = \ell(\boldsymbol{\beta}_j; \tilde{\mathbf{b}}_{-j})$ when $\boldsymbol{\beta}_j = \tilde{\boldsymbol{\beta}}_j$ for any $t_j > 0$. To ensure the convergence of the algorithm, the value of t_j can be determined using the backtracking line search (details given later in this section). In (7), the gradient $\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})$ can be written explicitly as

$$\begin{aligned} &\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j}) \\ &= \frac{\partial}{\partial \boldsymbol{\beta}_j} \ell(\boldsymbol{\beta}_j; \tilde{\mathbf{b}}_{-j})\Big|_{\boldsymbol{\beta}_j = \tilde{\boldsymbol{\beta}}_j} = \left((\tilde{\boldsymbol{\eta}}^{(k)} - \tilde{\mathbf{z}}^{(k)})^\top \widetilde{\mathbf{W}}^{(k)} X_j^{(k)}\right)_{k=1}^K, \end{aligned} \tag{8}$$

where $\tilde{\boldsymbol{\eta}}^{(k)} = (\tilde{\eta}_1^{(k)}, \ldots, \tilde{\eta}_{n_k}^{(k)})^\top$ with $\tilde{\eta}_i^{(k)} = \sum_{j=0}^p x_{ij}^{(k)} \tilde{\beta}_j^{(k)}$, $\tilde{\mathbf{z}}^{(k)} = (\tilde{z}_1^{(k)}, \ldots, \tilde{z}_{n_k}^{(k)})^\top$ with

$$\tilde{z}_i^{(k)} = \tilde{\eta}_i^{(k)} + \frac{w_i^{(k)}}{\tilde{w}_i^{(k)}} \left(y_i^{(k)} e^{(1-\rho)\tilde{\eta}_i^{(k)}} - e^{(2-\rho)\tilde{\eta}_i^{(k)}}\right), \tag{9}$$

and $\widetilde{\mathbf{W}}^{(k)} = \operatorname{diag}(\tilde{w}_1^{(k)}, \ldots, \tilde{w}_{n_k}^{(k)})$ with

$$\tilde{w}_i^{(k)} = w_i^{(k)} \left((\rho - 1) y_i^{(k)} e^{(1-\rho)\tilde{\eta}_i^{(k)}} + (2 - \rho) e^{(2-\rho)\tilde{\eta}_i^{(k)}}\right). \tag{10}$$

Now we apply the proximal gradient algorithm on $\ell_{Q_j}(\boldsymbol{\beta}_j; \tilde{\mathbf{b}}, t_j)$ to update $\boldsymbol{\beta}_j$ as follows. For $\tau > 0$, define the proximal mapping

of $h(\cdot) = (1-\alpha)||\cdot||_q + \alpha||\cdot||_1$ as the minimizer of the following problem

$$\text{prox}_{\tau h}(\mathbf{u}) = \arg\min_{\mathbf{v}} \left(\tau h(\mathbf{v}) + \frac{1}{2}\|\mathbf{v}-\mathbf{u}\|_2^2 \right). \quad (11)$$

For now, suppose that the solution to (11) is given (methods for computing the minimizer is deferred to Sections 3.3 and 3.4). We update $\boldsymbol{\beta}_j$ by minimizing the following penalized problem

$$\begin{aligned}\boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j) &= \arg\min_{\boldsymbol{\beta}_j} \ell_{Q_j}(\boldsymbol{\beta}_j; \tilde{\mathbf{b}}, t_j) + \lambda P_{\alpha,j}(\boldsymbol{\beta}_j) \\ &= \arg\min_{\boldsymbol{\beta}_j} \frac{1}{2}\|\boldsymbol{\beta}_j - (\tilde{\boldsymbol{\beta}}_j - t_j \nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j}))\|_2^2 + \lambda v_j t_j h(\boldsymbol{\beta}_j) \\ &= \text{prox}_{\lambda v_j t_j h}(\tilde{\boldsymbol{\beta}}_j - t_j \nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})). \quad (12)\end{aligned}$$

Note that in (12), when $j = 0$, we have $\boldsymbol{\beta}_0^+(\tilde{\mathbf{b}}, t_0) = \tilde{\boldsymbol{\beta}}_0 - t_0 \nabla_j \ell(\tilde{\boldsymbol{\beta}}_0; \tilde{\mathbf{b}}_{-0})$, since the intercept term is not penalized, that is, $P_{\alpha,0}(\boldsymbol{\beta}_0) \equiv 0$.

To guarantee convergence, we determine the step size t_j in (12) using backtracking line search. Define

$$\begin{aligned}G_{t_j}(\tilde{\boldsymbol{\beta}}_j) &= \frac{1}{t_j}\{\tilde{\boldsymbol{\beta}}_j - \boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j)\} \\ &= \frac{1}{t_j}\left\{\tilde{\boldsymbol{\beta}}_j - \text{prox}_{\lambda v_j t_j h}(\tilde{\boldsymbol{\beta}}_j - t_j \nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j}))\right\}.\end{aligned}$$

We initialize t_j with some $t_{\max} > 0$ and repeatedly shrink t_j with $t_j \leftarrow \delta t_j$ for some prechosen $0 < \delta < 1$ until

$$\begin{aligned}\ell(\boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j)) &= \ell(\tilde{\boldsymbol{\beta}}_j - t_j G_{t_j}(\tilde{\boldsymbol{\beta}}_j)) \quad (13) \\ &\le \ell(\tilde{\mathbf{b}}) - t_j \nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})^\top G_{t_j}(\tilde{\boldsymbol{\beta}}_j) + \frac{t_j}{2}\|G_{t_j}(\tilde{\boldsymbol{\beta}}_j)\|_2^2.\end{aligned}$$

Once (13) is satisfied by $\boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j)$ for some t_j, we set $\tilde{\boldsymbol{\beta}}_j \leftarrow \boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j)$ and move on to the next group $j+1$ and compute the update $\boldsymbol{\beta}_{j+1}^+(\tilde{\mathbf{b}}, t_{j+1})$. The algorithm cyclically updates groups $j = 0, 1, \ldots, p, 0, 1, \ldots, p, \ldots$ until convergence of $(\tilde{\boldsymbol{\beta}}_0, \tilde{\boldsymbol{\beta}})$.

We summarize our proposal above with backtracking line search in Algorithm 1, and call it MStweedie-GPG for short. Moreover, we show that the proposed iterative approach is guaranteed to converge with at least linear rate in the following theorem, whose proof can be found in Appendix D (supplementary materials).

Theorem 1. In the MStweedie-GPG algorithm, let $(\boldsymbol{\beta}_0^{(r)}, \boldsymbol{\beta}^{(r)})$ be the update of $(\boldsymbol{\beta}_0, \boldsymbol{\beta})$ after the rth cycle, $r \ge 0$. The algorithm with backtracking line search converges to the global minimum f^* of (6) with at least a linear rate of convergence, that is,

$$f(\boldsymbol{\beta}_0^{(r+1)}, \boldsymbol{\beta}^{(r+1)}) - f^* \le c(f(\boldsymbol{\beta}_0^{(r)}, \boldsymbol{\beta}^{(r)}) - f^*)$$

for large enough r, where $c \in (0, 1)$ is a constant.

Algorithm 1: MStweedie-GPG with backtracking line search.

1 Initialize the coefficients with $(\tilde{\boldsymbol{\beta}}_0, \tilde{\boldsymbol{\beta}})$ and choose some $0 < \delta < 1$;
2 **repeat**
3 **for** $j = 0, 1, 2, \ldots, p$ **do**
4 Initialize t_j with $t_{\max} > 0$;
5 **repeat**
6 Compute $\boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j) = \text{prox}_{\lambda v_j t_j h}(\tilde{\boldsymbol{\beta}}_j - t_j \nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j}))$ using the proximal operator in (21), where $\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})$ is calculated from (8);
7 Compute $G_{t_j}(\tilde{\boldsymbol{\beta}}_j) = \frac{1}{t_j}\{\tilde{\boldsymbol{\beta}}_j - \boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j)\}$;
8 Set $t_j \leftarrow \delta t_j$;
9 **until** $\ell(\tilde{\boldsymbol{\beta}}_j - t_j G_{t_j}(\tilde{\boldsymbol{\beta}}_j)) < \ell(\tilde{\mathbf{b}}) - t_j \nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})^\top G_{t_j}(\tilde{\boldsymbol{\beta}}_j) + \frac{t_j}{2}\|G_{t_j}(\tilde{\boldsymbol{\beta}}_j)\|_2^2$;
10 Set $\tilde{\boldsymbol{\beta}}_j \leftarrow \boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j)$;
11 **end**
12 **until** *Convergence of* $(\tilde{\boldsymbol{\beta}}_0, \tilde{\boldsymbol{\beta}})$;
13 Return$(\tilde{\boldsymbol{\beta}}_0, \tilde{\boldsymbol{\beta}})$;
14 Note: when $j = 0$, $\boldsymbol{\beta}_0^+(\tilde{\mathbf{b}}, t_0) = \tilde{\boldsymbol{\beta}}_0 - t_0 \nabla_0 \ell(\tilde{\boldsymbol{\beta}}_0; \tilde{\mathbf{b}}_{-0})$ and $G_{t_0}(\tilde{\boldsymbol{\beta}}_0) = \nabla_0 \ell(\tilde{\boldsymbol{\beta}}_0; \tilde{\mathbf{b}}_{-0})$

3.2. Accelerated MStweedie-GPG

In the vanilla MStweedie-GPG algorithm, operation (13) for backtracking is repeatedly evaluated during each groupwise update, and is thus computationally expensive. We can accelerate our algorithm by fixing the step sizes and only update them after $(\boldsymbol{\beta}_0, \boldsymbol{\beta})$ converges in each loop. Specifically, instead of searching for a new step size to update $\boldsymbol{\beta}_j$ during each iteration within a loop, we use a fixed step size t_j^* as follows: given $(\tilde{\boldsymbol{\beta}}_0, \tilde{\boldsymbol{\beta}})$ at the beginning of each loop, we set the step sizes to $t_j^* = \sigma_j^{-1}$ for $j = 0, 1, \ldots, p$, where σ_j is the largest element of $\nabla_j^2 \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})$ with

$$\begin{aligned}\nabla_j^2 \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j}) &= \frac{\partial^2}{\partial \boldsymbol{\beta}_j \partial \boldsymbol{\beta}_j^\top} \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j}) \\ &= \text{diag}\big(X_j^{(k)\top} \widetilde{\mathbf{W}}^{(k)} X_j^{(k)}, k = 1, \ldots, K\big). \quad (14)\end{aligned}$$

Next, we make the cyclic updates $\tilde{\boldsymbol{\beta}}_j \leftarrow \boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j^*)$ with

$$\boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j^*) = \text{prox}_{\lambda v_j \sigma_j^{-1} h}(\tilde{\boldsymbol{\beta}}_j - \sigma_j^{-1} \nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})), \quad (15)$$

for $j = 0, 1, \ldots, p, 0, 1, \ldots, p \ldots$ until $(\tilde{\boldsymbol{\beta}}_0, \tilde{\boldsymbol{\beta}})$ converges during this loop. Then we recompute step sizes t_j^* using (14) and repeat the above process. We refer to this scheme as the Accelerated MStweedie-GPG (or MStweedie-AGPG for short). We summarize this practically important acceleration strategy in Algorithm 2. It can be seen that the algorithm only updates step sizes after $(\tilde{\boldsymbol{\beta}}_0, \tilde{\boldsymbol{\beta}})$ converges in the sub-iteration 2(b) of Algorithm 2. A similar technique for accelerating coordinate descent algorithms can be found in Friedman, Hastie, and Tibshirani (2010). Our empirical evidence shows that MStweedie-AGPG

converges very fast and follows an overall descending trend; see Figure A1 in the appendix (supplementary materials) for an illustration. This is the algorithm we use for all our numerical studies.

Algorithm 2: Accelerated MStweedie-GPG.

Initialize the coefficients with $(\tilde{\boldsymbol{\beta}}_0, \tilde{\boldsymbol{\beta}})$;
repeat
  Compute step sizes $t_j^* = \sigma_j^{-1}$ for $j = 0, 1, \ldots, p$, where σ_j is defined in (14);
  repeat
    for $j = 0, 1, 2, \ldots, p$ **do**
      Update $\tilde{\boldsymbol{\beta}}_j \leftarrow \boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j^*) = \text{prox}_{\lambda v_j \sigma_j^{-1} h}(\tilde{\boldsymbol{\beta}}_j - \sigma_j^{-1}\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j}))$ with the fixed step sizes $t_j^* = \sigma_j^{-1}$, where the proximal operator is given in (21) and $\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})$ is calculated from (8);
    end
  until *Convergence of* $(\tilde{\boldsymbol{\beta}}_0, \tilde{\boldsymbol{\beta}})$;
until *Convergence of* $(\tilde{\boldsymbol{\beta}}_0, \tilde{\boldsymbol{\beta}})$;
Return$(\tilde{\boldsymbol{\beta}}_0, \tilde{\boldsymbol{\beta}})$;

3.3. L_1/L_q Regularization

In the unified algorithm of Section 3.1, it remains to show how to solve (12). We first discuss the L_1/L_q regularization case ($\alpha = 0$), which will be used in the next subsection to derive solutions to the more general $L_1/L_q(\alpha)$ regularization with $\alpha \in [0, 1]$.

The following lemma translates the proximal operator of the L_∞ regularization ($q = \infty$) into a projection. Its proof is given in Appendix A (supplementary materials).

Lemma 1. The minimization problem

$$\boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j) = \arg\min_{\boldsymbol{\beta}_j} \frac{1}{2}\|\boldsymbol{\beta}_j - (\tilde{\boldsymbol{\beta}}_j - t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j}))\|_2^2 + \lambda v_j t_j \|\boldsymbol{\beta}_j\|_\infty \tag{16}$$

is equivalent to

$$\boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j) = \tilde{\boldsymbol{\beta}}_j - t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j}) - \text{Proj}_{B_1(\lambda v_j t_j)}(\tilde{\boldsymbol{\beta}}_j - t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})), \tag{17}$$

where $\text{Proj}_{B_1(\tau)}(\cdot)$ is the L_2-projection onto $B_1(\tau) = \{\mathbf{v} \mid ||\mathbf{v}||_1 \le \tau\}$, the L_1-ball with radius τ.

We use an extension of the algorithm suggested by Duchi et al. (2008) to perform fast projections onto the L_1-ball (see Appendix A for details in the supplementary materials). The KKT conditions of (16) can be shown (see Appendix B for details in the supplementary materials) as follows

$$\begin{cases} \|\tilde{\boldsymbol{\beta}}_j - t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})\|_1 \le \lambda v_j t_j, & \boldsymbol{\beta}_j = \mathbf{0}, \\ \|\tilde{\boldsymbol{\beta}}_j - t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j}) - \boldsymbol{\beta}_j\|_1 = \lambda v_j t_j, & \boldsymbol{\beta}_j \ne \mathbf{0}, \\ \tilde{\beta}_j^{(k)} - t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})^{(k)} - \beta_j^{(k)} = 0, & \boldsymbol{\beta}_j \ne \mathbf{0}, k \notin M(\boldsymbol{\beta}_j), \end{cases} \tag{18}$$

where $M(\boldsymbol{\beta}_j) = \{k \in \{1, \ldots, K\} : \|\boldsymbol{\beta}_j\|_\infty = |\beta_j^{(k)}|\}$ is the maximizing index set.

Next, we still assume $\alpha = 0$ and briefly discuss the L_2 regularization case ($q = 2$) in (12). We will omit most of the details and focus only on its differences from the L_1/L_∞ case. The minimizer of the penalized objective

$$\boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j) = \arg\min_{\boldsymbol{\beta}_j} \frac{1}{2}\|\boldsymbol{\beta}_j - (\tilde{\boldsymbol{\beta}}_j - t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j}))\|_2^2 + \lambda v_j t_j \|\boldsymbol{\beta}_j\|_2$$

has closed form

$$\boldsymbol{\beta}_j^+(\tilde{\mathbf{b}}, t_j) = (\|\tilde{\boldsymbol{\beta}}_j - t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})\|_2 - \lambda v_j t_j)_+ \frac{\tilde{\boldsymbol{\beta}}_j - t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})}{\|\tilde{\boldsymbol{\beta}}_j - t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})\|_2}, \tag{19}$$

and the corresponding KKT conditions are

$$\begin{cases} \|\tilde{\boldsymbol{\beta}}_j - t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})\|_2 \le \lambda v_j t_j, & \boldsymbol{\beta}_j = \mathbf{0}, \\ \lambda v_j t_j \frac{\boldsymbol{\beta}_j}{\|\boldsymbol{\beta}_j\|_2} + t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j}) + (\boldsymbol{\beta}_j - \tilde{\boldsymbol{\beta}}_j) = \mathbf{0}_K, & \boldsymbol{\beta}_j \ne \mathbf{0}. \end{cases} \tag{20}$$

3.4. $L_1/L_q(\alpha)$ Regularization

With the L_1/L_q regularization discussed in the previous subsection to take advantage of possibly common covariates across data sources, we are now ready to discuss the more general $L_1/L_q(\alpha)$ regularization ($0 \le \alpha \le 1$) to achieve the goal of uncovering relevant covariates unique to some data source.

It could seem complicated to derive a closed form expression of the above proximal operator (the Fenchel conjugate of f cannot be derived explicitly), but it is possible to solve it with a proximal technique originally developed for the hierarchical group lasso (Jenatton et al. 2010). Specifically, we rewrite our composite penalty as a sum of L_q-norms ($q = 2$ or ∞) on a set of groups $\mathcal{G}$ that is tree-structured by noting that $\|\boldsymbol{\beta}_j\|_1$ is separable across $k = 1, \ldots, K$

$$\begin{aligned}(1-\alpha)\|\boldsymbol{\beta}_j\|_q + \alpha\|\boldsymbol{\beta}_j\|_1 &= (1-\alpha)\|\boldsymbol{\beta}_j\|_q + \alpha\sum_{k=1}^K |\beta_j^{(k)}| \\ &= (1-\alpha)\|\boldsymbol{\beta}_j\|_q + \sum_{k=1}^K \alpha\|\beta_j^{(k)}\|_q,\end{aligned}$$

where we can identify $\mathcal{G} = \{\{1\}, \ldots, \{K\}, \{1, \ldots, K\}\}$, which is tree-structured. Consequently, we only require the proximal operator of each norm and compose them according to the tree ordering. Let $\mathbf{u} = \tilde{\boldsymbol{\beta}}_j - t_j\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})$ and $\tau = \lambda v_j t_j$. It is known from Section 3.3 that the proximal operator of $(1-\alpha)\tau\|.\|_q$ is

$$\text{prox}_{(1-\alpha)\tau\|\cdot\|_q}(\mathbf{u}) = \begin{cases} \mathbf{u} - \text{Proj}_{B_1((1-\alpha)\tau)}(\mathbf{u}), & q = \infty, \\ (\|\mathbf{u}\|_2 - (1-\alpha)\tau)_+ \frac{\mathbf{u}}{\|\mathbf{u}\|_2}, & q = 2, \end{cases}$$

and the proximal operator of $\alpha\tau|\cdot|$ is given by the soft-thresholding operator

$$\text{prox}_{\alpha\tau|\cdot|}(u_k) = \text{sgn}(u_k)(|u_k| - \alpha\tau)_+ =: S(u_k, \alpha\tau).$$

Defining $S(\mathbf{u}, \alpha\tau)$ as the component-wise soft-thresholding operator, that is, $[S(\mathbf{u}, \alpha\tau)]_k = S(u_k, \alpha\tau)$, we get

$$\begin{aligned}\text{prox}_{\tau h}(\mathbf{u}) &= \text{prox}_{(1-\alpha)\tau\|\cdot\|_q}(S(\mathbf{u}, \alpha\tau)) \\ &= \begin{cases} S(\mathbf{u}, \alpha\tau) - \text{Proj}_{B_1((1-\alpha)\tau)}(S(\mathbf{u}, \alpha\tau)), & q = \infty, \\ (\|S(\mathbf{u}, \alpha\tau)\|_2 - (1-\alpha)\tau)_+ \frac{S(\mathbf{u}, \alpha\tau)}{\|S(\mathbf{u}, \alpha\tau)\|_2}, & q = 2, \end{cases} \end{aligned} \tag{21}$$

the computation of which has been already studied in Section 3.3.

Remark. Although we could wish for a general algorithm for all $q \geq 1$, our construction is only valid for $q \in \{2, \infty\}$. As shown in Jenatton et al. (2010), the property used to derive the proximal operator of the composite penalty is only true when $q \in \{2, \infty\}$. Note also that the case $q = 1$ is simply the Lasso.

3.5. Missing Features Properties

One of the assumptions behind our algorithm is that all sources share exactly the same set of features. In practice, distinct sets of features may be encountered from different sources. For example, if a dataset contains policies from different years where some additional information is available in the later years, we may split the data into two sources where the first source contains fewer predictors than the second one. Another example is the case where data come from—literally—different sources that do not keep track of exactly the same information on the policy.

Suppose that the jth feature is missing from the kth source. We can set $X_j^{(k)} = \mathbf{0}$ for the corresponding j and k. It can be shown that this treatment, together with the initialization $\beta_j^{(k)} = 0$, keeps $\beta_j^{(k)}$ at zero throughout the entire algorithm for all choices of $q \in \{2, \infty\}$ and $0 \leq \alpha \leq 1$. This way, predictor j of source k is systematically excluded from the model.

Indeed, at any point of the algorithm, we have

$$\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})^{(k)} = (\tilde{\boldsymbol{\eta}}^{(k)} - \tilde{\mathbf{z}}^{(k)})^\top \widetilde{\mathbf{W}}^{(k)} X_j^{(k)} = 0.$$

Hence, in the proximal operator, we have $u_k = \tilde{\beta}_j^{(k)} - t_j \nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})^{(k)} = 0 - 0 = 0$. Then, the soft-thresholding operator produces

$$S(u_k, \alpha\tau) = \text{sgn}(u_k)(|u_k| - \alpha\tau)_+ = 0$$

for any $0 \leq \alpha \leq 1$. Thus, for $q = 2$, we get

$$[\text{prox}_{\tau h}(\mathbf{u})]_k = (||S(\mathbf{u}, \alpha\tau)||_2 - (1-\alpha)\tau)_+ \frac{S(u_k, \alpha\tau)}{||S(\mathbf{u}, \alpha\tau)||_2} = 0,$$

and, for $q = \infty$,

$$\begin{aligned}[\text{prox}_{\tau h}(\mathbf{u})]_k &= -\left[\text{Proj}_{B_1((1-\alpha)\tau)}(S(\mathbf{u}, \alpha\tau))\right]_k \\ &= -\text{sgn}(u_k)(|u_k| - \xi)_+ = 0.\end{aligned}$$

In any case, we obtain $\beta_j^{(k)+} = [\text{prox}_{\tau h}(\mathbf{u})]_k = 0$. It should be pointed out that this property does not prevent the same feature from being included in the model for other sources, though.

4. Implementation

4.1. Regularization Path

To select the tuning parameter, we apply the MStweedie-GPG algorithm on a decreasing sequence $(\lambda_l)_{l=1}^L$. The sequence of the corresponding solutions produces the solution path when a fine grid of λ is used. We present the solution path algorithm for solving MStweedie in Algorithm 3, where we wrap the MStweedie-GPG algorithm in an outer loop over the λ sequence. The sequence starts at $\lambda_1 = \lambda_{\max}$, chosen so that all coefficients except the intercepts are shrunken to zero, and iterates successively to smaller values of λ until the last value, λ_L, is reached.

The full sequence of λ is chosen as follows. We first compute $\lambda_{\max}$ via the KKT conditions (see below for details) and set $\lambda_{\min} = \varepsilon\lambda_{\max}$ for some small ε (e.g., $\varepsilon = 10^{-3}$). Then, we construct a logarithmically decreasing sequence from $\lambda_{\max}$ to $\lambda_{\min}$, that is, $\lambda_l = \lambda_{\max}(\lambda_{\min}/\lambda_{\max})^{\frac{l-1}{L-1}}$, where $l = 1, \ldots, L$. Note that we want $\tilde{\boldsymbol{\beta}}_j = \mathbf{0}$ for all $j \neq 0$ when $\lambda = \lambda_{\max}$. From the KKT conditions, that requires $\lambda \geq \nu_j^{-1}\|\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})\|_1$ for all $j \neq 0$. Therefore, we can choose $\lambda_{\max} = \max_{1 \leq j \leq p} \nu_j^{-1}\|\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})\|_1$. Now at $\lambda = \lambda_{\max}$, we have $\tilde{\boldsymbol{\beta}}(\text{init}) = \mathbf{0}$ and

$$\begin{aligned}\tilde{\beta}_0^{(k)}(\text{init}) &= \arg\min_{\beta_0^{(k)}} \sum_{i=1}^{n_k} w_i^{(k)} \left\{ -y_i^{(k)} \frac{e^{(1-\rho)\beta_0^{(k)}}}{1-\rho} + \frac{e^{(2-\rho)\beta_0^{(k)}}}{2-\rho} \right\}, \\ &= \log \frac{\sum_{i=1}^{n_k} w_i^{(k)} y_i^{(k)}}{\sum_{i=1}^{n_k} w_i^{(k)}}, \qquad k = 1, \ldots, K. \end{aligned} \tag{22}$$

Consequently, we obtain $\tilde{\eta}_i^{(k)} = \tilde{\beta}_0^{(k)}(\text{init})$ and

$$\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})^{(k)} = \sum_{i=1}^{n_k} \tilde{w}_i^{(k)} (\tilde{\eta}_i^{(k)} - \tilde{z}_i^{(k)}) x_{ij}^{(k)}, \tag{23}$$

which now can be used to determine $\lambda_{\max}$.

Algorithm 3: Solution path algorithm for solving MStweedie

1. Initialize $\tilde{\boldsymbol{\beta}}_j = \mathbf{0}$ and $\tilde{\boldsymbol{\beta}}_0 = \tilde{\boldsymbol{\beta}}_0(\text{init})$ according to (22).
2. Compute $\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})$ using (23) and set $\lambda_{\max} = \max_{1 \leq j \leq p} \nu_j^{-1}\|\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})\|_1$ and $\lambda = \lambda_{\max}$.
3. For $l = 2, \ldots, L$, do
   (a)Increment $\lambda \leftarrow \lambda \left(\frac{\lambda_{\min}}{\lambda_{\max}}\right)^{\frac{1}{L-1}}$,
   (b)Update $\tilde{\boldsymbol{\beta}}$ using Algorithm 1.

Once a solution path $\left\{(\tilde{\boldsymbol{\beta}}_0^{[l]}, \tilde{\boldsymbol{\beta}}^{[l]})\right\}_{l=1}^L$ is obtained, we could use cross-validation (CV) to perform the model selection, where the out-of-sample prediction deviance may be used as the guided criterion. The scaled deviance from a single observation is

$$\begin{aligned} d_i^{(k)} &= -2\phi\left\{\log f_Y(y_i^{(k)}|\mu_i^{(k)}, \phi, \rho) - \log f_Y(y_i^{(k)}|y_i^{(k)}, \phi, \rho)\right\} \\ &= 2\left\{ \frac{y_i^{(k)(2-\rho)} - y_i^{(k)}\mu_i^{(k)(1-\rho)}}{1-\rho} - \frac{y_i^{(k)(2-\rho)} - \mu_i^{(k)(2-\rho)}}{2-\rho} \right\}, \end{aligned}$$

where $\mu_i^{(k)} = \exp(\tilde{\beta}_0^{(k)} + \mathbf{x}_i^{(k)\top}\tilde{\boldsymbol{\beta}}^{(k)})$, and the full deviance is then the weighted sum across all observations from all sources. Often, we choose the optimal λ as the one that minimizes the CV deviance (call it λ_m). If model simplicity and interpretability are more of a concern, one may prefer the one-standard-error rule (Hastie, Tibshirani, and Friedman 2009), that is, choose optimal λ as the largest λ_l within one standard error of λ_m.

4.2. Further Acceleration and Stabilization Strategies

Two tricks suggested by Friedman, Hastie, and Tibshirani (2010) are added to our algorithm. First, the solution path is computed using warm starts at each iteration to increase the stability of the algorithm. This means that the initialization at $\lambda = \lambda_l$ is chosen to be the solution $\tilde{\mathbf{b}}^{[l-1]} = (\tilde{\boldsymbol{\beta}}_0^{[l-1]}, \tilde{\boldsymbol{\beta}}^{[l-1]})$ from previously $\lambda = \lambda_{l-1}$. Second, the MStweedie-GPG algorithm is augmented with the active set updates: we first run a full cycle of the updates and identify the set of active predictors $A = \{j \in \{1, \ldots, p\} | \tilde{\boldsymbol{\beta}}_j \neq 0\}$, and then repeat the cycles only over $j \in A$ until convergence.

Another method to speed up the calculations, similar to the active set updates, is the sequential strong rule (Tibshirani et al. 2012). Specifically, it is designed to identify an active set on which to perform the full MStweedie-GPG algorithm at each λ. Before entering the algorithm at λ_l, we check the following conditions for each $j = 1, \ldots, p$

$$\left\| \nabla_j \ell(\tilde{\boldsymbol{\beta}}_j^{[l-1]}; \tilde{\mathbf{b}}_{-j}^{[l-1]}) \right\|_1 < \nu_j(2\lambda_l - \lambda_{l-1}).$$

We exclude every predictor with index j that meets the above condition and run the MStweedie-GPG algorithm on the remaining predictors. Once the algorithm reaches convergence with these remaining variables, we perform a final check to verify that we do not accidentally exclude a predictor that should have been included. The check is based on the KKT conditions: for each predictor j initially excluded, we verify the KKT condition with $\boldsymbol{\beta}_j = 0$, which requires $\|\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})\|_{q^*} \leq \lambda \nu_j$, where $q^* = 1$ if $q = \infty$ and $q^* = 2$ if $q = 2$. If at least one condition is violated, then the corresponding predictor is added back to the active set. This process is repeated until the KKT condition is satisfied for all excluded predictors.

The algorithm with the sequential strong rule is presented in Algorithm 4.

Algorithm 4: MStweedie sequential strong rule.

1. Do while $V \neq \emptyset$:
   (a) Identify $S = \{j : \|\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})\|_{q^*} \geq \nu_j(2\lambda_l - \lambda_{l-1})\}$ and $S^C = \{1, \ldots, p\} \setminus S$.
   (b) Update $\tilde{\boldsymbol{\beta}}$ as in Algorithm 1 while keeping $\tilde{\boldsymbol{\beta}}_j = 0$ for all $j \in S^C$.
   (c) Identify the violations $V = \{j : \|\nabla_j \ell(\tilde{\boldsymbol{\beta}}_j; \tilde{\mathbf{b}}_{-j})\|_{q^*} \leq \lambda \nu_j,\ j \in S^C\}$.

Note: $q^* = 1$ if $q = \infty$ and $q^* = 2$ if $q = 2$.

4.3. Adaptive MStweedie

We also consider an adaptive version of MStweedie (a-MStweedie). The a-MStweedie is motivated from Zou (2006), where the adaptive lasso is used to improve model selection performance over the regular lasso. In a-MStweedie, we first obtain $\widehat{\boldsymbol{\beta}}^*$, the cross-validated parameter estimate under equal penalty factors (i.e., $\nu_j = 1$ for all $1 \leq j \leq p$). Then, we update the penalty factors $\nu_j = ||\widehat{\boldsymbol{\beta}}_j^*||_q^{-\varphi}$ for some $\varphi > 0$ (default is $\varphi = 1$) and refit the model with these new penalty factors. When the initial CV yields $\widehat{\boldsymbol{\beta}}_j^* = \mathbf{0}$ for some j, we set ν_j to a large machine number to ensure that this variable is not included in the adaptive modeling.

5. Numerical Studies

5.1. Performance Assessment

We use deviance as the criterion to access model fit. First of all, we split the data into two parts: a training set on which a model is fit to yield the coefficient estimates, and a testing set on which these estimates are used for prediction. The train and test deviances are then obtained, respectively, from these two sets.

Three measures are considered for assessing selection performance: the percentage of variables correctly identified (*accuracy*), the percentage of identified variables that are indeed true variables (*precision*), and the percentage of true variables identified (*recall*). These three measures describe different aspects of a variable selection result and are widely used in classification and pattern recognition (see, e.g., Fawcett 2006). In terms of overall performance, accuracy is perhaps a more interesting measure as our goal is not only to find the true predictors but also to exclude those spurious ones.

5.2. Synthetic Data

We consider a variety of settings under which our algorithm is tested and compared to existing ones.

5.2.1. Setting 1—Unequal Coefficients, $p < n_k$

This simulation setting is inspired by Gong, Ye, and Zhang (2012), in which we set the number of sources to $K = 10$, the number of observations to $n_k = 400$, $k = 1, \ldots, 10$, and the number of covariates to $p = 100$. The covariates are generated from independent normal distributions. Moreover, we set the coefficient matrix $\boldsymbol{\beta}$ to zero everywhere except the last 10 columns, which are generated from independent normal distributions of mean 0 and variance $4^2\sigma$ with $\sigma = 0.1$. Finally, we generate the responses $y_i^{(k)}$ from $\text{Tw}(\mu_i^{(k)}, \phi, \rho)$ with $\phi = 1$ and $\rho = 1.5$, where $\mu_i^{(k)} = \exp(\mathbf{x}_i^{(k)\top}\boldsymbol{\beta}^{(k)})$ for all i and k.

We randomly split the above data into two equal parts ($n_k = 200$ for each source): the first part is used to tune the model via 10-fold CV, while the second is used for testing the model. The results are averaged over 100 replications. The following models are compared: Full Lasso (L_1-regularized Tweedie model on the full dataset, using the HDtweedie package by Qian, Yang, and Zou (2016)), Individual Lasso (Individual L_1-regularized Tweedie model for each source, also using HDtweedie), and

Table 1. Results from Setting 1 with 100 replications.

(a) Setting 1: Mean (standard error)

	Full Lasso		Ind. Lasso		L_1/L_∞		a-L_1/L_∞		L_1/L_2		a-L_1/L_2	
Test dev.	2,642,427 (553,415)		17,710 (3234)		17,330 (2066)		5963 (523)		7763 (717)		5590 (723)	
Size	1.02	(0.36)	89.37	(0.43)	89.46	(0.89)	10.00	(0.00)	40.19	(1.34)	10.00	(0.00)
Accuracy	89.9	(0.2)	20.6	(0.4)	20.5	(0.9)	100.0	(0.0)	69.8	(1.3)	100.0	(0.0)
Precision	94.0	(1.9)	11.2	(0.1)	11.3	(0.1)	100.0	(0.0)	28.3	(1.1)	100.0	(0.0)
Recall	4.6	(1.2)	100.0	(0.0)	100.0	(0.0)	100.0	(0.0)	100.0	(0.0)	100.0	(0.0)
L_2 loss	12.76	(0.01)	1.56	(0.04)	1.87	(0.06)	1.20	(0.06)	1.30	(0.06)	1.03	(0.07)

(b) Setting 1: Mean rank (# of times best)

	Full Lasso		Ind. Lasso		L_1/L_∞		a-L_1/L_∞		L_1/L_2		a-L_1/L_2	
Test dev.	6.00	(0)	4.00	(5)	4.74	(0)	2.08	(16)	2.89	(2)	1.29	(77)
Size	1.06	(97)	5.36	(0)	5.58	(0)	1.97	(3)	4.00	(0)	1.97	(3)
Accuracy	3.04	(0)	5.36	(0)	5.58	(0)	1.00	(100)	3.93	(0)	1.00	(100)
Precision	1.27	(89)	5.35	(0)	5.57	(0)	1.00	(100)	3.97	(0)	1.00	(100)
Recall	6.00	(0)	1.00	(100)	1.00	(100)	1.00	(100)	1.00	(100)	1.00	(100)
L_2 loss	6.00	(0)	3.74	(9)	4.86	(0)	2.29	(9)	2.77	(1)	1.34	(81)

NOTE: Part (a) shows the mean values of the statistics (with their standard errors listed in the parentheses). Part (b) shows the mean rank across the six models and, in parentheses, the number of times the model is best.

MStweedie with L_1/L_∞, L_1/L_2, a-L_1/L_∞ (adaptive L_1/L_∞), and a-L_1/L_2 (adaptive L_1/L_2) regularizations.

Part (a) of Table 1 lists the averages and standard errors of different statistics. The test deviance, measuring the goodness of fit of the corresponding model, shows that MStweedie with the adaptive L_1/L_2 regularization is the best while the Full Lasso performs very poorly on this matter. The poor performance of Full Lasso is due to the fact that it has identical estimates across sources, which is apparently not true according to our data-generating mechanism.

If we disregard the Full Lasso (which selected no features 81 out of 100 times), the two adaptive procedures performed the best in terms of variable selection performance, where each picks exactly 10 predictors in every replication. This selection matches exactly the true active variables so that both a-L_1/L_2 and a-L_1/L_∞ achieve perfect accuracy, precision and recall. For the other models, the number of selected variables is much larger, yielding low precision and accuracy even with perfect recall. Finally, a-L_1/L_2 produces estimates that are closest (in L_2 norm) to the true coefficients. For both L_1/L_∞ and L_1/L_2, their adaptive versions greatly increase the selection accuracy and precision by picking much fewer variables while achieving lower deviance and L_2-loss. Overall, L_1/L_∞ exhibits similar performance to Individual Lasso, but its adaptive version increases the performance significantly.

The results about the ranking of these methods, reported in part (b) of Table 1, gives similar conclusions. However, we can see that, although being the best on average, a-L_1/L_2 is occasionally outperformed by either Individual Lasso or a-L_1/L_∞ in terms of test deviance.

5.2.2. Setting 2—Equal Coefficients, $p > n_k$

In this setting, we consider the high-dimensional scenario ($p > n_k$) with local correlation structure. The data are generated similarly as in Gu and Zou (2016) with $n_k = 300$, $p = 600$, and $K = 5$. We generate the covariates $\mathbf{x}_i^{(k)}$ from the multivariate normal distribution with mean $\mathbf{0}$ and covariance matrix $\mathbf{\Sigma} = (0.5^{|i-j|})_{i,j=1}^{p}$ and set $\boldsymbol{\beta}_j = 0$ for all j except $j \in \{2, 4, 8, 16, 32\}$ where it is set to 2 in all sources. We then simulate the responses as in Setting 1 using $\mu_i^{(k)} = \exp(\mathbf{x}_i^{(k)\top}\boldsymbol{\beta}^{(k)})$, $\phi = 1$, and $\rho = 1.5$. Results below are summarized from 100 replications.

Part (a) of Table 2 contains the average values and standard errors of the different statistics. The lowest average test deviance is achieved by a-L_1/L_∞ followed closely by Full Lasso while Individual Lasso is significantly worse. The models selected by L_1/L_∞ are much more complex than any other method. As in Setting 1, the two adaptive methods performed perfectly in terms of accuracy, precision and recall, since they select the five true predictors exactly. Also, a-L_1/L_∞ produces the best estimates in term of L_2-loss. The study of the rankings, in part (b) of Table 2, leads to the same observations except that a-L_1/L_2 and Full Lasso outperform a-L_1/L_∞ on some occasions in terms of test deviance or L_2-loss.

5.2.3. Setting 3—Within-Feature Sparsity

In multisource insurance claim data, some predictors may not be relevant to all sources. For example, property age may only help predict the property claim amount. Some information of the same policyholders, such as credit history, however, may be relevant for both sources. The model thus exhibits both *within-feature* and *between-sources* sparsity. We consider a scenario designed to generate such a model to specifically test our $L_1/L_q(\alpha)$ regularization.

The setting is similar to Setting 2, except that we voluntarily set the coefficients of some true generating variables to zero in certain sources

$$(\boldsymbol{\beta}_2, \boldsymbol{\beta}_4, \boldsymbol{\beta}_8, \boldsymbol{\beta}_{16}, \boldsymbol{\beta}_{32}) = \begin{bmatrix} 2 & 0 & 2 & 2 & 0 \\ 0 & 2 & 2 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 2 \\ 0 & 2 & 0 & 2 & 2 \end{bmatrix},$$

$$\boldsymbol{\beta}_j = \mathbf{0}, j \notin \{2, 4, 8, 16, 32\}.$$

Table 2. Results from Setting 2 with 100 replications.

(a) Setting 2: Mean (standard error)

	Full Lasso		Ind. Lasso		L_1/L_∞		a-L_1/L_∞		L_1/L_2		a-L_1/L_2	
Test dev.	1430 (142)		1,136,266 (265,655)		3096 (604)		1161 (129)		6968 (1277)		2572 (642)	
Size	22.07	(0.76)	29.73	(1.44)	71.10	(2.33)	5.00	(0.00)	37.86	(0.94)	5.00	(0.00)
Accuracy	97.2	(0.1)	95.7	(0.2)	89.0	(0.4)	100.0	(0.0)	94.5	(0.2)	100.0	(0.0)
Precision	24.5	(0.6)	22.6	(2.0)	8.1	(0.4)	100.0	(0.0)	14.2	(0.4)	100.0	(0.0)
Recall	100.0	(0.0)	91.2	(2.5)	100.0	(0.0)	100.0	(0.0)	100.0	(0.0)	100.0	(0.0)
L_2 loss	0.43	(0.03)	8.86	(0.07)	0.70	(0.02)	0.34	(0.01)	1.03	(0.06)	0.53	(0.05)

(b) Setting 2: Mean rank (# of times best)

	Full Lasso		Ind. Lasso		L_1/L_∞		a-L_1/L_∞		L_1/L_2		a-L_1/L_2	
Test dev.	2.09	(30)	6.00	(0)	3.89	(0)	1.57	(55)	4.90	(0)	2.55	(15)
Size	3.34	(0)	3.87	(7)	5.97	(0)	1.06	(94)	4.67	(0)	1.06	(94)
Accuracy	3.34	(0)	4.01	(0)	5.97	(0)	1.00	(100)	4.67	(0)	1.00	(100)
Precision	3.32	(0)	3.94	(5)	5.96	(0)	1.00	(100)	4.66	(0)	1.00	(100)
Recall	1.00	(100)	1.70	(86)	1.00	(100)	1.00	(100)	1.00	(100)	1.00	(100)
L_2 loss	2.16	(25)	6.00	(0)	3.84	(0)	1.44	(61)	4.97	(0)	2.59	(14)

NOTE: Part (a) shows the mean values of the statistics and their standard errors in parentheses. Part (b) shows the mean rank across the six models and, in parentheses, the number of times the model is best.

Thus, the true model is sparse in terms of features (only five generating variables), but it is also sparse within features since some of the true features do not generate the responses in certain sources.

Under Setting 2, we see that a-L_1/L_∞ produces the best fit. In this setting, we compare Full Lasso (which should not perform well) and Individual Lasso to a-$L_1/L_\infty(\alpha)$ for different choices of the mixing parameter α. The case $\alpha = 0$ is exactly a-L_1/L_∞ and we also consider $\alpha = 0.5$, $\alpha = 0.8$ and $\alpha = 1$. We note that there is a small difference between Individual Lasso and the case $\alpha = 1$: both models consider the same regularization, but the former selects a model through CV in each source, while the latter selects a model through CV for all sources simultaneously.

Under this setting, the statistics of size, accuracy, precision and recall are calculated for each component $\beta_j^{(k)}$ instead of the vectors $\boldsymbol{\beta}_j$. This means that the true model has size $3 + 3 + 2 + 2 + 2 = 12$. Using this definition, we will see more clearly the effect of α on the sparsity of the selected model. The results from 100 replications are summarized in Table 3.

The lowest test deviance is achieved by a-$L_1/L_\infty(\alpha)$ with $\alpha = 0.5$. It is not significantly better than other values of the parameter, but clearly has improvement over the Full and Individual Lasso for out-of-sample adjustment. We also observe a decrease in the size of the model as α increases: starting from 25 selected features with $\alpha = 0$ (i.e., five features selected in five sources since there is no selection performed across sources) to less than 15 for $\alpha = 1$, closing in to the 12 generating features. With perfect recall for all MStweedie algorithms, this means that with $\alpha = 1$ we achieve the best accuracy and precision. Finally, the L_2 loss being the smallest under $\alpha = 0.5$ means that its extra selected features have coefficient estimates very close to 0 and that its coefficient estimates for the true features are closer to the true values.

5.2.4. Setting 4—Different Datasets

To test how our algorithm behave under circumstances where some features are missing from certain sources, we consider three simulation setups: (4A) some true generating variables are missing from certain sources, (4B) some spurious variables are missing from certain sources, and (4C) both true and spurious variables are missing from certain sources. For all cases, we generate data as in Setting 2 with $K = 5$, $n_k = 300$, $p = 600$ and the true variable indices are $\{2, 4, 8, 16, 32\}$. In Setting 4A, we set to 0 column 32 for sources 1 and 2 and columns 16 and 32 of source 3. In Setting 4B, we set to 0 the last 100 columns of sources 1 and 2 and the last 200 columns of source 3. In Setting 4C, we consider the zero columns of Settings 4A and 4B simultaneously. For demonstration purposes, we compare Full Lasso, Individual Lasso and a-L_1/L_∞. The results over 100 replications are reported in Table 4.

Under Setting 4A, where true variables are omitted in some sources, we find that a-L_1/L_∞ clearly outperforms both Full Lasso and Individual Lasso under all criteria. As we would expect, it does not achieve the same performance as when using the complete dataset (Setting 2) due to removal of important features.

Under Setting 4B, where only spurious variables are removed from some sources, we do not observe significant difference in any statistic compared to the models trained on the complete data.

Under Setting 4C, where both true and spurious variables are removed from some sources, we observe similar behavior as in Setting 4A, with a-L_1/L_∞ having slightly better performance. It seems that both Full Lasso and a-L_1/L_∞ are less inclined to overfit the spurious information when it is missing from some sources.

5.2.5. Setting 5—Scalability Study

Under the same construction of Setting 1, we conduct a short scalability study of the influence of the number of covariates

Table 3. Results from Setting 3 with 100 replications.

(a) Setting 3: Mean (standard error)

					a-$L_1/L_\infty(\alpha)$							
	Full Lasso		Ind. Lasso		$\alpha=0$		$\alpha=0.5$		$\alpha=0.8$		$\alpha=1$	
Test dev.	52,398 (5830)		11,590 (1448)		861 (9)		852 (8)		863 (10)		866 (10)	
Size	61.20	(5.52)	30.63	(0.55)	25.00	(0.00)	16.97	(0.20)	15.34	(0.18)	14.76	(0.17)
Accuracy	98.1	(0.2)	99.3	(0.0)	99.6	(0.0)	99.8	(0.0)	99.9	(0.0)	99.9	(0.0)
Precision	27.3	(2.4)	38.0	(0.5)	48.0	(0.0)	71.7	(0.9)	79.4	(1.0)	82.4	(1.0)
Recall	72.2	(3.0)	95.1	(1.0)	100.0	(0.0)	100.0	(0.0)	100.0	(0.0)	100.0	(0.0)
L_2 loss	5.99	(0.05)	2.73	(0.09)	0.38	(0.01)	0.34	(0.01)	0.36	(0.01)	0.36	(0.01)

(b) Setting 3: Mean rank (# of times best)

					a-$L_1/L_\infty(\alpha)$							
	Full Lasso		Ind. Lasso		$\alpha=0$		$\alpha=0.5$		$\alpha=0.8$		$\alpha=1$	
Test dev.	5.96	(0)	5.04	(0)	2.68	(31)	2.19	(31)	2.47	(15)	2.66	(23)
Size	4.82	(19)	5.10	(0)	4.37	(0)	2.89	(8)	1.68	(43)	1.23	(78)
Accuracy	5.56	(0)	5.15	(0)	4.19	(0)	2.68	(8)	1.52	(49)	1.08	(92)
Precision	5.42	(7)	5.14	(0)	4.19	(0)	2.75	(8)	1.59	(48)	1.15	(86)
Recall	4.27	(34)	2.14	(74)	1.00	(100)	1.00	(100)	1.00	(100)	1.00	(100)
L_2 loss	6.00	(0)	5.00	(0)	2.87	(18)	2.00	(40)	2.41	(21)	2.72	(21)

NOTE: Part (a) shows the mean values of the statistics and their standard errors in parentheses. Part (b) shows the mean rank across the five models and, in parentheses, the number of times the model is best.

Table 4. Results from Setting 4 with 100 replications: values represent mean values of the statistics and their standard errors in parentheses.

Setting 4: Mean (standard error)

(2) Complete data

	Full Lasso		Ind. Lasso		a-L_1/L_∞	
Test dev.	1430 (142)		1,136,266 (265,655)		1161 (129)	
Size	22.07	(0.76)	29.73	(1.44)	5.00	(0.00)
Accuracy	97.2	(0.1)	95.7	(0.2)	100.0	(0.0)
Precision	24.5	(0.6)	22.6	(2.0)	100.0	(0.0)
Recall	100.0	(0.0)	91.2	(2.5)	100.0	(0.0)
L_2 loss	0.43	(0.03)	8.86	(0.07)	0.34	(0.01)

(4A) Missing true features

	Full Lasso		Ind. Lasso		a-L_1/L_∞	
Test dev.	272,840 (49,327)		2,122,348 (839,704)		152,315 (31,194)	
Size	34.74	(2.83)	21.38	(1.33)	11.95	(0.44)
Accuracy	94.9	(0.5)	96.9	(0.2)	98.7	(0.1)
Precision	19.2	(1.3)	31.5	(2.7)	44.8	(1.8)
Recall	89.4	(1.4)	78.8	(3.5)	92.8	(1.5)
L_2 loss	6.01	(0.13)	9.34	(0.04)	5.15	(0.08)

(4B) Missing spurious features

	Full Lasso		Ind. Lasso		a-L_1/L_∞	
Test dev.	1376 (133)		1,119,800 (265,508)		1164 (130)	
Size	19.53	(0.53)	29.77	(1.33)	5.00	(0.00)
Accuracy	97.6	(0.1)	95.8	(0.2)	100.0	(0.0)
Precision	27.4	(0.7)	22.1	(1.8)	100.0	(0.0)
Recall	100.0	(0.0)	94.4	(2.0)	100.0	(0.0)
L_2 loss	0.41	(0.03)	8.79	(0.07)	0.34	(0.01)

(4C) Missing true and spurious features

	Full Lasso		Ind. Lasso		a-L_1/L_∞	
Test dev.	245,761 (46,198)		2,116,184 (839,728)		119,081 (18,616)	
Size	31.25	(2.53)	21.49	(1.25)	11.60	(0.42)
Accuracy	95.5	(0.4)	96.9	(0.2)	98.8	(0.1)
Precision	21.1	(1.5)	29.9	(2.5)	46.4	(1.7)
Recall	90.4	(1.3)	80.8	(3.2)	94.6	(1.5)
L_2 loss	5.99	(0.12)	9.33	(0.04)	5.09	(0.08)

NOTE: The four parts, respectively, show the results from Settings 2, 4A, 4B, and 4C for comparison.

Table 5. Description of the different parameters used in Setting 5.

Setting 5: Description of the scenarios

	K	p	# of true variables	% true variables	n_k
(a)	20	$10 \times 3^i, i = 0, \ldots, 5$	10	–	300
(b)	20	$10 \times 3^i, i = 0, \ldots, 5$	–	10%	300
(c)	$5 \times 2^i, i = 0, \ldots, 5$	100	10	–	300
(d)	20	100	$2^i, i = 0, \ldots, 5$	–	300
(e)	5	50	10	–	$5 \times 4^i, i = 0, \ldots, 5$
(f)	5	1000	10	–	$30 \times 2^i, i = 0, \ldots, 5$

p, the number of sources K and sample sizes n_k on the CPU time. We consider different scenarios as shown in Table 5. The running times are averaged over 10 independent runs and are used to compare the L_1/L_∞ and L_1/L_2 regularizations to the Individual Lasso.

Figure 1 contains the plot of the average CPU time versus the variable of interest under the four schemes considered. In parts (a) and (b), the running times of all three algorithms increase at a similar linear rate. In part (c), we clearly see, as we would expect, that the running time of individual regularization increases linearly with the number of sources. In contrast, the CPU times of the two MStweedie algorithms increase faster than the linear rate and seem to diminish with K. Note that the iteration complexity of the MStweedie algorithm is influenced by K mainly in the step that requires Euclidean projections. For L_1/L_∞ regularization, Condat (2016) pointed out that the

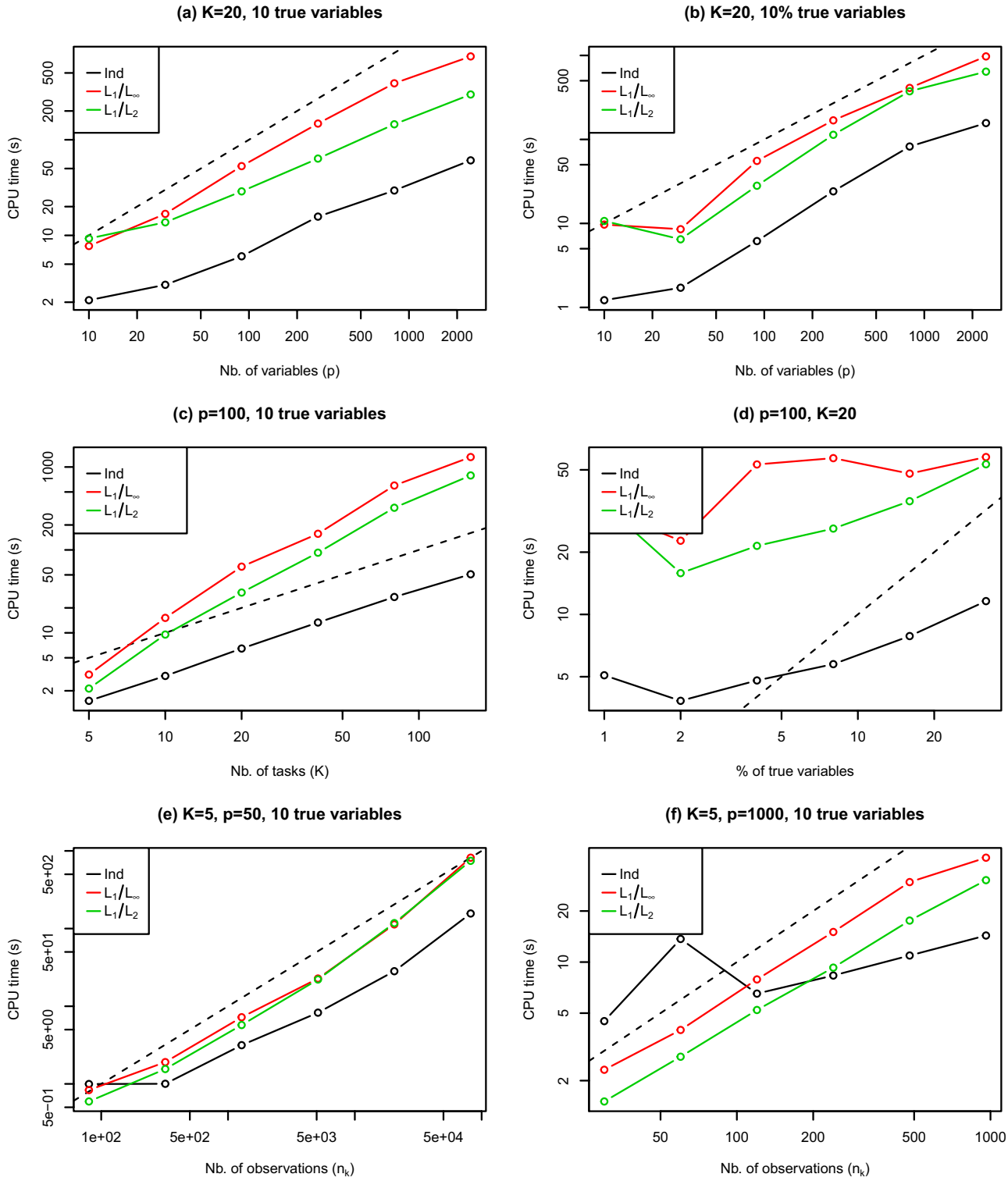

Figure 1. Results from the scalability study under various conditions for synthetic data. The dashed line represents what a linear relation between the CPU time and the variable of interest would follow. All axes are in logarithmic scales.

algorithm by Duchi et al. (2008) has expected and observed complexity $\mathcal{O}(K)$, but can be slower (up to $\mathcal{O}(K^2)$) in sparse problems.

In part (d), we study the effect of sparsity by varying the proportion of true variables in the model. For all three algorithms, we note a slight increase of the computing time when the proportion increases. In parts (e) and (f), we look at the effect of the sample size n_k in the cases $n_k > p$ and $n_k < p$, respectively. A linear rate can be observed for both cases with the MStweedie algorithms. In contrast, the Individual Lasso has CPU time increasing only sublinearly when $n_k < p$.

Overall, L_1/L_∞ regularization is systematically slower than L_1/L_2 regularization by a multiplicative constant. Both MStweedie algorithms are slower than individual regularization only by a multiplicative constant.

5.2.6. Setting 6—Correlated Responses in

Upon the request of a referee, we also study the impact of having correlated responses on the performance of our proposed algorithm. The simulation results show that all versions of our algorithm significantly produce better test deviance and the two adaptive versions clearly beats all other models. Due to limited space, we provide the simulation results in Appendix E (supplementary materials).

5.3. Real Data—Automobile Insurance Claims

We apply our algorithm to the analysis of a real dataset studied in Yip and Yau (2005) and Qian, Yang, and Zou (2016). The dataset consists of many automobile insurance policy records and is available as `AutoClaim` in the R package `cplm` (Zhang 2011, 2013). A preprocessed version of the data is also available in our R package. It contains the records of 10,296 policies of which 6290 (61.1%) have no claims. We are interested in predicting the aggregate claim loss of the policy using the 15 predictors (along with their necessary transformations) described in Table 6. We split the dataset into two sources corresponding to potentially different types of driving license (according to whether or not the policyholder had his or her license revoked). Source 1 contains 9036 policies of which 5643 (62.5%) have no insurance claims and Source 2 contains 1260 policies of which 646 (51.3%) have no insurance claims. Figure 2 plots the histogram of the aggregate claims for both sources.

The following models are considered: the Full and Individual Lasso, and the MStweedie with both $L_1/L_2(\alpha)$ and $L_1/L_\infty(\alpha)$ regularizations as well as their adaptive counterparts under different values of the mixing parameter $\alpha \in \{0, 0.5, 0.8, 1\}$. We split the dataset into a training and a testing set consisting, respectively, of two-thirds and one-third of the policies of each source. The 10-fold CV is then performed to select the best model. Finally, we summarize the results by averaging them over 100 replications of training/testing random partition.

The results of the study are reported in Table 7. In terms of model fit, we note that all adaptive MStweedie methods perform very similarly while the nonadaptive procedures and the Individual Lasso are slightly worse and the Full Lasso is the worst. In terms of model sparsity, the Individual Lasso produces the simplest models on average followed by the adaptive MStweedie algorithms and then the Full Lasso. The nonadaptive MStweedie algorithms yield models that have significantly more variables.

Now, by looking at the exact variables selected within each source, we first see that `MVR_PTS` and `AREA` are systematically included in every model except the Individual Lasso which does not include `AREA` for source 2. When α is nonzero, there is no major difference between the models under different values of

Table 6. Description of the variables in the auto insurance claim dataset.

AutoClaim Dataset Variable Description			
Variable	Type	Transformation	Description
Response			
`CLM_AMT5`	Numerical	$\times 10^{-3}$	Aggregate claim loss of policy
Source identifier			
`REVOKED`	Categorical(2)	1/2	Whether the policyholder's license was (2) revoked in the past or (1) not
Predictors			
`KIDSDRIV`	Numerical	–	Number of child passengers
`TRAVTIME`	Numerical	–	Commute time
`CAR_USE`	Categorical(2)	1/2	(1) Private or (2) Commercial use
`BLUEBOOK`	Numerical	log	Car value
`NPOLICY`	Numerical	–	Number of policies
`RED_CAR`	Categorical(2)	1/2	Whether the color of the car is (2) red or (1) not
`MVR_PTS`	Numerical	–	Number of motor vehicle record points
`AGE`	Numerical	–	Age of policyholder
`HOMEKIDS`	Numerical	–	Number of children at home
`GENDER`	Categorical(2)	1/2	Gender of policyholder: (2) male or (1) female
`PARENT1`	Categorical(2)	1/2	Whether (2) the policyholder grew up in a single-parent family or (1) not
`AREA`	Categorical(2)	1/2	(1) Rural or (2) urban area
`CAR_TYPE`	Categorical(6)	Dummy(5)	Type of car: (base) Panel Truck, (2) Pickup, (3) Sedan, (4) Sports Car, (5) SUV, (6) Van
`JOBCLASS`	Categorical(9)	Dummy(8)	Job class of policyholder: (base) Unknown, (2) Blue Collar, (3) Clerical, (4) Doctor, (5) Home Maker, (6) Lawyer, (7) Manager, (8) Professional, (9) Student
`MAX_EDUC`	Categorical(5)	Dummy(4)	Maximal level of education of policyholder: (base) less than High School, (2) Bachelors, (3) High School, (4) Masters, (5) PhD

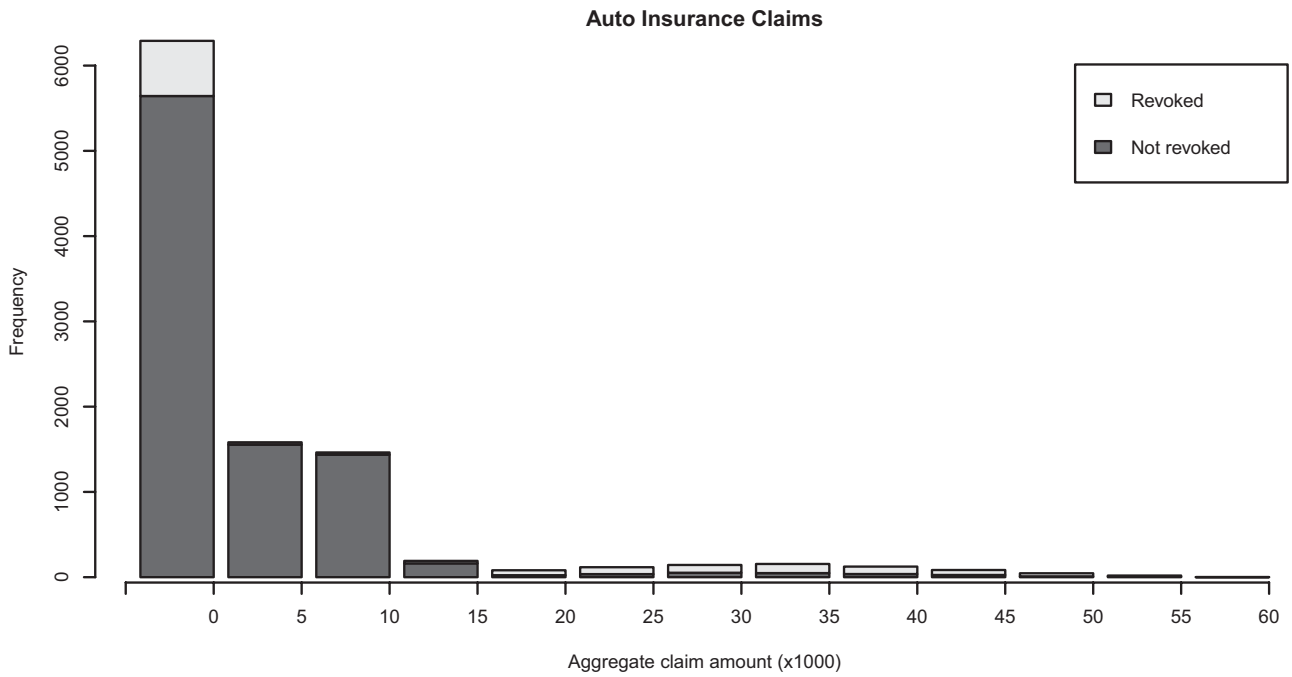

Figure 2. Frequency of the aggregate claim amounts in the AutoClaim dataset according the whether or not the policyholder's license was revoked (defining the two sources).

Table 7. Test deviance, size of the selected model and selected variables under different regularization schemes on the AutoClaim dataset.

Auto Claims: Mean (standard error)					
Algorithm	Test Deviance		Size		Selected variables (# of times in source 1, in source 2)
Full Lasso	22,203	(35)	5.32	(0.30)	MVR_PTS(100,100), AREA(100,100), CAR_TYPE_4(25,25), MARRIED(14,14), PARENT1(6,6), KIDSDRIV(4,4), CAR_TYPE_3(4,4), JOBCLASS_6(3,3), MAX_EDUC_3(3,3), BLUEBOOK(2,2), JOBCLASS_3(2,2), JOBCLASS_4(2,2), MAX_EDUC_5(1,1)
Ind. Lasso	19,493	(33)	3.77	(0.11)	MVR_PTS(100,100), AREA(100,36), CAR_TYPE_4(0,27), CAR_USE(0,1), MARRIED(2,1), JOBCLASS_3(0,1), JOBCLASS_6(1,1), MAX_EDUC_4(0,1), AGE_CAT_5(0,1)
L_1/L_∞	19,475	(32)	13.08	(0.63)	MVR_PTS(100,100), AREA(100,100), CAR_TYPE_4(98,98), JOBCLASS_3(59,59), JOBCLASS_6(59,59), CAR_TYPE_5(33,33), MARRIED(25,25), JOBCLASS_7(22,22), KIDSDRIV(21,21), AGE_CAT_5(20,20), AGE_CAT_2(16,16), JOBCLASS_5(15,15), CAR_USE(12,12), BLUEBOOK(10,10), CAR_TYPE_6(10,10), MAX_EDUC_4(10,10), JOBCLASS_4(8,8), JOBCLASS_8(7,7), RED_CAR(4,4), TRAVTIME(3,3), CAR_TYPE_2(3,3), CAR_TYPE_3(3,3), MAX_EDUC_2(3,3), AGE_CAT_4(3,3), PARENT1(2,2), MAX_EDUC_3(2,2), AGE_CAT_3(2,2), NPOLICY(1,1), GENDER(1,1), JOBCLASS_2(1,1), MAX_EDUC_5(1,1)
a-$L_1/L_\infty(0)$	19,438	(32)	5.00	(0.12)	MVR_PTS(100,100), AREA(100,100), CAR_TYPE_4(45,45), JOBCLASS_6(4,4), MARRIED(1,1)
a-$L_1/L_\infty(0.5)$	19,437	(31)	4.31	(0.05)	MVR_PTS(100,100), AREA(100,100), CAR_TYPE_4(1,28), MARRIED(0,1), JOBCLASS_6(0,1)
a-$L_1/L_\infty(0.8)$	19,431	(32)	4.29	(0.05)	MVR_PTS(100,100), AREA(100,100), CAR_TYPE_4(1,26), JOBCLASS_6(0,2)
a-$L_1/L_\infty(1)$	19,431	(32)	4.29	(0.05)	MVR_PTS(100,100), AREA(100,100), CAR_TYPE_4(0,28), JOBCLASS_6(0,1)
L_1/L_2	19,456	(30)	9.86	(0.29)	MVR_PTS(100,100), AREA(100,100), CAR_TYPE_4(99,99), JOBCLASS_3(50,50), JOBCLASS_6(44,44), CAR_TYPE_5(16,16), JOBCLASS_7(16,16), JOBCLASS_5(12,12), AGE_CAT_5(11,11), AGE_CAT_2(9,9), MAX_EDUC_4(8,8), CAR_USE(6,6), MARRIED(6,6), BLUEBOOK(5,5), KIDSDRIV(2,2), RED_CAR(2,2), GENDER(1,1), CAR_TYPE_3(1,1), CAR_TYPE_6(1,1), JOBCLASS_4(1,1), JOBCLASS_8(1,1), MAX_EDUC_2(1,1), AGE_CAT_4(1,1)
a-$L_1/L_2(0)$	19,434	(31)	5.00	(0.11)	MVR_PTS(100,100), AREA(100,100), CAR_TYPE_4(48,48), MARRIED(1,1), JOBCLASS_6(1,1)

(continued)

Table 7. Continued

Auto Claims: Mean (standard error)					
Algorithm	Test Deviance		Size		Selected variables (# of times in source 1, in source 2)
a-$L_1/L_2(0.5)$	19,442	(32)	4.61	(0.05)	MVR_PTS(100,100), AREA(100,100), CAR_TYPE_4(1,56), JOBCLASS_6(0,2), MAX_EDUC_4(0,1), AGE_CAT_5(0,1)
a-$L_1/L_2(0.8)$	19,432	(31)	4.68	(0.05)	MVR_PTS(100,100), AREA(100,100), CAR_TYPE_4(0,62), JOBCLASS_6(0,4), MAX_EDUC_4(0,1), AGE_CAT_5(0,1)
a-$L_1/L_2(1)$	19,428	(31)	4.72	(0.05)	MVR_PTS(100,100), AREA(100,100), CAR_TYPE_4(0,66), JOBCLASS_6(0,3), MARRIED(0,1), MAX_EDUC_4(0,1), AGE_CAT_5(0,1)

NOTE: The results are averaged over 100 replications of the training/testing splitting.

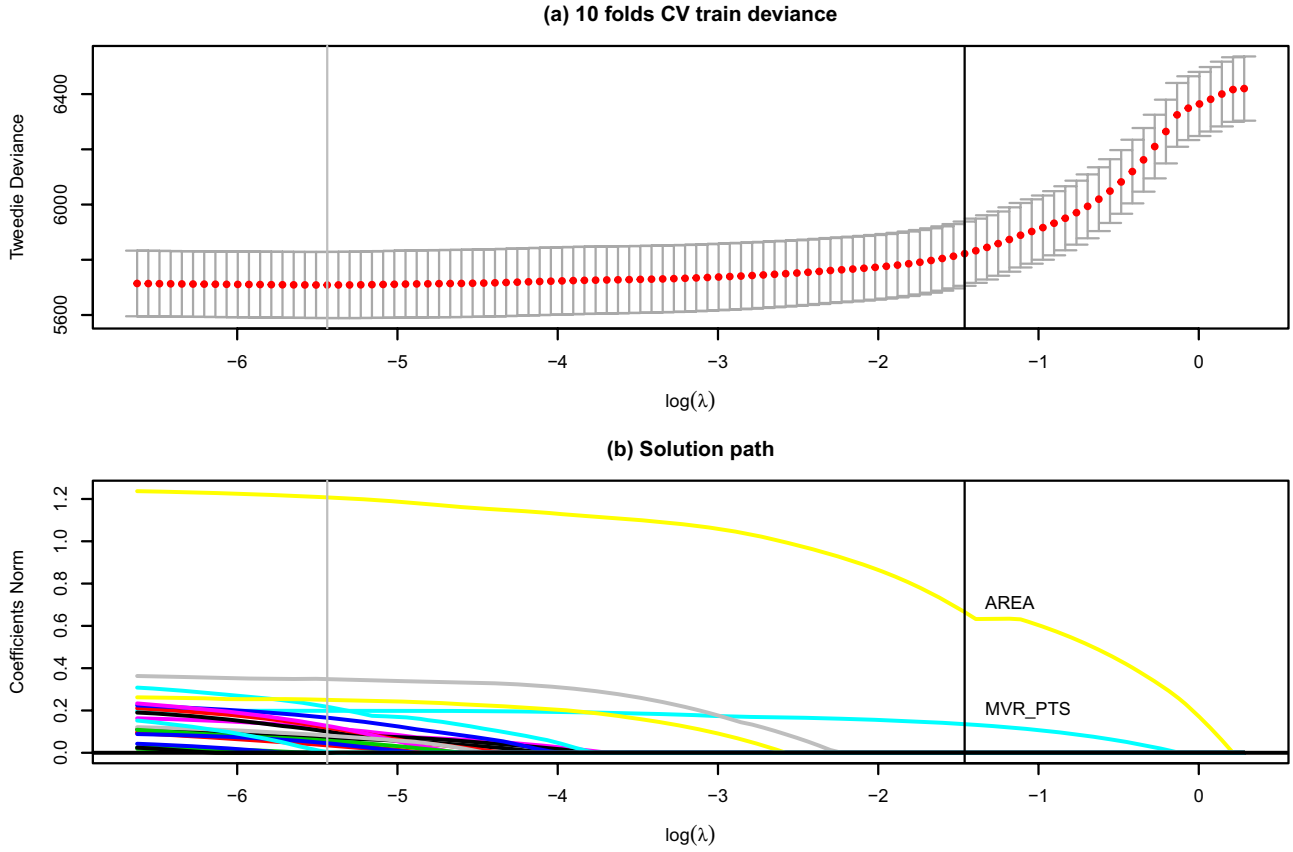

Figure 3. MStweedie with adaptive L_1/L_∞ regularization on AutoClaim data. Panel (a) shows the plot of the 10-fold CV mean deviance (and its standard error) along the λ sequence. Panel (b) plots the norm of the estimates, $||\boldsymbol{\beta}_j||_\infty$, along the λ sequence. In each pane, the grey vertical line indicates the λ for which the CV deviance is minimal and the black vertical line indicates the λ value selected according to the one-standard-error rule.

α, but they all behave as expected: for example, they select the variable CAR_TYPE_4 (corresponding to "Sports Car") only for source 2, corresponding to between-sources sparsity that the $\alpha = 0$ method cannot uncover.

CV is used to select the optimal value of λ. We plot the CV deviance as well as its standard error along the sequence of λ values and display the minimal value as well as the selected λ according to the one-standard-error rule in Figure 3. The figure also contains the plot of the norm of the estimated coefficients for a-L_1/L_∞. It provides an excellent example of why the one-standard-error rule is often favored in practice: its selected model does not have a significantly different model fit than the one minimizing the CV error, but it is considerably sparser.

Furthermore, to have a real data example that approaches more real-world situations, we artificially increase the proportion of zeros of the dataset by sub-sampling the nonzero responses. We consider target proportions between 65% and 95% and remove enough nonzero claim amounts observation from the dataset to reach the given proportion. The new datasets will be smaller in size and the proportion of zeroes may differ between the two sources: the simple random sampling ensures that the disproportion remains the same on average. Table 8 contains details on the new datasets.

The same experimental methodology as with the original data is performed; Figure 4 contains the normalized test deviance and model size, both averaged over 10 replications of the sampling, of the two base algorithms and of a-$L_1/L_\infty(\alpha)$ for $\alpha \in \{0.8, 1\}$, which performed the best on the original dataset. Uniformly over the range of proportion of zeros, our algorithm exhibit performance similar or better than individual Lasso and significantly better than

Table 8. Number of observations and proportion of zeros in the whole dataset and within each sources for the datasets sampled from the AutoClaim dataset to yield a target global proportion of zeros.

AutoClaim: resampled datasets			
	Proportion of zeros		
N	Global	Source 1	Source 2
10,296	61.6	62.5	51.3
9677	65.0	66.3	55.5
8986	70.0	71.2	60.8
8387	75.0	76.1	66.5
7863	80.0	81.0	72.4
7400	85.0	85.8	78.9
6989	90.0	90.6	85.4
6622	95.0	95.2	92.8

the Lasso on the complete dataset. Except for a proportion of zeros of 95%, the adjustment to test data and the sparsity are essentially the same between the multisource algorithms and the independent Lasso. For a proportion of 95%, the multisource algorithm produce significantly sparser models with very similar adjustment. Hence, sharing information between sources for variable selection seem to allow the algorithm to discard more efficiently faint signals from particular features.

5.4. Discussion—Choosing the Regularization

Before fitting the model, it is not obvious which regularization should be used. All simulations as well as the real data example indicate that adaptive regularization should always be preferred to increase the prediction, estimation and selection efficiency of our model. In our experiments, it seems that L_1/L_∞ performs better than L_1/L_2 when the coefficients are the same across tasks for a same feature while the converse is true when the coefficients differ between sources. Additionally, the sparse penalty only helps reduce the size of the model for variable selection (it does not improve the fit) and it seems that most of the gain comes from relatively small α values.

Hence, we can outline some general guideline in selecting the regularization. If the user suspect that the coefficient will vary wildly between sources, then L_1/L_2 should be preferred while L_1/L_∞ should be preferred when the coefficient are thought to be roughly the same. A sparse penalty never seem to hurt, so using $\alpha = 0.5$ may uncover some additional sparsity. Nonetheless, it is difficult to predict what the optimal solution would look like and it might be useful to study the results of a trial run to tune the regularization of the real fit.

6. Conclusion

In this article, we develop a unified algorithm for sparse learning of multisource insurance data using the MStweedie method. The Mstweedie-GPG algorithm we proposed cyclically updates each group of coefficients via the proximal gradient descent scheme and enjoys fast convergence guarantee. This procedure is embedded in a solution path algorithm to achieve the best balance between goodness of fit and model sparsity.

Experiments on simulated data show that our approach clearly outperforms simpler methods in prediction and selection accuracy. It is particularly effective for datasets having distinct structures across the sources. The various regularization schemes behave as expected and thus provide additional flexibility for our algorithm to allow user specification of the desired type of sparsity. While our implementation scales well

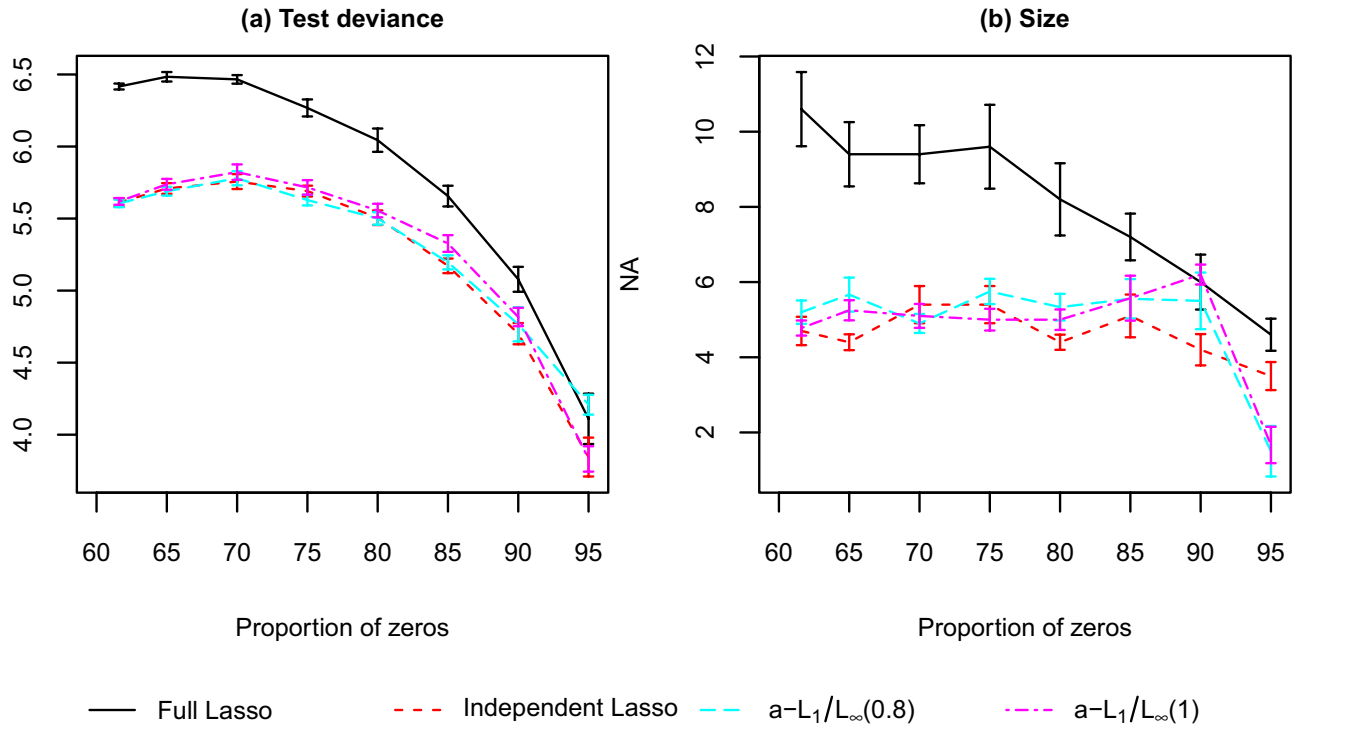

Figure 4. For the two base algorithms and the two best multisource algorithms: (a) normalized test deviance and (b) number of variables in the model, both averaged over 10 replications of the sampling of the AutoClaim dataset to yield target global proportions of zeros. The error bars represent one standard error around the mean.

with the number of observations and variables in a dataset, we caution that an increasing number of sources may slow down the calculation because of the increased number of Euclidean projections required. When applied to real data constituted of aggregate claim amount of the automobile insurance, our procedure convey similar messages to those from the simulated experiments. We also note that although our approach is specifically designed for the Tweedie model with actuarial applications, it is possible to develop similar algorithms for alternative model choices.

In addition, though beyond the scope of our work, a promising approach is to use the multivariate copula to account for the conditional correlation between data sources. For example, Shi (2016) and Frees et al. (2018) proposed multivariate Tweedie copula models, Czado et al. (2012) used a copula on the frequency-severity pair of a single claim with Gamma severity (see also Shi and Zhang 2013 and Shi, Feng, and Ivantsova 2015), and Frees, Shi, and Valdez (2009) use a copula to jointly model a single frequency with hierarchical Generalized Beta claim amounts (see also Frees and Valdez 2008). There is also work on joint modeling of multivariate claim counts (e.g., Bermúdez and Karlis 2011; Nikoloulopoulos 2013; Shi and Valdez 2014). See Frees, Lee, and Yang (2016) and many references therein for a comprehensive review of multivariate insurance claim data modeling. Accordingly, variable selection of multisource data within a multivariate copula model framework can be a promising topic and we leave it for further investigation.

Last but not least, we note that the Tweedie model has a wide range of applications well beyond the scope of our presentation in this article. Examples of nonnegative valued data with excess zeros can also be found in other actuarial settings (Tong, Mues, and Thomas 2013; Frees, Jin, and Lin 2013; Frees, Gao, and Rosenberg 2011; Lauderdale 2012), and in ecology (Blakey et al. 2016; Foster and Bravington 2013; Zhang 2011), fishery (Ancelet et al. 2010; Shono 2008), meteorology (Dunn 2004; Smyth 1996; Swan 2006), and health (Buu et al. 2011; Moger and Aalen 2005; Smyth 1996), to name a few. We hope that this work builds new and useful research tool for many of these promising applications.

Supplementary Materials

MSTweedie The `R` package implementing our proposed methods available at the address *https://github.com/fontaine618/MSTweedie*.

AutoClaim The dataset used in the real data experiment is available within `MSTweedie` package.

Appendices This appendix file contains additional numerical examples and results not shown in the main article. (appendix.pdf)

Acknowledgments

We sincerely thank the editor, the associate editor, and two anonymous reviewers for their valuable comments.

Funding

Yang's research is partially supported by NSERC RGPIN-2016-05174 and FRQ-NT NC-205972. Qian's research is partially supported by NSF DMS-1916376 and JMPC Faculty Fellowship.

References

Ancelet, S., Etienne, M.-P., Benoît, H., and Parent, E. (2010), "Modelling Spatial Zero-Inflated Continuous Data With an Exponentially Compound Poisson Process," *Environmental and Ecological Statistics*, 17, 347–376. [17]

Beck, A., and Teboulle, M. (2009), "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems," *SIAM Journal on Imaging Sciences*, 2, 183–202. [4]

Bermúdez, L., and Karlis, D. (2011), "Bayesian Multivariate Poisson Models for Insurance Ratemaking," *Insurance: Mathematics and Economics*, 48, 226–236. [17]

Blakey, R. V., Law, B. S., Kingsford, R. T., Stoklosa, J., Tap, P., and Williamson, K. (2016), "Bat Communities Respond Positively to Large-Scale Thinning of Forest Regrowth," *Journal of Applied Ecology*, 53, 1694–1703. [17]

Buu, A., Johnson, N. J., Li, R., and Tan, X. (2011), "New Variable Selection Methods for Zero-Inflated Count Data With Applications to the Substance Abuse Field," *Statistics in Medicine*, 30, 2326–2340. [17]

Chandler, R. E., and Bate, S. (2007), "Inference for Clustered Data Using the Independence Loglikelihood," *Biometrika*, 94, 167–183. [3]

Condat, L. (2016), "Fast Projection Onto the Simplex and the ℓ_1 Ball," *Mathematical Programming*, 158, 575–585. [12]

Czado, C., Kastenmeier, R., Brechmann, E. C., and Min, A. (2012), "A Mixed Copula Model for Insurance Claims and Claim Sizes," *Scandinavian Actuarial Journal*, 2012, 278–305. [17]

Duchi, J., Shalev-Shwartz, S., Singer, Y., and Chandra, T. (2008), "Efficient Projections Onto the ℓ_1-Ball for Learning in High Dimensions," in *Proceedings of the 25th International Conference on Machine Learning*, ACM, pp. 272–279. [6,13]

Dunn, P. K. (2004), "Occurrence and Quantity of Precipitation can be Modelled Simultaneously," *International Journal of Climatology*, 24, 1231–1239. [17]

Dunn, P. K., and Smyth, G. K. (2005), "Series Evaluation of Tweedie Exponential Dispersion Model Densities," *Statistics and Computing*, 15, 267–280. [3]

Fawcett, T. (2006), "An Introduction to ROC Analysis," *Pattern Recognition Letters*, 27, 861–874. [8]

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. [17]

Frees, E. W., Bolancé, C., Guillen, M., and Valdez, E. (2018), "Joint Models of Insurance Lapsation and Claims," arXiv no. 1810.04567. [17]

Frees, E. W., Gao, J., and Rosenberg, M. A. (2011), "Predicting the Frequency and Amount of Health Care Expenditures," *North American Actuarial Journal*, 15, 377–392. [1,17]

Frees, E. W., Jin, X., and Lin, X. (2013), "Actuarial Applications of Multivariate Two-Part Regression Models," *Annals of Actuarial Science*, 7, 258–287. [17]

Frees, E. W., Lee, G., and Yang, L. (2016), "Multivariate Frequency-Severity Regression Models in Insurance," *Risks*, 4, 4. [1,17]

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. [1]

Frees, E. W., Shi, P., and Valdez, E. A. (2009), "Actuarial Applications of a Hierarchical Insurance Claims Model," *ASTIN Bulletin: The Journal of the IAA*, 39, 165–197. [17]

Frees, E. W., and Valdez, E. A. (2008), "Hierarchical Insurance Claims Modeling," *Journal of the American Statistical Association*, 103, 1457–1469. [17]

Friedman, J., Hastie, T., and Tibshirani, R. (2010), "Regularization Paths for Generalized Linear Models via Coordinate Descent," *Journal of Statistical Software*, 33, 1. [2,5,8]

Gong, P., Ye, J., and Zhang, C. (2012), "Robust Multi-Task Feature Learning," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, pp. 895–903. [8]

Gu, Y., and Zou, H. (2016), "High-Dimensional Generalizations of Asymmetric Least Squares Regression and Their Applications," *The Annals of Statistics*, 44, 2661–2694. [9]

Hastie, T., Tibshirani, R., and Friedman, J. (2009), *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer Series in Statistics (2nd ed.), New York: Springer. [8]

Huang, J., and Zhang, T. (2010), "The Benefit of Group Sparsity," *The Annals of Statistics*, 38, 1978–2004. [2]

Jenatton, R., Mairal, J., Bach, F. R., and Obozinski, G. R. (2010), "Proximal Methods for Sparse Hierarchical Dictionary Learning," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 487–494. [2,6,7]

Jørgensen, B. (1987), "Exponential Dispersion Models," *Journal of the Royal Statistical Society*, Series B, 49, 127–162. [2]

Kim, S., and Xing, E. P. (2012), "Tree-Guided Group Lasso for Multi-Response Regression With Structured Sparsity, With an Application to EQTL Mapping," *The Annals of Applied Statistics*, 6, 1095–1117. [2]

Lauderdale, B. E. (2012), "Compound Poisson–Gamma Regression Models for Dollar Outcomes That Are Sometimes Zero," *Political Analysis*, 20, 387–399. [17]

Lounici, K., Pontil, M., Tsybakov, A. B., and Van De Geer, S. (2009), "Taking Advantage of Sparsity in Multi-Task Learning," arXiv no. 0903.1468. [2]

Lounici, K., Pontil, M., Van De Geer, S., and Tsybakov, A. B. (2011), "Oracle Inequalities and Optimal Inference Under Group Sparsity," *The Annals of Statistics*, 39, 2164–2204. [2]

Moger, T. A., and Aalen, O. O. (2005), "A Distribution for Multivariate Frailty Based on the Compound Poisson Distribution With Random Scale," *Lifetime Data Analysis*, 11, 41–59. [17]

Morales, J., Micchelli, C. A., and Pontil, M. (2010), "A Family of Penalty Functions for Structured Sparsity," in *Advances in Neural Information Processing Systems*, pp. 1612–1623. [2]

Nikoloulopoulos, A. K. (2013), "Copula-Based Models for Multivariate Discrete Response Data," in *Copulae in Mathematical and Quantitative Finance*, Berlin, Heidelberg: Springer, pp. 231–249. [17]

Obozinski, G., Taskar, B., and Jordan, M. (2006), "Multi-Task Feature Selection," Statistics Department, UC Berkeley, Tech. Rep. 2. [3]

——— (2010), "Joint Covariate Selection and Joint Subspace Selection for Multiple Classification Problems," *Statistics and Computing*, 20, 231–252. [2]

Obozinski, G., Wainwright, M. J., and Jordan, M. I. (2008), "Union Support Recovery in High-Dimensional Multivariate Regression," in *2008 46th Annual Allerton Conference on Communication, Control, and Computing*, IEEE, pp. 21–26. [2]

Qian, W., Li, W., Sogawa, Y., Fujimaki, R., Yang, X., and Liu, J. (2019), "An Interactive Greedy Approach to Group Sparsity in High Dimensions," *Technometrics*, 61, 409–421. [2]

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. [2,8,13]

Shi, P. (2016), "Insurance Ratemaking Using a Copula-Based Multivariate Tweedie Model," *Scandinavian Actuarial Journal*, 2016, 198–215. [1,3,17]

Shi, P., Feng, X., and Boucher, J.-P. (2016), "Multilevel Modeling of Insurance Claims Using Copulas," *The Annals of Applied Statistics*, 10, 834–863. [1]

Shi, P., Feng, X., and Ivantsova, A. (2015), "Dependent Frequency–Severity Modeling of Insurance Claims," *Insurance: Mathematics and Economics*, 64, 417–428. [1,17]

Shi, P., and Valdez, E. A. (2014), "Multivariate Negative Binomial Models for Insurance Claim Counts," *Insurance: Mathematics and Economics*, 55, 18–29. [17]

Shi, P., and Zhang, W. (2013), "Managed Care and Health Care Utilization: Specification of Bivariate Models Using Copulas," *North American Actuarial Journal*, 17, 306–324. [17]

Shono, H. (2008), "Application of the Tweedie Distribution to Zero-Catch Data in CPUE Analysis," *Fisheries Research*, 93, 154–162. [17]

Simon, N., Friedman, J., Hastie, T., and Tibshirani, R. (2013), "A Sparse-Group Lasso," *Journal of Computational and Graphical Statistics*, 22, 231–245. [2,4]

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*, Citeseer, pp. 572–580. [17]

Smyth, G. K., and Jørgensen, B. (2002), "Fitting Tweedie's Compound Poisson Model to Insurance Claims Data: Dispersion Modelling," *ASTIN Bulletin: The Journal of the IAA*, 32, 143–157. [1,3]

Swan, T. (2006), "Generalized Estimating Equations When the Response Variable has a Tweedie Distribution: An Application for Multi-Site Rainfall Modelling," Ph.D. dissertation, University of Southern Queensland. [17]

Tibshirani, R. (1996), "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society*, Series B, 58, 267–288. [2]

Tibshirani, R., Bien, J., Friedman, J., Hastie, T., Simon, N., Taylor, J., and Tibshirani, R. J. (2012), "Strong Rules for Discarding Predictors in Lasso-Type Problems," *Journal of the Royal Statistical Society*, Series B, 74, 245–266. [8]

Tong, E. N., Mues, C., and Thomas, L. (2013), "A Zero-Adjusted Gamma Model for Mortgage Loan Loss Given Default," *International Journal of Forecasting*, 29, 548–562. [17]

Turlach, B. A., Venables, W. N., and Wright, S. J. (2005), "Simultaneous Variable Selection," *Technometrics*, 47, 349–363. [4]

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. [1]

Varin, C., Reid, N., and Firth, D. (2011), "An Overview of Composite Likelihood Methods," *Statistica Sinica*, 21, 5–42. [3]

Vincent, M., and Hansen, N. R. (2014), "Sparse Group Lasso and High Dimensional Multinomial Classification," *Computational Statistics & Data Analysis*, 71, 771–786. [2]

Yang, Y., Qian, W., and Zou, H. (2018), "Insurance Premium Prediction via Gradient Tree-Boosted Tweedie Compound Poisson Models," *Journal of Business & Economic Statistics*, 36, 456–470. [3]

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. [1,13]

Yuan, M., and Lin, Y. (2006), "Model Selection and Estimation in Regression With Grouped Variables," *Journal of the Royal Statistical Society*, Series B, 68, 49–67. [3]

Zhang, H. H., Liu, Y., Wu, Y., and Zhu, J. (2008), "Variable Selection for the Multicategory SVM via Adaptive Sup-Norm Regularization," *Electronic Journal of Statistics*, 2, 149–167. [2]

Zhang, W. (2011), "cplm: Monte Carlo EM Algorithms and Bayesian Methods for Fitting Tweedie Compound Poisson Linear Models," R Package, available at *http://cran.r-project.org/web/packages/cplm/index.html*. [13,17]

Zhang, Y. (2013), "Likelihood-Based and Bayesian Methods for Tweedie Compound Poisson Linear Mixed Models," *Statistics and Computing*, 23, 743–757. [1,13]

Zhao, P., Rocha, G., and Yu, B. (2009), "The Composite Absolute Penalties Family for Grouped and Hierarchical Variable Selection," *The Annals of Statistics*, 37, 3468–3497. [3]

Zou, H. (2006), "The Adaptive Lasso and Its Oracle Properties," *Journal of the American Statistical Association*, 101, 1418–1429. [2,4,8]