Efficient and proper Generalised Linear Models with power link functions

Vali Asimit

Bayes Business School, City, University of London, 106 Bunhill Row, EC1Y 8TZ, UK

Alexandru Badescu

Department of Mathematics and Statistics, University of Calgary, Calgary, AB T2N 1N4, Canada

Feng Zhou

University of Cambridge, Cambridge, CB2 0SR, UK and Bayes Business School, City, University of London, 106 Bunhill Row, EC1Y 8TZ, UK

Abstract

The generalised linear model is a flexible predictive model for observational data that is widely used in practice as it extends linear regression models to non-Gaussian data. In this paper we introduce the concept of a properly defined generalised linear model by requiring the conditional mean of the response variable to be properly mapped through the chosen link function and the log-likelihood function to be concave. We provide a comprehensive classification of proper generalised linear models for the Tweedie family and its popular subclasses under different link function specifications. Our main theoretical findings show that most Tweedie generalised linear models are not proper for canonical and log link functions, and identify a rich class of proper Tweedie generalised linear models with power link functions. Using self-concordant log-likelihoods and linearisation techniques, we provide novel algorithms for estimating several special cases of proper and not proper Tweedie generalised linear models with power link functions. The effectiveness of our methods is determined through an extensive numerical comparison of our estimates and those obtained using three built-in packages, **MATLAB** *fitglm*, R *glm*2 and **Python** *sm.GLM* libraries, which are all implemented based on the standard Iteratively Reweighted Least Squares method. Overall, we find that our algorithms consistently outperform these benchmarks in terms of both accuracy and efficiency, the largest improvements being documented for high-dimensional settings. This is concluded for both simulated data and real data, which shows that our optimisation-based GLM implementation is a good alternative to the standard Iteratively Reweighted Least Squares implementations available in well-known software.

Keywords: Generalised linear model, Tweedie regression, power link function, self-concordance.

JEL classification: C13, C35, C44

Email addresses: `asimit@city.ac.uk` (Vali Asimit), `abadescu@math.ucalgary.ca` (Alexandru Badescu), `feng.zhou@city.ac.uk and feng.zhou@mrc-bsu.cam.ac.uk` (Feng Zhou)

1. Introduction

1.1. Literature Review and Main Goals

Generalised linear modelling (GLM) is a predictive model for observational data which creates a bridge between statistics and machine/statistical learning. That is, GLM provides not only statistical goodness of fit evidence (Nelder and Wedderburn, 1972; McCullagh et al., 1989; Bickel and Doksum, 2015) but also machine/statistical learning evidence such as feature/variable selection (Kuo and Mallick, 1998; Hastie et al., 2001).

GLMs have been successfully implemented in different research fields, and it is vastly used in insurance risk modelling; see e.g., (Debón et al., 2008) for mortality modelling, (Eling and Wirfs, 2019) for cyber risk modelling, (Delong et al., 2021) for insurance pricing, etc. Most of these applications assume independent observations, but insurance applications may require non-independent data, and one example is longitudinal data, and this setting is investigated in Antonio and Beirlant (2007).

The basic GLM requires assumptions about two key quantities, the underlying parametric distribution and the choice of *link function (LF)*. The estimation procedure is based on an optimisation algorithm if the most common estimation method is chosen, i.e. *maximum likelihood estimation (MLE)*. The asymptotic theory of M-estimators requires a concave log-likelihood function, which is the ideal setting so that efficient and stable estimates are obtained; the existence and uniqueness of the MLE estimator is an essential assumption that requires some regularity conditions (Wedderburn, 1976; Mäkeläinen et al., 1981). Consequently, we introduce the concept of a *proper* GLM which requires the conditional mean of the response variable to be properly mapped through the chosen LF and for the log-likelihood function to be well-defined and concave. Since the GLM literature typically relies on exponential dispersion models (Jørgensen, 1987), *our first main goal* is to provide a classification of proper GLMs under this modelling assumption for different LF specifications. This allows the modeller to reduce the numerical issues and understand which combination of parametric family and LF would provide the best possible setting for implementation purposes. The most common LFs belong to the class of *log* or *power* functions, see e.g. McCullagh et al. (1989) and Bickel and Doksum (2015), and thus the main focus will be on these choices.

The most popular algorithms for fitting exponential dispersion GLMs are *Iteratively Reweighted Least Squares (IRLS)*, *Broyden-Fletcher-Goldfarb-Shanno (BFGS)* and *Limited-memory BFGS (L-BFGS)*. IRLS is the standard algorithm which is reasonably scalable when the number of covariates/features is smaller than the sample size. However, IRLS requires inverting the Hessian matrix at every step, which is computationally challenging in non-sparse problems when either the number of features/covariates or the sample size is small. A remedy for this is given by either BFGS or L-BFGS, where the inverse of the Hessian is approximated so that it is feasible to solve higher-dimensional GLM Regressions.*The second main goal* of the paper is to identify viable alternative estimation algorithms to IRLS. Given that the underlying distribution of the response variable is parametrised according to an exponential dispersion family, the MLE could also be obtained via the vanilla Newton's method, which by design is the same as IRLS if the *canonical* LF is in place; the application of Newton's method is also known as the Fisher Scoring method in the GLM literature. Our aim is to improve this estimation method for both proper and not proper GLM settings, by making use of the mathematical properties of *power* LFs. For convex problems, Newton's algorithm can be further refined if the objective function is in addition *self concordant (SC)*, i.e. a convex function whose third derivative is bounded relative to the second derivative in the interior of its domain.[1] This property allows defining an augmented Newton's method which requires a fewer number of iterations for convergence to the optimal solution, see e.g. Boyd and Vandenberghe (2004) or Nesterov (2004) for further details on SC and their fast convergence iterative methods. Since the log-likelihood associated to special cases of Tweedie GLMs (e.g. Poisson and Gamma) equipped with some particular *power* LF specifications is an SC function, we rely on this method for implementing them.[2] For non-convex problems, which is typically the case for many exponential dispersion GLMs (e.g. Inverse Gaussian with *power* LFs), the use of standard IRLS-type algorithms leads to significant computational problems, as illustrated in the next subsection. In such cases, one could either construct bespoke optimisation algorithms designed to tackle a specific problem or rely on mainstream optimisation tools (e.g. generic interior-point methods) if the former are not available. In this paper we also aim to identify tractable solutions for non-convex GLM instances by exploring linearisation techniques, see e.g. Boyd et al. (2011).

Finally, we would like to reiterate that the IRLS methodology approximates the MLE estimates, though IRLS is a very general method that is implemented in various forms in all well-known software. The differences between various implementations is given by bespoke solutions to overcome the lack of convergence that is not guaranteed by IRLS, and therefore, software engineers came up with different solutions; this explains why **R/Python/MATLAB** usually lead (if convergence is achieved) to different estimates even if the starting values and all other settings are the same. Our Sections 5 and 6 show that the lack of convergence is not a negligible issue in GLM deployment for both simulated and real data. The IRLS lack of convergence lead the developers of **MATLAB** *fitglm* and **R** *glm2* to provide bespoke solutions for this problem by adding *step-halving* arguments that are convergence enablers. For example, the earlier version of **R** *glm* was modified for this reason, and **R** *glm2* has an enhanced *step-halving* implementation that is a step ahead, but it does not resolve the issue in its totality; for details, see Marschner (2011) or the **R** *glm2* documentation.

1.2. Motivation and Contributions

The impact of using standard IRLS-based built-in packages on fitting not proper exponential dispersion GLMs is illustrated in the following motivational example. Specifically, using synthetic data, we compare the estimates of an Inverse Gaussian GLM based on the *log* LF, which is an example of a not proper GLM due to the non-concavity of its log-likelihood function, ob-

[1] In a GLM context, a modified version of the SC property with a different control of the third derivative has been used by Bach (2010) for analyzing the statistical properties of Logistic Regressions.

[2] We should note that the augmented Newton's method for SC objective functions still requires the inverse of the Hessian matrix, but in a much lower number of iterations, which reduces the computational time. If the size of the GLM is large, then one may need compromises like those given by BFGS and L-BFGS algorithms where the inverse of the Hessian is efficiently computed, although we do not recommend this choice unless the augmented Newton's method is overwhelmed by the size of the problem. In conclusion, the SC objective functions are expected to bring an improvement to IRLS, and large sized problems could be combined with the Hessian inverse approximations brought by L-BFGS or BFGS.

tained with either **MATLAB**'s *fitglm* library or the non-linear optimisation solver provided by **MATLAB**'s *fmincon* function. Figure 1 displays box plots of the ratio between the L_1 distance (from the true value) of the estimates obtained with the latter method and those computed using **MATLAB**'s *fitglm* values. The results suggest that the *fmincon*-based estimation significantly outperforms the *fitglm* counterpart, especially for large size problems, which indicates that IRLS is not designed to perform well for not proper GLM settings.

To summarize, for any GLM implementation, one should not only consider a proper framework, but also construct bespoke algorithms to deal with the optimisation problem when possible. *Our contributions* address both these fundamental issues. *First*, we provide a comprehensive characterisation of proper MLE-based GLMs for a variety of exponential dispersion models, including the Tweedie family and its well-known special cases, under various LF specifications. Our main theoretical findings indicate that most of Tweedie generalised linear models are not proper for *canonical* and *log link* functions, and identify a rich class of proper Tweedie generalised linear models with power link functions. Consequently, using the Tweedie family for GLM implementation needs a careful approach, since, despite its very flexible parametrisation, the non-standard (Tweedie) models may lead to serious computational issues. *Second*, for a few standard Tweedie GLMs equipped with special cases of *power* LFs, we provide efficient and accurate bespoke algorithms for solving high-dimensional problems which cannot be properly tackled with standard IRLS-type methods. Specifically, we propose the *Newton's method for Self-Concordant problems (NSC)* for solving Poisson and Gamma Regressions and the *Alternating Linearisation Methods (ALM)* algorithm for Inverse Gaussian Regressions. We provide a comprehensive comparison between these algorithms and those available in the standard built-in GLM libraries from various software, such as, **MATLAB** *fitglm*, **R** *glm*2 and **Python** statsmodels *sm.GLM*. We find that our methods outperform these benchmarks in terms of both accuracy and efficiency, the largest improvements being documented for high-dimensional problems.

The remainder of the paper is organized as follows. Section 2 introduces the notion of *proper* GLMs for exponential dispersion models and reviews the LF candidates. Section 3 provides a comprehensive classification of proper Tweedie GLMs and its subclasses. Section 4 introduces the NSC and ALM algorithms for solving Poisson and Gamma, and Inverse Gaussian Regressions, respectively. The numerical comparison between these algorithms and the standard built-in libraries from **MATLAB**, **R** and **Python** is illustrated in Section 5 for simulated data, while a vast real data analysis is provided in Section 6 for two well-known insurance datasets; both sections show ample evidence for our optimisation-based implementation that is an alternative GLM implementation to the usual IRLS implementation. Section 7 concludes the paper.

2. Proper GLMs and LF candidates for exponential dispersion models

A univariate GLM setting assumes that the response variable Y, defined on $\mathcal{Y} \subseteq \Re$, is explained by covariates/features $\boldsymbol{X}$ defined on $\mathcal{X} \subseteq \Re^d$. Let $\{P_{\theta,\phi} : \theta \in \Theta \subseteq \Re, \phi \in \Phi \subseteq \Re\}$ be the parametric set of distributions for Y, which is assumed to be an *exponential dispersion model*

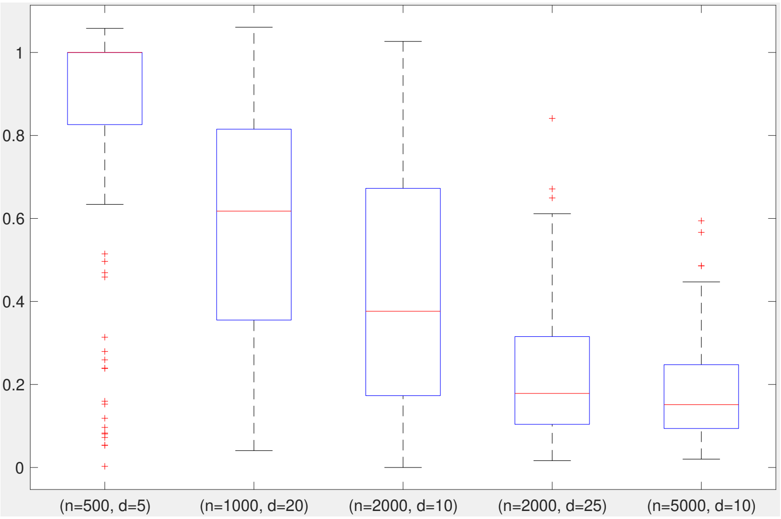

Figure 1: Box plots of MATLAB *fmincon* vs *fitglm* for Inverse Gaussian GLM
Notes: This figure shows the box plots of the ratio between the L_1 distance (from the true value) of the MLE-based GLM solutions obtained with **MATLAB**'s *fmincon* function and the IRLS-based GLM solution obtained with **MATLAB**'s *fitglm* library. Each box plot is constructed based on $N = 500$ simulations according to the DGP scheme outlined in Appendix C, for different specifications for the number of observations and the number of covariates. All GLMs are fitted with *log* LFs, i.e. a non-proper GLM.

characterised by the following probability density/mass function:[3]

$$\log\left(f_Y(y;\theta,\phi)\right) = \frac{\theta y - b(\theta)}{a(\phi)} + c(y,\phi). \tag{2.1}$$

Here, $a(\cdot)$, $b(\cdot)$ and $c(\cdot,\cdot)$ are real-valued functions defined on Φ, Θ and $\mathcal{Y}\times\Phi$, respectively, and ϕ is the dispersion parameter. When ϕ is fixed, (2.1) resembles an exponential family with *canonical* parameter θ. Under standard regularity conditions, the mean and variance of Y are

$$\mathrm{E}\left[Y\right] = b'(\theta) \quad \text{and} \quad \mathrm{Var}\left[Y\right] = a(\phi)b''(\theta). \tag{2.2}$$

The GLM consists of n independent r.v.'s (observations) $Y_1,\cdots,Y_n$ with Y_i distributed according to (2.1) with parameters θ_i and ϕ, and functions $a_i\left(\phi\right), b\left(\theta_i\right)$ and $c\left(y_i,\phi\right)$, and conditional mean linked through a linear predictor $\eta_i = \boldsymbol{x}_i^\top\boldsymbol{\beta}$ via a real-valued function h, so that

$$\mathrm{E}\left[Y_i \mid \boldsymbol{X}_i = \boldsymbol{x}_i\right] = h\left(\boldsymbol{x}_i^\top\boldsymbol{\beta}\right). \tag{2.3}$$

Here, $\boldsymbol{x}_i$ is a d-dimensional vector of realized features/covariates for any $i = 1,\ldots,n$.[4]
The inverse function of h, provided that it exists, is known as the *link function (LF)* and it is denoted by $g = h^{-1}$. The standard GLM literature differentiates the GLMs by the parametric choice made in (2.1) and the preferred LF g. However, from the maximum likelihood estimation (MLE) perspective, the function h is more relevant than g, and thus, the remaining results

[3] Although the univariate assumption for the response variable Y is not essential, it simplifies the exposition.

[4] Note that although the linear predictor suggests observing d covariates/features (since $\mathcal{X}\subseteq\Re^d$), in fact we only assume $d-1$ covariates as we impose $x_{i,1} = 1$ for any $i = 1,\ldots,n$ almost surely. This convention simplifies the notation, so that the linear predictor becomes $\eta_i = \boldsymbol{x}_i^\top\boldsymbol{\beta} = \beta_0 + \beta_1 x_{i,1} + \ldots + \beta_{d-1}x_{i,d-1}$.

are described in terms of the former. If the dispersion parameter ϕ is known (otherwise it is estimated through the variance function from (2.2)), the MLE associated with the GLM defined in Equations (2.1) and (2.3) is obtained by solving the following non-linear optimisation problem

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}\in\Re^d}{\arg\max} \;\; \ell\left(\boldsymbol{\beta}\right) = \sum_{i=1}^{n} \frac{\theta_i y_i - b\left(\theta_i\right)}{a_i\left(\phi\right)} \;\; \text{with} \;\; \theta_i = \left(b'^{-1} \circ h\right) \left(\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right). \tag{2.4}$$

Without loss of generality, we let $a_i\left(\phi\right) = a\left(\phi\right)$.[5] The above optimisation problem is well-defined and admits a (unique) solution if the functions a, b and h satisfy certain regularity conditions. These constraints formalise the concept of a *proper GLM* and are summarised below.

Definition 2.1. *The GLM defined in Equations* (2.1) *and* (2.3) *is said to be* proper *if the following two conditions are satisfied:*

C1. *The conditional mean relationship from* (2.3) *is properly mapped, i.e.* $h : \Re \to b'\left(\Theta\right) \subseteq Conv\left(\mathcal{Y}\right)$ *with* $b' : \Theta \to b'\left(\Theta\right)$ *an injective function.*[6]

C2. *Assume that the likelihood function is well-defined in* (2.4)*. The individual likelihood contribution is a (strictly) concave function, i.e.*

$$\begin{cases} \operatorname{sgn}\left(a\left(\phi\right)\right)\cdot\left(y\cdot\left(b'^{-1}\circ h\right)\left(\eta\right) - \left(b\circ b'^{-1}\circ h\right)\left(\eta\right)\right) \; \textit{is (strictly) concave} \\ \qquad\qquad\qquad \textit{in } \eta \textit{ on } \Re \textit{ for any given } y \in \mathcal{Y}, \end{cases}$$

where sgn *is the signum function.*

Condition **C1** ensures that the GLM estimation is well-defined. More specifically, we require the function b' to be injective, so that it admits an inverse.[7] Condition **C2** implies that the likelihood function ℓ defined in (2.4) is a concave function in $\eta \in \Re$, since the composition of a concave function with an affine mapping is concave and the sum of concave functions is also concave; in other words, (2.4) is a concave programming instance. Consequently, under the constraints from Definition 2.1, the optimisation problem in (2.4) leads to solutions which are global maximum (see e.g. Boyd and Vandenberghe (2004)). Note that the asymptotic distribution of $\hat{\boldsymbol{\beta}}$ – like any M-estimator – requires Equation (2.4) to have a unique solution, which is not always guaranteed. However, this condition is always satisfied if the function from Condition **C2** is strictly concave.The technical conditions for existence and uniqueness of the MLE estimate are well-known (see e.g. Wedderburn (1976) and Mäkeläinen et al. (1981)), and are standard in the literature, i.e. the log-likelihood function is strictly concave and some boundary conditions are

[5] A popular choice in the GLM literature is to consider $a_i\left(\phi\right) = a\left(\phi\right)/w_i$ with $a\left(\phi\right) = \phi$ and w_i non-negative fixed weights for all $i = 1, \ldots, n$. Under this assumption, the non-linear optimisation from Equation (2.4) is equivalent to solving a weighted MLE for a GLM where the response variable follows a canonical one-parameter exponential family distribution. While this could simplify the estimation of $\boldsymbol{\beta}$, and some bespoke model adequacy is typically available to check whether the predefined weights w_i are acceptable, in reality, this is more like a trial error approach which is often resolved by relying on domain knowledge.

[6] Note that $Conv$ is the convex-hull of a set. In addition, $Conv(\mathcal{Y})$ should be read as $\mathcal{Y}$ when Y is continuously distributed, while the convex hull operator makes a difference when Y is a discrete random variable (see e.g. Bernoulli and Poisson families).

[7] The function b' is automatically surjective since the codomain coincides with its image $b'\left(\Theta\right)$.

satisfied. The MLE solutions could be on the boundary of the parameter space, which makes the estimation quite problematic, but we exclude such extreme cases from our analysis.
The standard choice for solving (2.4) is to assume the function h satisfies

$$h\left(\eta\right) = b'\left(\eta\right), \qquad \eta \in \Re. \tag{2.5}$$

Under the specification from (2.5), its equivalent LF g is known as the *canonical LF*. The sufficient conditions for a proper *canonical LF*-based GLM are summarised in the lemma below.

Lemma 2.2. *Let a GLM be equipped with its canonical LF. The MLE-based GLM is proper if* $\Theta = \Re$ *and* b *is strictly convex (concave) on* Θ *provided that* $a\left(\phi\right) > 0$ *(*$a\left(\phi\right) < 0$*) for all* $\phi \in \Phi$.

Although the *canonical* LF has useful mathematical/statistical properties, it does not always satisfy the conditions from Lemma 2.2, and therefore leads to not proper GLMs. Below, we briefly introduce two of the most popular alternative choices in the literature, namely the *log* and *power* classes of LFs.[8] The *log* LF is defined by taking

$$h(\eta) = e^{\eta}, \qquad \eta \in \Re. \tag{2.6}$$

Similar to the previous case, this choice may fail to produce a proper GLM in certain situations, but a general classification as in Lemma 2.2 for such models is not available. Moreover, *log* LFs has been further associated to computationally unstable MLE procedures, which leads us to considering the following family of LFs which could address some of these issues due to their appealing mathematical properties.[9] The *power* LF is defined via the following expression

$$h\left(\eta\right) = \eta^{\gamma}, \qquad \eta \in \Re \quad \text{and} \quad \gamma \in \Re^{*}. \tag{2.7}$$

Popular cases of *power* LFs used in numerical applications are the *identity*, *square* and *square-root* functions which are obtained by taking $\gamma = 1, 1/2$ and 2 in (2.7), respectively. Furthermore, the *reciprocal* versions of these cases (i.e *reciprocal identity*, *reciprocal square* and *reciprocal square-root*) are obtained by letting $\gamma = -1, -1/2$ and -2, respectively.
Lemma 2.3 provides the sufficient conditions for **C1** to be satisfied under the choice from (2.7).[10]

Lemma 2.3. *Let a GLM with a power LF be chosen. Condition **C1** in Definition 2.1 is satisfied if either of the following conditions are satisfied:*

(i) γ *is a non-zero even integer and* $b\prime(\Theta) = \Re_{+} \subseteq Conv\big(\mathcal{Y}\big)$ *such that* $b' : \Theta \rightarrow \Re_{+}$ *is an injective mapping.*

[8] Note that both these functions are also *canonical* LFs for certain GLM cases. A detailed characterisation of these LFs within the context of a proper GLM is provided in Section 3 for several well-known cases of exponential dispersion models. Other classes of LFs such as *probit* and *complementary log-log* are introduced and discussed in Appendix B.3 for Logistic Regressions.

[9] Generaly speaking, *power* LFs are useful for constructing convex optimisation algorithms for estimating GLMs in an accurate and efficient way. Examples of such algorithms are provided in Section 4.

[10] Note that a general characterisation for Condition **C2** cannot be provided for the *power* LF. The proofs of Lemmas 2.2 and 2.3 follow immediately from Definition 2.1.

(ii) γ *is an odd integer and* $b'(\Theta) = Conv(\mathcal{Y}) = \Re$ *such that* $b' : \Theta \to \Re$ *is an injective mapping.*

The above result helps us identify when a GLM is not proper due to Condition **C1** violation. For example, a direct consequence of Lemma 2.3 is that *power* LFs are not appropriate choices for GLMs where the function b' has a bounded image; this is the case of Logistic Regression (see Appendix B.2 for more details).

One way to tackle the not proper GLM issue for *power* LFs is to consider restrictions and/or modifications to these functions. For this purpose, we first introduce the class of *half-power* LFs which corresponds to taking

$$h(\eta) = \begin{cases} \eta^{\gamma}, & \eta > 0, \\ +\infty, & \eta \leq 0, \end{cases} \tag{2.8}$$

with $\gamma \in \Re^*$.[11] Finally, one can consider the *negative* versions of the *power*/*half-power* functions, called *negative power*/*negative half-power*, respectively, which are obtained by multiplying h from (2.7)/(2.8) by -1.

3. Special examples of GLMs and main results

This section provides a classification of proper MLE-based GLM for a variety of exponential dispersion models and discusses the potential issues associated with the use of the different LFs introduced in Section 2. Specifically, we focus on the more general Tweedie family, together with three of its most popular special cases, namely the Poisson, Gamma and Inverse Gaussian distributions.[12] A summary of proper GLMs is provided at the end of the section.

3.1. Poisson Regression – Poisson family

We assume $Y \sim Poisson(\theta)$ with probability mass function given by

$$\log\left(f_Y(y;\theta,\phi)\right) = \theta y - e^{\theta} - \log(y!), \quad (y,\theta,\phi) \in \mathbb{N} \times \Re \times \{1\}.$$

The above expression is obtained as a special case of (2.1) by taking

$$a(\phi) = \phi = 1, \quad b(\theta) = e^{\theta}, \quad c(y,\phi) = -\log(y!).$$

In addition, $b'(\Theta) = \Re_+^*$ and $b'^{-1}(\mu) = \log(\mu)$. Proposition 3.1 provides a characterisation of a proper Poisson Regression model according to our Definition 2.1.

Proposition 3.1. *Assume that* $Y \sim Poisson(\theta)$*. The Poisson GLM is proper if and only if* $h : \Re \to \Re_+^*$*, and*

$$-y\log(h(\eta)) + h(\eta) \quad \text{is convex in } \eta \text{ on } \Re \text{ for any given } y \in \mathbb{N}. \tag{3.1}$$

[11] Note that the special cases for γ that we considered for the standard *power* LFs are defined in the same way for the *half-power* scenarios, e.g. we use the term *reciprocal half-square-root* for h following Equation (2.8) with $\gamma = -2$.

[12] In addition, the Linear and Logistic Regression models are also illustrated in Section Appendix B though we mention that only the Linear Regression is a special case of the Tweedie Regression.

The Poisson *canonical* LF is the *log* function and this choice leads to a proper GLM due to either Lemma 2.2 or Proposition 3.1. The *power* LF does not satisfy the conditions from Proposition 3.1 unless $\gamma = 2k$ with $k \in \mathbb{N}^*$; specifically, Condition **C1** does not hold unless γ is a non-zero even integer, while Condition **C2** requires $\gamma \geq 1$. The *half-power* LF satisfies the conditions stated in Proposition 3.1 for any $\gamma \in [1, \infty)$. Thus, the simplified Poisson regression (i.e. $\phi = 1$) with a proper *half-power* LF, obtained by taking any $\gamma \geq 1$ in (2.8), leads to solving

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \Re^d}{\arg\max} \quad \ell\left(\boldsymbol{\beta}\right) = \sum_{i=1}^{n} \left(\gamma\, y_i \log\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right) - \left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)^\gamma \right). \tag{3.2}$$

While these *half-power* LFs lead to proper GLMs that could be solved via a general convex programming algorithm, the *half-identity* and *half-square-root* cases can be solved via a computationally efficient algorithm, as outlined in Section 4.1. Finally, note that the *half-square-root* and the standard *square-root* LFs are closely related, but the latter does not satisfy (3.1) because Condition **C2** does not hold in this case. Essentially, the *half-square-root* case optimises the strictly concave instance in (3.2) on the $\Re^d$ cone such that $\boldsymbol{x}_i^\top \boldsymbol{\beta} > 0$ for all $i = 1, \ldots, n$, while the *square-root* solves a similar problem to (3.2) (where $\log\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)$ is replaced by $\log\left|\boldsymbol{x}_i^\top \boldsymbol{\beta}\right|$) on $\Re^d$, but its objective function is not concave on the entire feasibility set, namely $\Re^d$. An analogous differentiation between the *half-identity* and *identity* LFs can be formulated as well. Finally, Condition **C1** is not satisfied for any *negative power* LF or *negative half-power* LF, which are not proper for Poisson GLM.

3.2. Gamma Regression – Gamma family

We assume $Y \sim Gamma(\theta, \phi)$ with probability distribution function given by

$$\log\left(f_Y(y;\theta,\phi)\right) = \frac{\theta y + \log\left(-\theta\right)}{\phi} + \frac{1-\phi}{\phi}\log\left(y\right) - \log\left(\phi^{\frac{1}{\phi}}\Gamma\left(\frac{1}{\phi}\right)\right), \; (y,\theta,\phi) \in \Re_+^* \times \Re_-^* \times \Re_+^*.$$

The above expression is obtained as a special case of (2.1) by taking

$$a\left(\phi\right) = \phi, \quad b\left(\theta\right) = -\log\left(-\theta\right), \quad c\left(y,\phi\right) = \frac{1-\phi}{\phi}\log\left(y\right) - \log\left(\phi^{\frac{1}{\phi}}\Gamma\left(\frac{1}{\phi}\right)\right).$$

In addition, $b'\left(\Theta\right) = \Re_+^*$ and $b'^{-1}\left(\mu\right) = -\mu^{-1}$. Proposition 3.2 provides a characterisation of a proper Gamma Regression model according to our Definition 2.1.

Proposition 3.2. *Assume that $Y \sim Gamma(\theta, \phi)$. The Gamma GLM is proper if and only if $h : \Re \to \Re_+^*$, and*

$$\frac{y}{h(\eta)} + \log\left(h(\eta)\right) \quad \textit{is convex in } \eta \textit{ on } \Re \textit{ for any given } y \in \Re_+^*. \tag{3.3}$$

The *canonical* LF associated to the Gamma GLM is the *reciprocal identity* function. This function does not satisfy the conditions stated in Lemma 2.2 or Proposition 3.2, since Condition **C1** does not hold, and therefore, unlike in the Poisson case, the *canonical* GLM is not proper. A popular alternative for Gamma GLM is represented by the *log* LF; this choice satisfies the conditions stated in Proposition 3.2 and is thus appropriate for Gamma GLM. As in Section 3.1, we now discuss the impact of using *power/half-power* LFs in Gamma GLM. First, a *power* LF does not satisfy the conditions from Proposition 3.2 unless $\gamma = -2k$, with $k \in \mathbb{N}^*$; specifically, Condition **C1** does not hold unless γ is a non-zero even integer, while Condition **C2** requires $\gamma \leq -1$. Second, one could find that *half-power* LFs always satisfy Condition **C1**, but Condition **C2** holds if and only if $\gamma \leq -1$, leading to proper Gamma GLM in this case. Note that the simplified Gamma GLM (i.e. $\phi = 1$) with such proper *half-power* LF is equivalent to solving

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}\in\Re^d}{\arg\max} \quad \ell\left(\boldsymbol{\beta}\right) = \sum_{i=1}^{n} \left(-\gamma \log\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right) - y_i \left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)^{-\gamma} \right), \tag{3.4}$$

where $\gamma \leq -1$. While *half-power* LFs with $\gamma \leq -1$ lead to proper GLMs that could be solved via a general convex programming algorithm, the *half-reciprocal identity* and *half-reciprocal-square-root* cases could be solved via a computationally efficient algorithm, as outlined in Section 4.1. Finally, Condition **C1** is not satisfied for any *negative power* LF or *negative half-power* LF, which are not proper for Poisson GLM.

3.3. Inverse Gaussian Regression – Inverse Gaussian (IG) family

We assume $Y \sim IG(\theta, \phi)$ with probability distribution function given by

$$\log\left(f_Y(y;\theta,\phi)\right) = \frac{\theta y - \sqrt{2\theta}}{-1/\phi} + \frac{1}{2}\left(\log\left(\frac{\phi}{2\pi y^3}\right) - \frac{\phi}{y}\right), \quad (y,\theta,\phi) \in \Re_+^* \times \Re_+^* \times \Re_+^*.$$

The above function is also a special case of (2.1) where

$$a\left(\phi\right) = -\frac{1}{\phi}, \quad b\left(\theta\right) = \sqrt{2\theta}, \quad c\left(y,\phi\right) = \frac{1}{2}\left(\log\left(\frac{\phi}{2\pi y^3}\right) - \frac{\phi}{y}\right).$$

In addition, $b'\left(\Theta\right) = \Re_+^*$ and $b'^{-1}\left(\mu\right) = \frac{1}{2}\mu^{-2}$. Proposition 3.3 provides the characterisation of a proper Inverse Gaussian Regression model according to our Definition 2.1.

Proposition 3.3. *Assume that $Y \sim IG(\theta,\phi)$. The Inverse Gaussian GLM is proper if and only if $h : \Re \to \Re_+^*$, and*

$$\frac{y}{2h^2(\eta)} - \frac{1}{h(\eta)} \quad \textit{is convex in } \eta \textit{ on } \Re \textit{ for any given } y \in \Re_+^* \tag{3.5}$$

The *canonical* LF for the GLM-based on the *IG* distribution is the *reciprocal square* function. Similar to the Gamma scenario, this function does not satisfy the conditions stated in Lemma 2.2 or Proposition 3.3, namely Condition **C1**, and therefore, it is not a proper GLM. Under the *log* LF assumption Condition **C1** is satisfied, but Condition **C2** is violated since (3.5) does not hold. The effect of non-convexity is depicted in our motivational example from Figure 1.

As before, we also investigate the *power* and *half-power* LFs in the context of an *IG* GLM. *First*, we notice that there is no *power* LF that satisfies the conditions in Proposition 3.3; specifically, Condition **C1** does not hold unless γ is a non-zero even integer, while Condition **C2** is satisfied if and only if $\gamma \in [-1, -1/2]$. *Second*, one could find that *half-power* LFs always satisfy Condition **C1**, but Condition **C2** holds if and only if $\gamma \in [-1, -1/2]$, concluding that *half-power* LF leads to a proper GLM only in this case. Given the previous findings, running IG Regressions with *power* or *half-power* LFs would require a compromise. That is, the *power* LF with $\gamma = 2k,\ k \in \mathbb{Z}^*$ is the best possible choice so that constrained programming is avoided (for proper IG GLM with *half-power* LFs such that $\gamma \in [-1, -1/2]$ for which n linear inequality constrains are needed), which is computationally undesirable for large samples. Such choice require an efficient algorithm to solve the non-concave log-likelihood function optimisation. We show how to achieve this in Section 4.2 for the *reciprocal-square-root* LF.

3.4. Main results on Tweedie Regression – Tweedie family

In this section we focus our analysis on a more general class of GLMs based on the Tweedie family, which includes the previous distributions as special/limiting cases. As before, our main goal is to investigate if the Tweedie distribution leads to proper GLMs. Assume that $Y \sim Tweedie(\theta, \phi)$ with probability distribution function defined below

$$\log\left(f_Y(y;\theta,\phi)\right) = \frac{\theta y - K_p(\theta)}{\phi} + \log\left(\mu'_\phi\left((-\infty, y]\right)\right), \qquad (y,\theta,\phi) \in \mathcal{Y} \times \Theta \times \Re^*_+, \tag{3.6}$$

where $\Theta \subseteq \Re$, μ_ϕ is a Radon measure on $\mathcal{Y} \subseteq \Re$ and the function K_p is given by

$$K_p(\theta) := \begin{cases} \dfrac{\alpha-1}{\alpha}\left(\dfrac{\theta}{\alpha-1}\right)^\alpha, & p \in (-\infty, 0] \cup (1, \infty) \setminus \{2\}, \\ e^\theta, & p = 1, \\ -\log(-\theta), & p = 2, \end{cases}$$

with $\alpha = \dfrac{p-2}{p-1}$. The expression from (3.6) is obtained as a special case of (2.1) by taking

$$a(\phi) = \phi, \quad b(\theta) = K_p(\theta), \quad c(y,\phi) = \log\left(\mu'_\phi\left((-\infty, y]\right)\right).$$

Moreover, the Poisson, Gamma and Inverse Gaussian families are obtained as special cases by taking $p = 1$ with $\mathcal{Y} = \mathbb{N}$ and $\Theta = \Re$, $p = 2$ with $\mathcal{Y} = \Re^*_+$ and $\Theta = \Re^*_-$, and $p = 3$ with $\mathcal{Y} = \Re^*_+$ and $\Theta = \Re^*_-$, respectively.[13]

Without loss of generality we henceforth assume that $p \neq \{1, 2\}$, since these two cases have already been investigated in Sections 3.1 and 3.2. Note that one should carefully choose $\Theta, \mathcal{Y}$ and p so that $K_p(\cdot)$ is well-defined on Θ. In this section, we assume that $\Theta \in \{\Re, \Re^*, \Re^*_+, \Re^*_-\}$, and thus, the function b' is well-defined and bijective on Θ only under the three settings considered in the theorem below. Extensions to subsets of these sets are obtainable at the expense of the exposition, and for this reason, we proceed with this simplification.

We now provide a characterisation of proper Tweedie GLMs, where we exclude the previous cases investigated in Sections 3.1 and 3.2 and Appendix B.1. First, we identify in Theorem 3.4 all possible settings under which Condition **C1** from Definition 2.1 is satisfied.

[13] Other notable examples are Gaussian ($p = 0$ with $\mathcal{Y} = \Theta = \Re$), Compound Poisson-Gamma ($1 < p < 2$ with $\mathcal{Y} = \Theta = \Re_+$) and Positive stable ($p > 2$ with $\mathcal{Y} = \Theta = \Re_+$).

Theorem 3.4. *Let $Y \sim Tweedie(\theta, \phi)$ parameterised as in (3.6) with $p \in (-\infty, 0) \cup (1, 2) \cup (2, \infty)$ (or equivalently, $\alpha \in (-\infty, 2)\backslash\{0, 1\}$) such that $\mathcal{Y}, \Theta \in \{\Re, \Re^*, \Re^*_+, \Re^*_-\}$. Then, Condition **C1** is only satisfied for the following settings:*

a) $\Theta = b'(\Theta) = \Re^*_+$ (or $\Re_+$), $\mathcal{Y} \in \{\Re^*_+, \Re\}$ (or $\mathcal{Y} \in \{\Re_+, \Re\}$) and $1 < \alpha < 2$ (which is equivalent to $p < 0$), with $h : \Re \to \Re^*_+$ (or $h : \Re \to \Re_+$);

b) $\Theta = \Re^*_-$, $b'(\Theta) = \Re^*_+$, $\mathcal{Y} \in \{\Re^*_+, \Re_+, \Re\}$ and $\alpha \in (-\infty, 1) \setminus \{0\}$ (which is equivalent to $p \in (1, \infty) \setminus \{2\}$), with $h : \Re \to \Re^*_+$;

c) $\Theta = \Re$, $b'(\Theta) = \Re^*_+$, $\mathcal{Y} \in \{\Re^*_+, \Re_+, \Re^*\}$, $\alpha \in \{-2l + 1 : l \in \mathbb{N}^*\}$, with $h : \Re \to \Re^*_+$.

d) $\Theta = \Re$, $b'(\Theta) = \Re^*$, $\mathcal{Y} \in \{\Re^*, \Re\}$, $\alpha \in \{-2l : l \in \mathbb{N}^*\}$, with $h : \Re \to \Re^*$.

Setting a) includes a pedantic reference on whether the response variable could or could not include $y = 0$, and thus, we made a difference between the cases $\Theta = \Re^*_+$ and $\Theta = \Re_+$. Note that the generic Condition **C1** in Definition 2.1 requires the range of $\mathrm{E}[Y]$, namely $b'(\Theta)$, to be a subset of $Conv(\mathcal{Y})$, though a more practical condition would be $b'(\Theta) = Conv(\mathcal{Y})$, which we assume henceforth. Setting c) is a subcase of setting b) from the implementation point of view, since the modeller chooses the Tweedie models so that $\mathcal{Y}$ matches the data range of values. However, our classification in Theorem 3.4 has to differentiate between models with different parameter sets Θ. The next results focus on the validity of Condition **C2** from Definition 2.1 for the above Tweedie GLM settings under the LF specifications introduced in Section 2. The *power* LF class, together with its restrictions/modifications, is investigated in Theorem 3.5 below.

Theorem 3.5. *Let $Y \sim Tweedie(\theta, \phi)$ parameterised as in (3.6) with $b'(\Theta) = \mathcal{Y}$, for which condition **C1** is satisfied. Then, Condition **C2** is not satisfied by settings a)–d), for any*

(i) power LF, except for the following cases:

- *setting b) with $0 < \alpha < 1$ and $\gamma = -2k$, for any $k \in \mathbb{N}^*$, with $(1 - \gamma)\alpha \leq 1$,*
- *setting b) with $\alpha < 0$ and $\gamma = 2k$, for any $k \in \mathbb{Z}^*$,*
- *setting c) and $\gamma = 2k$, for any $k \in \mathbb{Z}^*$.*

(ii) half-power LF, except for the following cases:

- *setting a) with $1 < \alpha < 2$ and $\frac{\alpha-1}{\alpha} \leq \gamma \leq \alpha - 1$,*
- *setting b) with $0 < \alpha < 1$ and $\frac{\alpha-1}{\alpha} \leq \gamma \leq \alpha - 1$,*
- *setting b) with $\alpha < 0$ and $\gamma \leq \alpha - 1$ or $\frac{\alpha-1}{\alpha} \leq \gamma$,*
- *setting c) with $\alpha \in \{-2l + 1 : l \in \mathbb{N}^*\}$ and $\gamma \leq \alpha - 1$ or $\frac{\alpha-1}{\alpha} \leq \gamma$.*

(iii) negative power or negative half-power LF.

We notice that the above results are in agreement with our previous findings. For example, one could recover our discussion from Section 3.3 on proper IG GLMs, which is a special case of Theorem 3.5 if we take $p = 3$ (or equivalently $\alpha = 1/2$), where we found that proper IG GLMs with *half-power* LF are achieved if and only if $\gamma \in [-1, -1/2]$. In addition, Theorem 3.5 provides necessary and sufficient conditions for proper GLMs under other distributional assumptions.

For example, Tweedie GLMs based on Positive stable distributions (i.e. $p > 2$ or equivalently $0 < \alpha < 1$) are proper only for *power LFs* with $\gamma = -2k$, $k \in \mathbb{N}^*$, with $(1-\gamma)\alpha \leq 1$ and *half-power LFs* with $\frac{\alpha-1}{\alpha} \leq \gamma \leq \alpha - 1$. Similarly, the Compund Poisson-Gamma GLM (i.e. $1 < p < 2$ or equivalently $\alpha < 0$) is proper only for *power* LFs with $\gamma = 2k$, $k \in \mathbb{Z}^*$ or *half-power* LFs with $\gamma \leq \alpha - 1$ or $\frac{\alpha-1}{\alpha} \leq \gamma$. A complete summary of proper Tweedie GLMs is illustrated in Table 1 of Section 3.5.

Note that if $p \in (-\infty, 0] \cup (1, \infty) \setminus \{2\}$, which is equivalent to $\alpha \in (-\infty, 2] \setminus \{0, 1\}$, then the simplified Tweedie regression (i.e. $\phi = 1$) with LF h is equivalent to solving

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \Re^d}{\arg\max} \quad \ell(\boldsymbol{\beta}) = \sum_{i=1}^{n} \left(y_i(\alpha-1)\left(h\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)\right)^{\frac{1}{\alpha-1}} - \frac{\alpha-1}{\alpha}\left(h\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)\right)^{\frac{\alpha}{\alpha-1}} \right). \tag{3.7}$$

A few comments on (3.7) would help understanding the issues with deploying Tweedie GLMs. *First*, one may discard Condition **C2** at the expense of losing all useful properties of the M-estimators (MLE is only a special case), such as the asymptotic distribution, which questions the asymptotic bias and variance of these estimators. If that is the case, one can only hope for the numerical optimisation to behave well, but this is possible from case to case, and one would need to perform extensive numerical implementations to check whether the optimisation algorithm shows a reasonable performance for specific choices of $(\alpha, \mathcal{Y}, h)$. Such compromise is done in Algorithm 2 for solving (4.5), where $\alpha = \frac{1}{2}$ as $p = 3$, $\mathcal{Y} = \Re_+$, and *reciprocal square-root* LF; one could recover (4.5) from (3.7) for this particular choice of $(\alpha, \mathcal{Y}, h)$. *Second*, there are other parametrisations other than the one in Algorithm 2 for which Condition **C2** is not satisfied while all other regularity conditions in Definition 2.1 hold. In these instances, one has to rely on non-convex optimisation, but more importantly, has to accept that some (possibly all) statistical properties of the MLE estimator may not hold. The modeller needs to identify stable computational methods (as in Algorithm 2) instead of assuming that the general purpose GLM solvers are indeed computationally stable. Finally, we notice that the proper GLMs identified in Theorem 3.5 (ii) require solving a constrained optimisation problem on the convex cone $\left\{\boldsymbol{\beta} \in \Re^d : \boldsymbol{x}_i^\top \boldsymbol{\beta} \geq 0, i = 1, \ldots, n\right\}$. Unfortunately, this is computationally expensive for large values of n, which is a negative attribute. These optimisations could be solved via convex programming and not via off-the-shelf GLM packages that relies on IRLS which cannot be adapted when such constraints are needed.

The classification of proper Tweedie GLMs based on *canonical* and *log* LF is illustrated below.

Theorem 3.6. *Let $Y \sim$ Tweedie(θ, ϕ) parameterised as in (3.6) with $b'(\Theta) = \mathcal{Y}$, for which condition **C1** is satisfied. Then, Condition **C2** is not satisfied by settings a)–d), for any*

(i) canonical LF.

(ii) log LF, except for setting b) with $\alpha < 0$ or setting c).

Thereom 3.6 shows that there are no proper Tweedie GLMs if the *canonical* LF is chosen. In addition, we notice that the Compound Poisson-Gamma GLM is proper for any *log* LF.

Table 1: Summary of proper GLMs and violations of Conditions C1 and C2

Regression model	**LF**	**Predictor** $\left(\hat{y} = h\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)\right)$	**Violations**
Gaussian/Linear	*identity* (canonical)	$\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}$	No
Logistic	*logit* (canonical)	$\left(1+\exp\left(-\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)\right)^{-1}$	No
	probit	$\Phi\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)$	No
	complementary log-log	$1-\exp\left(-\exp\left(-\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)\right)$	No
Poisson	*log* (canonical)	$\exp\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)$	No
	power	$\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{\gamma}$	No, if $\gamma = 2k,\ k \in \mathbb{N}^*$
	half-power	$\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{\gamma} \cdot I_{\{\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}>0\}}$	No, if $\gamma \geq 1$
Gamma	*reciprocal identity* (canonical)	$\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{-1}$	**C1**
	log	$\exp\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)$	No
	power	$\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{\gamma}$	No, if $\gamma = -2k,\ k \in \mathbb{N}^*$
	half-power	$\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{\gamma} \cdot I_{\{\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}>0\}}$	No, if $\gamma \leq -1$
Inverse Gaussian	*reciprocal square* (canonical)	$\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{-1/2}$	**C1**
	log	$\exp\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)$	**C2**
	power	$\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{\gamma}$	**C1**, if $\gamma \neq 2k,\ k \in \mathbb{Z}^*$, and **C2**, if $\gamma \notin [-1,-1/2]$
	half-power	$\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{\gamma} \cdot I_{\{\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}>0\}}$	No, if $\gamma \in [-1,-1/2]$
Tweedie (except of some of the above special cases: Gaussian, Poisson and Gamma)	*canonical*	$\left((1-p)\cdot\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{1/(1-p)}$	see Theorem 3.6
	log	$\exp\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)$	see Theorem 3.6
	power or *negative power*	$\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{\gamma}$ or $-\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{\gamma}$, $\gamma \in \Re^*$	see Theorem 3.5
	half-power	$\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{\gamma} \cdot I_{\{\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}>0\}}$, $\gamma \in \Re^*$	see Theorem 3.5
	negative half-power	$-\left(\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}\right)^{\gamma} \cdot I_{\{\boldsymbol{x}^\top\hat{\boldsymbol{\beta}}>0\}}$, $\gamma \in \Re^*$	see Theorem 3.5

Notes: This table presents a summary of proper GLMs equipped with the LFs discussed in Section 3 and Appendix B, and the potential violations of Conditions **C1** and **C2** from Definition 2.1 associated with these regressions. Φ stands for the $\mathcal{N}(0,1)$ cumulative distribution functionand I_A represents the indicator function for set A.

3.5. Summary results

Table 1 summarises our findings discussed in Section 3 and Appendix B. *First*, we recall that the *canonical* LFs, which are the standard choices in all built-in GLM implementations (available in **MATLAB**, **Python**, **R**, etc.), lead to not proper Tweedie GLMs, except for the Gaussian and Poisson cases. *Second*, *log* LFs tend to have the similar limitations to *canonical* LFs for Tweedie modelling. *Third*, the *power* and *half-power* LFs allow more flexibility than *log* LFs to GLM modelling when proper GLM are sought.

4. Alternative algorithms for GLMs with power LFs

The goal of this section is to not only provide efficient methods for solving high-dimensional problems while addressing the potential numerical issues in the optimisation stage, but to also create tractable models for dealing with non-convex instances, which cannot be tackled with standard built-in GLM algorithms. In this sense, we introduce the *Newton's method for Self-Concordant problems (NSC)* for Poisson and Gamma regressions equipped with some bespoke *half-power* LFs, and the *Alternating Linearisation Method (ALM)* for solving Inverse Gaussian regressions based on the *reciprocal-square-root* LF.[14]

4.1. The NSC *algorithm for Poisson and Gamma Regressions*

The explicit structure of such *self-concordant* functions allows defining a refined Newton's method which is generally more efficient due to a reduced number of iterations.[15] First, we introduce the definition of a *self-concordant* function, which was first provided by Nesterov (2004), although a simplified version is provided in Boyd and Vandenberghe (2004), which we follow in this paper.

Definition 4.1. *Let $f:\Omega\rightarrow\Re$ be a closed convex function*[16] *where $\Omega=dom\,(f)$ is an open set in $\Re^d$ and $f\in\mathcal{C}^3\,(dom\,(f))$. The function f is self-concordant on Ω if the function $g\,(t):=f\,(\boldsymbol{u}+t\,\boldsymbol{v})$ satisfies $|g'''\,(t)|\leq 2\,(g''\,(t))^{3/2}$ for any $t\in dom\,(g)\subseteq\Re$, $\boldsymbol{u}\in dom\,(f)$, and $\boldsymbol{v}\in\Re^d$ such that $\boldsymbol{u}+t\,\boldsymbol{v}\in dom\,(f)$.*

Note that the constant 2 in Definition 4.1, see $|g'''\,(t)|\leq 2\,(g''\,(t))^{3/2}$, is chosen for convenience and helps to identify an explicit upper bound for the total number of iterations required by the Newton's method for SC functions. If constant 2 is replaced by M, i.e. $|g'''\,(t)|\leq M\,(g''\,(t))^{3/2}$, then we say that its equivalent function f is SC with constant M; e.g., if f is SC with constant M, then it is not difficult to show that $\tilde{f}\,(\cdot):=\frac{M^2}{4}f\,(\cdot)$ is SC with constant 2.

We explore the Poisson and Gamma Regressions based on some special choices of *half-power* LFs by solving (3.2) and (3.4), since the associated negative log-likelihoods are not only convex (actually strictly convex in those two cases), but also *self-concordant*. This is illustrated in Theorem 4.2 below, where the *half-identity* and *half-square-root* LFs for Poisson Regression are explored in Theorem 4.2 a), while the *half-reciprocal identity* and *half-reciprocal-square-root* LFs for Gamma Regression are explored in Theorem 4.2 b).

Theorem 4.2. *Let $\{(y_i,\boldsymbol{x}_i):1\leq i\leq n\}$ be a sample of size n drawn from $(Y,\boldsymbol{X})$, where $\boldsymbol{X}=(X_1,X_2,\ldots,X_d)$ with $d\geq 1$ and define $\Omega:=\bigcup_{i=1}^{n}\left\{\boldsymbol{\beta}\in\Re^d:\boldsymbol{x}_i^\top\boldsymbol{\beta}>0\right\}$. The following statements hold:*

a) The MLE-based Poisson GLM equipped with the half-power LF from (2.8) with either $\gamma=2$ (and $\gamma=1$) is self-concordant, and it leads to an optimisation problem with a self-concordant objective function f_P ($\check{f}_P$) on Ω, where

$$\min_{\boldsymbol{\beta}\in\Omega}\ f_P\,(\boldsymbol{\beta}):=\sum_{i=1}^{n}\left(\frac{1}{2}\left(\boldsymbol{x}_i^\top\boldsymbol{\beta}\right)^2-y_i\log\left(\boldsymbol{x}_i^\top\boldsymbol{\beta}\right)\right), \tag{4.1}$$

$$\min_{\boldsymbol{\beta}\in\Omega}\ \check{f}_P\,(\boldsymbol{\beta}):=\sum_{i=1}^{n}\left(\boldsymbol{x}_i^\top\boldsymbol{\beta}-y_i\log\left(\boldsymbol{x}_i^\top\boldsymbol{\beta}\right)\right). \tag{4.2}$$

[14] This is also known as *inverse-square-root* LF, but we avoid referring to 'inverse' since the GLM uses the inverse of a function to identify the functional estimator h with the LF g.

[15] For further details on SC problems and their fast convergence iterative methods, see Boyd and Vandenberghe (2004); Nesterov (2004).

[16] A function $f:A\subseteq\Re^d\rightarrow B$ is closed convex if f is convex and closed on A, where f is closed if for any $\alpha\in\Re$, $\{\boldsymbol{x}\in dom(f):f\,(\boldsymbol{x})\leq\alpha\}$ is a closed set.

b) The MLE-based Gamma GLM equipped with the half-power LF from (2.8) with $\gamma = -2$ (and $\gamma = -1$) is self-concordant, and it leads to an optimisation problem with a self-concordant objective function f_G ($\check{f}_G$) on Ω, where

$$\min_{\boldsymbol{\beta}\in\Omega} \; f_G\left(\boldsymbol{\beta}\right) := \sum_{i=1}^{n} \left(\frac{y_i}{2} \left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)^2 - \log\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)\right), \tag{4.3}$$

$$\min_{\boldsymbol{\beta}\in\Omega} \; \check{f}_G\left(\boldsymbol{\beta}\right) := \sum_{i=1}^{n} \left(y_i \cdot \boldsymbol{x}_i^\top \boldsymbol{\beta} - \log\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)\right). \tag{4.4}$$

As previously mentioned, the constant of an SC function does not have any impact in the actual iterative algorithm, and could change only the upper bound of the total number of steps (that is in an explicit form for SC functions; for details, see the Newton's step in Algorithm 1). One may show that a tighter bound could be obtained for (4.1) and (4.2), i.e. the objective function is SC with constant M_P and $\check{M}_P$, respectively, where

$$M_P = \check{M}_P := 2 \max_{1\le i\le n} \left\{ y_i^{-1/2} I_{\{y_i>0\}} + I_{\{y_i=0\}} \right\},$$

which satisfies $M_P \le 2$. However, no tighter bound (tighter than 2) is possible for the Gamma GLMs in either (4.3) and (4.4).

Theorem 4.2 allows us to use the standard SC algorithm which is detailed in (Nesterov, 2004; Boyd and Vandenberghe, 2004), and is provided here as Algorithm 1.

Algorithm 1: Standard SC algorithm for solving (4.1) and (4.3)

```
Result: $\boldsymbol{z}^{(k^*)}$ which approximates $\boldsymbol{z}^*$, the global optimum of $\min_{\boldsymbol{z}\in\Omega} f(\boldsymbol{z})$ with $f(\cdot)$ being
        SC on $\Omega$, where $k^*$ is the termination step.

Choose $\boldsymbol{z}^{(0)} \in dom\left(f\right)$, $\epsilon > 0$, and $\lambda^* \in \left(0, \tilde{\lambda}\right)$ where $\tilde{\lambda} = \frac{3-\sqrt{5}}{2}$;
Let $\nabla f\left(\cdot\right)$ and $\nabla^2 f\left(\cdot\right)$ be the gradient and Hessian, respectively, of $f$ on $\Omega$;
Define the step/search direction function $\Delta\left(\cdot\right) := \left[\nabla^2 f\left(\cdot\right)\right]^{-1} \nabla f\left(\cdot\right)$ on $\Omega$;
Define $\lambda_f\left(\cdot\right) := \left(\nabla f\left(\cdot\right)^\top \left[\nabla^2 f\left(\cdot\right)\right]^{-1} \nabla f\left(\cdot\right)\right)^{1/2}$ on $\Omega$;

Step 1: Damped phase
  (i) If $\lambda_f\left(\boldsymbol{z}^{(0)}\right) < \lambda^*$ go to Step 2;
  (ii) While $\lambda_f\left(\boldsymbol{z}^{(k)}\right) \ge \lambda^*$ do $\boldsymbol{z}^{(k+1)} = \boldsymbol{z}^{(k)} - \frac{1}{1+\lambda_f\left(\boldsymbol{z}^{(k)}\right)}\Delta\left(\boldsymbol{z}^{(k)}\right)$ for all $k \ge 0$;

Step 2: Newton (or quadratically convergence) phase
While $\lambda_f\left(\boldsymbol{z}^{(k)}\right) > \epsilon$ do $\boldsymbol{z}^{(k+1)} = \boldsymbol{z}^{(k)} - \Delta\left(\boldsymbol{z}^{(k)}\right)$ for all $k \ge k^*_{DP}$, where $k^*_{DP}$ is the
termination step in Step 1.
```

This algorithm can be viewed as a modification of the Newton's method and consists of two phases that help reducing the number of iterations. More specifically, Step 1, called the *damped phase*, guarantees that $f\left(\boldsymbol{z}^{(k)}\right) - f\left(\boldsymbol{z}^{(k+1)}\right) \ge \omega\left(\lambda^*\right)$ and in turn, the number of iterations in

Step 1, denoted by N_{DP}, is bounded with

$$N_{DP} \leq \frac{f\left(\boldsymbol{z}^{(0)}\right) - f\left(\boldsymbol{z}^{*}\right)}{\omega(\lambda^{*})}, \quad \text{where } \omega(\lambda) := \lambda - \log\left(1+\lambda\right) \text{ on } \Re_{+}.$$

This represents the advantage of Algorithm 1 as compared to relying only on the Newton's method, see Theorem 4.1.10 of Nesterov (2004) or Section 9.6.4 of Boyd and Vandenberghe (2004) for further details on this issue.[17] The total number of iterations in Step 2 is $\log_2 \log_2 (1/\epsilon)$ if an accuracy of $f\left(\boldsymbol{z}^{(k^{*})}\right) - f\left(\boldsymbol{z}^{*}\right) \leq \epsilon$ is sought. The latter bound is very small, e.g., 4.32 and 5.82 for $\epsilon = 10^{-6}$ and $\epsilon = 10^{-17}$, respectively. Note that $\epsilon = 10^{-17}$ is the **MATLAB** machine epsilon, which is the top end tolerance level benchmark in **MATLAB**.

Remark 4.3. *Inverting the Hessian is often challenging, and an alternative solution to computing the step/search direction, i.e computing* $\Delta\left(\boldsymbol{z}\right) := \left[\nabla^2 f\left(\boldsymbol{z}\right)\right]^{-1} \nabla f\left(\boldsymbol{z}\right)$ *for a given* $\boldsymbol{z}$*, is to solve* $\nabla^2 f\left(\boldsymbol{z}\right)\ \boldsymbol{t} = \nabla f\left(\boldsymbol{z}\right)$ *in* $\boldsymbol{t}$*, which is a linear system of equations. If we denote by* $t^{*}_{f}\left(\boldsymbol{z}\right)$ *the latter solution, we have* $\Delta\left(\boldsymbol{z}^{(k)}\right) = t^{*}_{f}\left(\boldsymbol{z}^{(k)}\right)$ *and*

$$\lambda_f\left(\boldsymbol{z}^{(k)}\right) = \sqrt{\nabla f\left(\boldsymbol{z}^{(k)}\right)^{\top}\left[\nabla^2 f\left(\boldsymbol{z}^{(k)}\right)\right]^{-1}\nabla f\left(\boldsymbol{z}^{(k)}\right)} = \sqrt{\nabla f\left(\boldsymbol{z}^{(k)}\right)^{\top}\ t^{*}_{f}\left(\boldsymbol{z}^{(k)}\right)}.$$

4.2. The ALM algorithm for the Inverse Gaussian Regression

We showed in Section 3.3 that the Inverse Gaussian Regression model is not proper for any *power* LF. However, it is still possible to create a tractable model for this parametric family for a particular *power* LF. Indeed, we assume a *reciprocal-square-root* LF (i.e. *power* LF from (2.7) with $\gamma = -2$) which satisfies Condition **C1** but not Condition **C2** of Definition 2.1. This choice leads to solving the following (non-linear) optimisation problem:

$$\min_{\boldsymbol{\beta}\in\Omega} \; f_{IG}\left(\boldsymbol{\beta}\right) = \sum_{i=1}^{n}\left(\frac{y_i}{2}\left(\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right)^{4} - \left(\boldsymbol{x}_i^{\top}\boldsymbol{\beta}\right)^{2}\right). \tag{4.5}$$

The advantage of using the *reciprocal-square-root* LF is that (4.5) has a tractable solution via the *Alternating Linearisation Method (ALM)*, see e.g. Boyd et al. (2011) for further details. More specifically, the variable β can be split into two variables, so that the *ALM* reformulation of (4.5) is given by:

$$\min_{(\boldsymbol{z},\boldsymbol{t})\in\Re^d\times\Re^d} \; G\left(\boldsymbol{z},\boldsymbol{t}\right) = \sum_{i=1}^{n}\left(\frac{y_i}{2}\left(\boldsymbol{x}_i^{\top}\boldsymbol{z}\right)^{2}\left(\boldsymbol{x}_i^{\top}\boldsymbol{t}\right)^{2} - \left(\boldsymbol{x}_i^{\top}\boldsymbol{z}\right)\left(\boldsymbol{x}_i^{\top}\boldsymbol{t}\right)\right) \quad \text{so that } \; \boldsymbol{z} = \boldsymbol{t}. \tag{4.6}$$

The iterative algorithm that efficiently solves (4.6) is given as Algorithm 2 and is an *Alternating Linearisation Method with backtracking (ALM-bktr)*, i.e. a bespoke *ALM* algorithm. This algorithm provides an approximation for $\boldsymbol{\beta}^{*}$, which denotes a local optimum of (4.5), by generating two sequences $\{\boldsymbol{z}_s : s \geq 0\}$ and $\{\boldsymbol{t}_s : s \geq 0\}$ such that $\boldsymbol{z}_s \to \boldsymbol{\beta}^{*}$ and/or $\boldsymbol{t}_s \to \boldsymbol{\beta}^{*}$. The main idea is to solve a two-block variant of (4.6), which is a *convex quadratic programming (QP)* instance in $\boldsymbol{z}$ for any given $\boldsymbol{t}$ that could be efficiently solved, and the same holds if $\boldsymbol{z}$ and $\boldsymbol{t}$ are interchanged.

[17] More formal convergence measures for Step 1 that are compared to the equivalent convergence measures of the standard Newton's method are available in Theorems 4.1.11 and 4.1.12 of Nesterov (2004).

The *ALM* algorithm relies on replacing the function G by their linearisation and an additional regularization factor in order to obtain an approximation to the initial objective function f_{IG} from (4.5). Thus, we define the following functions

$$\begin{aligned} H_1(\boldsymbol{z},\boldsymbol{t};\mu) &:= G(\boldsymbol{z},\boldsymbol{t}) + \langle G_2(\boldsymbol{t},\boldsymbol{t}), \boldsymbol{z}-\boldsymbol{t}\rangle + \frac{1}{2\mu}\|\boldsymbol{z}-\boldsymbol{t}\|_2^2, \\ H_2(\boldsymbol{z},\boldsymbol{t};\mu) &:= G(\boldsymbol{z},\boldsymbol{t}) + \langle G_1(\boldsymbol{z},\boldsymbol{z}), \boldsymbol{t}-\boldsymbol{z}\rangle + \frac{1}{2\mu}\|\boldsymbol{z}-\boldsymbol{t}\|_2^2, \end{aligned}$$

where $\|\cdot\|_2$ is the L^2 norm on $\Re^d$, μ is a positive constant, and G_1 and G_2 are the partial derivatives of G given below:

$$\begin{aligned} G_1(\boldsymbol{z},\boldsymbol{t}) &:= \frac{\partial G}{\partial \boldsymbol{z}} = \sum_{i=1}^{n}\left(y_i\left(\boldsymbol{x}_i^\top\boldsymbol{z}\right)\left(\boldsymbol{x}_i^\top\boldsymbol{t}\right)^2 - \left(\boldsymbol{x}_i^\top\boldsymbol{t}\right)\right)\boldsymbol{x}_i, \\ G_2(\boldsymbol{z},\boldsymbol{t}) &:= \frac{\partial G}{\partial \boldsymbol{t}} = \sum_{i=1}^{n}\left(y_i\left(\boldsymbol{x}_i^\top\boldsymbol{z}\right)^2\left(\boldsymbol{x}_i^\top\boldsymbol{t}\right) - \left(\boldsymbol{x}_i^\top\boldsymbol{z}\right)\right)\boldsymbol{x}_i. \end{aligned}$$

Algorithm 2 for solving (4.5), and therefore (4.6), is described below.[18]

5. Simulation Study

This section presents several numerical experiments to determine the efficiency and accuracy of the proposed algorithms, and investigates to what extent they can improve the standard built-in GLM libraries from various software. Specifically, we implement the *NSC* Algorithm 1 introduced in Section 4.1 for the Poisson (with *half-square-root* LF) and Gamma Regressions (with *half-reciprocal-square-root* LF), and the *ALM* Algorithm 2 introduced in Section 4.2 for solving Inverse Gaussian Regressions (with *reciprocal-square-root* LF).

For each specification of the number of observations n and number of covariates d, we synthetically construct N data generating processes (henceforth called DGP) and perform the above GLM estimations using both algorithms.[19] The effectiveness of our methods is determined by comparing our estimates with the "true" regression coefficients $\boldsymbol{\beta}_k$, for any $k = 1, \ldots, N$, obtained by using three standard built-in packages: **MATLAB** *fitglm*, **R** *glm*2 and **Python** statsmodels *sm.GLM* libraries.[20] To assess the accuracy of Algorithms 1 and 2 relative to these benchmarks we consider two performance indicators. First, we compute the Absolute

[18] The algorithm stops whenever $\sum \frac{|\boldsymbol{z}_{s+1}-\boldsymbol{t}_{s+1}|}{|\boldsymbol{z}_{s+1}|}$ reaches the user's defined value (e.g. the default value in our numerical examples is taken to be 10^{-4} to balance the speed and precision with other benchmark algorithms). Once the process is stopped, we use $\boldsymbol{z}_{s+1}$ (or $\boldsymbol{t}_{s+1}$) if H_1 is smaller (or larger) than H_2.

[19] Note that unlike in the theoretical presentation, d represents here the number of covariates excluding the trivial one corresponding to the intercept β_0, so that the full matrix of explanatory variables is obtained by adding the n-dimensional unit vector to $\boldsymbol{X}$. Details on the DGP simulation are illustrated in Appendix C.

[20] We remark that all three softwares rely on the Iteratively Reweighted Least Squares (IRLS) method to estimate the regression coefficients. Generally speaking, **R** *glm*2 provides an improvement over the standard **R** *glm* package by using the step-halving approach in order to improve the convergence properties of IRLS (see e.g. Marschner (2011)).

Algorithm 2: Standard ALM algorithm for solving (4.5)

```
Result: $(\boldsymbol{z}_{s^*}, \boldsymbol{t}_{s^*})$ that approximates $\boldsymbol{\beta}^*$, a local optimum of (4.5), where $s^*$ is the
        termination step.

Choose $\mu_{1,0} = \mu_{2,0} = \mu_0 > 0$, $b \in (0,1)$, and $\boldsymbol{z}_0 = \boldsymbol{t}_0 \in \Re^d$;
for $s \in \{0, 1, \dots\}$ do
    $\boldsymbol{z}_{s+1} := \arg\min_{\boldsymbol{z} \in \Re^d} \ H_1(\boldsymbol{z}, \boldsymbol{t}_s; \mu_{1,s})$;
    if $f_{IG}(\boldsymbol{z}_{s+1}) \leq H_1(\boldsymbol{z}_{s+1}, \boldsymbol{t}_s; \mu_{1,s})$ then
        choose $\mu_{1,s+1} \geq \mu_{1,s}$;
    else
        find the lowest $n_{1,s} \geq 1$ such that $f_{IG}(\boldsymbol{u}_{1,s}) \leq H_1(\boldsymbol{u}_{1,s}, \boldsymbol{t}_s; \mu^*_{1,s})$, where
         $\mu^*_{1,s} = \mu_{1,s} b^{n_{1,s}}$ and $\boldsymbol{u}_{1,s} := \arg\min_{\boldsymbol{z} \in \Re^d} \ H_1(\boldsymbol{z}, \boldsymbol{t}_s; \mu^*_{1,s})$;
        $\mu_{1,s+1} := \mu^*_{1,s}/b$ and $\boldsymbol{z}_{s+1} := \boldsymbol{u}_{1,s}$;
    end
    $\boldsymbol{t}_{s+1} := \arg\min_{\boldsymbol{t} \in \Re^d} \ H_2(\boldsymbol{z}_{s+1}, \boldsymbol{t}; \mu_{2,s})$;
    if $f_{IG}(\boldsymbol{t}_{s+1}) \leq H_2(\boldsymbol{z}_{s+1}, \boldsymbol{t}_{s+1}; \mu_{2,s})$ then
        choose $\mu_{2,s+1} \geq \mu_{2,s}$;
    else
        find the lowest $n_{2,s} \geq 1$ such that $f_{IG}(\boldsymbol{u}_{2,s}) \leq H_2(\boldsymbol{z}_{s+1}, \boldsymbol{u}_{2,s}; \mu^*_{2,s})$, where
         $\mu^*_{2,s} = \mu_{2,s} b^{n_{2,s}}$ and $\boldsymbol{u}_{2,s} := \arg\min_{\boldsymbol{t} \in \Re^d} \ H_2(\boldsymbol{z}_{s+1}, \boldsymbol{t}; \mu^*_{2,s})$;
        $\mu_{2,s+1} := \mu^*_{2,s}/b$ and $\boldsymbol{t}_{s+1} := \boldsymbol{u}_{2,s}$;
    end
end
```

Error Ratio (AER) and its mean ($MAER$), defined as:

$$MAER = \frac{1}{N} \sum_{k=1}^{N} AER_k \quad \text{with} \quad AER_k = \frac{AE\left(\hat{\boldsymbol{\beta}}_k^{alg}\right)}{AE\left(\hat{\boldsymbol{\beta}}_k^{benchmark}\right)}, \quad k = 1, \dots, N, \tag{5.1}$$

Here, the Absolute Error (AE) associated to each estimator $\hat{\boldsymbol{\beta}}_k$ is defined by the L^1-norm:

$$AE\left(\hat{\boldsymbol{\beta}}_k\right) = \sum_{j=1}^{d} |\hat{\beta}_{k,j} - \beta_{k,j}^{true}|, \tag{5.2}$$

where $\beta_{k,j}^{true}$ is the j^{th} component of the k^{th} simulated "true" regression coefficient according to the DGP scheme outlined in Appendix C, and $\hat{\beta}_{k,j}^{alg}$ and $\hat{\beta}_{k,j}^{benchmark}$ are their corresponding estimated values obtained with Algorithms 1 and 2, and the three software benchmark packages, respectively. The performance of our approach is further evaluated by computing the log-likelihood ratio statistics, which compare the GLM with the saturated model. Thus, we introduce below the Deviance Ratio (DR) and its mean (MDR):

$$MDR = \frac{1}{N} \sum_{k=1}^{N} DR_k \quad \text{with} \quad DR_k = \frac{D\left(\hat{\boldsymbol{\beta}}_k^{alg}\right)}{D\left(\hat{\boldsymbol{\beta}}_k^{benchmark}\right)}, \quad k = 1, \dots, N. \tag{5.3}$$

Here, the Deviance (D) of each GLM is defined by:

$$D\left(\hat{\boldsymbol{\beta}}_k\right) = -2\phi\left(\ell\left(\hat{\boldsymbol{\beta}}_k\right) - \ell_s\right), \tag{5.4}$$

where $\ell\left(\hat{\boldsymbol{\beta}}_k\right)$ is the log-likelihood function corresponding to the fitted GLM for the k^{th} simulated DGP scenario, while ℓ_s is the maximum value of the log-likelihood of the saturated model that is computed using the same function as in (2.4) with $\theta_i = b'^{-1}(y_i)$. Explicit expressions for the deviance of all GLMs considered in our numerical experiments are provided in Appendix D. Note that an $MAER$ or MDR value smaller than 1 indicates that our approach is more accurate on average than the benchmark with respect to the corresponding performance measure.

The efficiency of our algorithms relative to their benchmarks is also investigated by reporting the Mean Computational Time Ratio ($MCTR$) introduced as:

$$MCTR = \frac{1}{N}\sum_{k=1}^{N} CTR_k \quad \text{with} \quad CTR_k = \frac{CT\left(\hat{\boldsymbol{\beta}}_k^{alg}\right)}{CT\left(\hat{\boldsymbol{\beta}}_k^{benchmark}\right)}, \quad k = 1, \ldots, N. \tag{5.5}$$

Here, $CT\left(\hat{\boldsymbol{\beta}}_k^{alg1}\right)$ and $CT\left(\hat{\boldsymbol{\beta}}_k^{benchmark}\right)$ are the Algorithm 1 and benchmark computational times recorded for the k^{th} simulated DGP scenario, respectively. It follows that our algorithms are faster on average whenever $MCTR < 1$.[21] For a consistent and fair comparison of the computational time efficiency, all benchmarks have been implemented using their corresponding default starting values.and the same specifications in the optimisation procedure, i.e. maximum number of iterations $= 10,000$ and tolerance level $= 10^{-6}$. Since Algorithms 1 and 2 are coded in **MATLAB**, we use the **MATLAB** fitglm starting values for our estimations.

The performance indicators $MAER$ and MDR (both in **bold**), and $MCTR$ are computed based on $N = 500$ replicates. Note that Algorithms 1 and 2 always converge within a very reasonable number of iterations, which is not the case for the three benchmarks. Therefore, the number of replicates (out of 500 simulations) for which the optimisation problem (associated to the benchmarks) do not converge within the allocated maximum number of iterations is illustrated as #NaN in our tables. Consequently, these cases are discarded from the computation of our performance indicators so that the benchmarks' performance are computed in the most advantageous possible to those benchmarks.

Table 2 presents the results for the Poisson GLM regression. We first notice that in terms of accuracy, Algorithm 1 consistently outperforms both **MATLAB** *fitglm* and **Python** *sm.GLM* libraries for all cases considered. The improvements relative to **Python** *sm.GLM* are quite significant with respect to both $MAER$ and MDR with the largest augmentations being noticed for larger scale settings when the ratio between the sample size and the number of covariates/features decreases; for example, when $n/d = 5$, the improvements for both indicators are on average of around $15\%, 37\%$ and 53% for $n = 100, 500$ and $1,000$, respectively. The $MAER$ and MDR for the **MATLAB** *fitglm* benchmark are closer to 1, but unlike in the previous case, there are many scenarios when the *fitglm* MLE does not converge. This typically happens

[21] Note that for streamline purposes we only report the $MCTR$ values for Algorithm 1.

for the bigger scale problems, as it is the case when $n = 1,000$ and $d = 200$ (our largest setting) where convergence was not achieved in half of the cases. Unlike the **MATLAB** and **Python** libraries, **R** *glm*2 seems to perform very similarly to our Algorithm 1 for the Poisson GLM, the $MAER/MDR$ values being typically slightly above/below 1. The $MCTR$ values indicate that Algorithm 1 is always more efficient than both **Python** *sm.GLM* and **MATLAB** *fitglm*, with the largest improvements observed for small dimension settings. The smallest differences in runtime happen when $n = 1,000$ and $d = 50$, when our algorithm is five and seven times faster then the aforementioned benchmarks, respectively. However, while **R** *glm*2 is also slower when $n = 100$ then our Algorithm 1, it becomes more efficient for larger values of n.

Table 2: MAER, MCTR and MDR for Poisson GLM

		n = 100			n = 500			n = 1,000		
		d = 5	d = 10	d = 20	d = 25	d = 50	d = 100	d = 50	d = 100	d = 200
MATLAB	**MAER**	**0.9730**	**0.9620**	**0.9523**	**0.9685**	**0.9721**	**0.9713**	**0.9758**	**0.9782**	**0.9816**
fitglm	**MDR**	**0.9947**	**0.9935**	**0.9883**	**0.9977**	**0.9986**	**0.9970**	**0.9998**	**1.0002**	**1.0021**
	MCTR	0.0134	0.0169	0.0272	0.0630	0.0625	0.0762	0.1446	0.1012	0.1069
	#NaN	16	32	58	37	67	182	46	87	256
Python	**MAER**	**0.9393**	**0.8998**	**0.8431**	**0.9002**	**0.8463**	**0.6227**	**0.8838**	**0.8131**	**0.4723**
sm.GLM	**MDR**	**0.9177**	**0.8972**	**0.8518**	**0.9093**	**0.8553**	**0.6268**	**0.8915**	**0.8166**	**0.4721**
	MCTR	0.0065	0.0082	0.0129	0.0551	0.0531	0.0340	0.2016	0.1022	0.0531
	#NaN	0	0	0	0	0	0	0	0	0
R	**MAER**	**0.9999**	**0.9967**	**1.0014**	**1.0082**	**1.0085**	**1.0161**	**1.0087**	**1.0168**	**1.0376**
glm2	**MDR**	**0.9579**	**0.9708**	**0.9858**	**0.9832**	**0.9882**	**0.9950**	**0.9870**	**0.9911**	**1.0057**
	MCTR	0.2553	0.2815	0.5043	1.5819	1.5695	1.0513	3.3093	2.0093	1.3328
	#NaN	0	0	0	0	0	0	0	0	0

Notes: This table reports the **Mean Absolute Error Ratio** ($MAER$), **Mean Deviance Ratio** (MDR) and Mean Computational Time Ratio ($MCTR$) of Algorithm 1 from Section 4.1 relative to its benchmarks, **MATLAB** *fitglm*, **Python** *sm.GLM* and **R** *glm*2, for the Poisson GLM equipped with the *half-power* LF from (2.8) with $\gamma = 2$. These indicators are computed based on the MLE values obtained from $N = 500$ simulations according to the DGP scheme outlined in Appendix C, for different specifications for the number of observations n and number of covariates d. The number of replicates (out of 500 simulations) that the benchmarks cannot converge is shown as #NaN. All benchmarks are implemented using the same starting values with a maximum of 10,000 iterations and 10^{-6} tolerance level.

The Gamma GLM results are illustrated in Table 3. First we notice that Algorithm 1 consistently outperforms all benchmarks in terms of both accuracy and efficiency. Unlike in the Poisson case, our method performs significantly better than **R** *glm*2 with respect to both accuracy indicators, with an average improvement ranging from $40\% - 77\%$ and $40\% - 68\%$ for $MAER$ and MDR, respectively, when $n = 1,000$. We further notice a reverse situation regarding the **MATLAB** *fitglm* and **Python** *sm.glm* GLM libraries when comparing to the results from Table 2. Specifically, on the one hand, the MLE procedure from **Python** *sm.glm* does not converge in many instances, but when it converges, the estimates are very close to those obtained via Algorithm 1. On the other hand, despite always converging, the **MATLAB** *fitglm* optimisation produces $MAER$ and MDR values which are significantly lower than 1, with the lowest values recorded when $n = 1,000$. The reported average computational times favours again our methodology; the only $MCTR$ values greater than 1 are spotted for the larger scale settings for **R** $glm2$, which provided inaccurate estimates in all these cases.

In summary, based on our DGP for Poisson and Gamma GLMs, we can argue that overall, our Algorithm 1 provides the most accurate and efficient estimation approach relative to the three benchmarks, while **R** $glm2$ is the second best, generally speaking being more stable than the **Python** $sm.GLM$ and **MATLAB** $fitglm$ counterparts.[22]

Table 3: MAER, MCTR and MDR for Gamma GLM

		n = 100			n = 500			n = 1,000		
		d = 5	d = 10	d = 20	d = 25	d = 50	d = 100	d = 50	d = 100	d = 200
MATLAB	**MAER**	**0.9216**	**0.9449**	**0.9722**	**0.6554**	**0.7141**	**0.8469**	**0.5547**	**0.5734**	**0.7167**
$fitglm$	**MDR**	**0.9534**	**0.9511**	**0.9687**	**0.6713**	**0.6753**	**0.8061**	**0.5065**	**0.4424**	**0.6202**
	MCTR	0.0579	0.0270	0.0404	0.2549	0.1142	0.0995	0.5530	0.1991	0.1954
	#NaN	0	0	0	0	0	0	0	0	0
Python	**MAER**	**0.9831**	**0.9930**	**0.9989**	**0.9962**	**0.9999**	**1.0000**	**0.9932**	**1.0000**	**1.0000**
$sm.GLM$	**MDR**	**0.9953**	**0.9980**	**1.0000**	**0.9998**	**1.0000**	**1.0000**	**0.9997**	**1.0000**	**1.0000**
	MCTR	0.0700	0.2049	0.2314	1.6705	0.9847	0.5505	3.7401	2.0635	0.8492
	#NaN	78	55	21	406	268	124	471	373	206
R	**MAER**	**0.9450**	**0.9608**	**0.9850**	**0.5843**	**0.7216**	**0.8928**	**0.4018**	**0.5434**	**0.7679**
$glm2$	**MDR**	**0.9496**	**0.9621**	**0.9878**	**0.5887**	**0.6859**	**0.8585**	**0.3944**	**0.4643**	**0.6840**
	MCTR	0.2945	0.5550	0.5101	6.5451	3.4493	1.6073	12.2574	5.1892	1.5737
	#NaN	0	0	0	0	0	0	0	0	0

Notes: This table reports the **Mean Absolute Error Ratio** ($MAER$), **Mean Deviance Ratio** (MDR) and Mean Computational Time Ratio ($MCTR$) of Algorithm 1 from Section 4.1 relative to its benchmarks, **MATLAB** $fitglm$, **Python** $sm.GLM$ and **R** $glm2$, for the Gamma GLM equipped with the *half-power* LF from (2.8) with $\gamma = -2$. These indicators are computed based on the MLE values obtained from $N = 500$ simulations according to the DGP scheme outlined in Appendix C, for different specifications for the number of observations n and number of covariates d. The number of replicates (out of 500 simulations) that the benchmarks cannot converge is shown as #NaN. All benchmarks are implemented using the same starting values with a maximum of $10,000$ iterations and 10^{-6} tolerance level.

We next turn our attention to the implementation results of the ALM Algorithm 2 for solving Inverse Gaussian Regressions based on the *reciprocal-square-root* LF. The benchmark chosen in our analysis is the **MATLAB** $fitglm$ package and we only focus on the accuracy of our methodology. Figure 2 illustrates the box plots of the **MATLAB** $fitglm$-based AER and DR for the same values of n and d as in the previous tables. First, we notice (in all nine cases) that the AER indicators are more or less symmetrically distributed around 1, with a median value smaller (but closer) to 1, suggesting that our Algorithm 2 slightly outperforms **MATLAB** $fitglm$ relative to this performance measure. However, our method performs much better in terms of the deviance measure, as almost all DR values are below 1, with the most significant differences being documented for larger dimension problems and the smallest n/d ratio (i.e. $n/d = 5$). Furthermore, for each value of n, we notice a decreasing trend in the median of

[22] Note that these conclusions are drawn solely based on our DGP and a limited number of experiments, so further implementations may be needed to further investigate this problem.

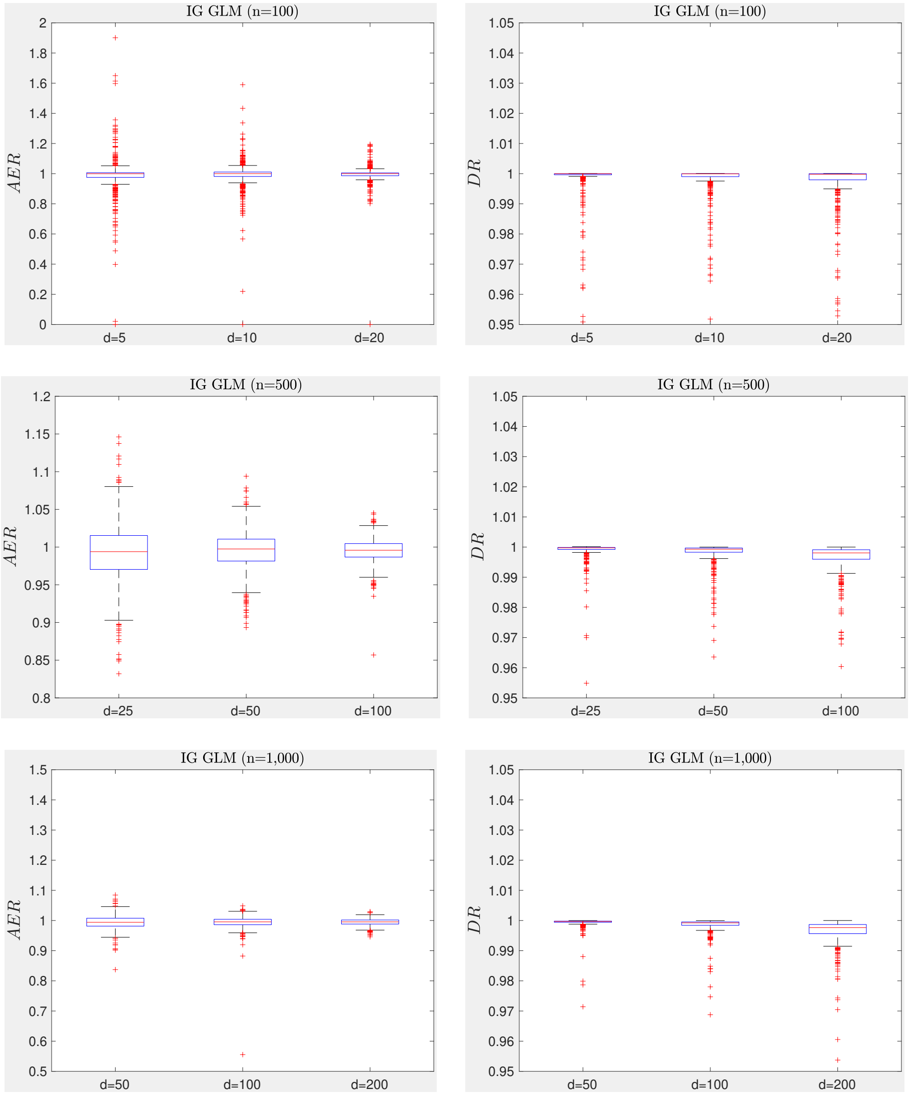

Figure 2: Absolute Error Ratio (AER) and Deviance Ratio (DR) for Inverse Gaussian GLM based on MATLAB *fitglm*.

Notes: This figure shows the box plots of Absolute Error Ratio (AER) in left panel and Deviance Ratio (DR) in right panel of Algorithm 2 from Section 4.2 relative to the **MATLAB** *fitglm* benchmark for the Inverse Gaussian GLM based on the *reciprocal-square-root* LF. Each box plot is constructed using AERs and DRs computed based on MLE values obtained from $N = 500$ simulations according to the DGP scheme outlined in Appendix C, for different specifications for the number of observations n and the number of covariates d. All implementations use the same starting value with a maximum of $10,000$ iterations and 10^{-6} tolerance level.

DRs as the number of covariates increases. These observations are consistent with the previous findings on Algorithm 1 regarding the significant improvements in accuracy for bigger datasets.

6. Real Data Analysis

This section replicates some of the numerical illustrations in Section 5 for two well-known insurance datasets. Specifically, we compare the in-sample and out-of-sample performances of our Algorithm 1 in Section 4.1 relative to its benchmarks, **MATLAB** *fitglm*, **Python** *sm.GLM* and **R** *glm*2. Algorithm 1 is implemented for Gamma GLM equipped with the *half-power* LF in (2.8) with $\gamma = -2$, while the benchmarks are implemented for Gamma GLM equipped with the *half-power* LF in (2.8) with $\gamma = -2$ or *log* LF. The results from this section refer to benchmark GLM implementations with *log* LF; e.g., *R* deployments are signified as "*R_log*". Similarly, GLM benchmark implementations with *half-power* LF are, e.g., signified as "*MATLAB_hp*" for *MATLAB* deployments.

Note that our comparisons (related to the two datasets) are summarised in Tables 4 and 5, where MSE ratios between Algorithm 1 and the other six benchmarks are provided. Values greater than 1 signify that Algorithm 1 outperforms the competitor that it is compared to; clearly, the larger the ratio (and greater than 1) it is, the "better" performance for Algorithm 1 is observed.

The first data analysis is based on a synthetic health insurance dataset, which mimics the characteristics of the US insurance market and census data. This Kaggle dataset is commonly used by the Machine Learning research community.[23] The dataset contains 1,338 data points regarding i) the dependent variable represented by medical costs and ii) six independent variables (*Age, Sex, BMI, Number of Children, Smoker* and *Region*). One-hot-encoding is implemented for two covariates, namely, *Number of Children* and *Region*.

We randomly split the health dataset into 70% training data and 30% testing data for 100 times in order to understand which GLM method consistently performs "best" in terms of MSE. For each split, we use the training set to estimate the model's parameters and make predictions over the testing set. The results are summarised in Table 4, and amongst 100 replicates, our Algorithm 1 performs very well and outperforms all other methods in all 100 splits. For each set of 100 splits, 100 MSE ratios (Algorithm 1 vs a given benchmark) are computed for which their average and three quantiles are computed and displayed in Table 4. It should be noted that all seven GLM implementations converge across the 100 replicates. Table 4 shows that the *log* LF is unacceptable for this data, while the *half-power* LF shows reasonable performance. Further, Algorithm 1 shows (on average) a MSE seven times lower than any of the three Gamma GLM implementations with *half-power* LF.

Table 4: MSE Ratios for Gamma GLM

	Average	**25% quantile**	**50% quantile**	**75% quantile**
R_log	1281.9688	792.1623	1134.2212	1613.5390
R_hp	**7.2525**	**3.5435**	**5.2979**	**8.6959**
MATLAB_log	1275.7212	794.2883	1118.9052	1615.7908
MATLAB_hp	**7.2505**	**3.5440**	**5.2948**	**8.7025**
Python_log	1309.4299	794.2882	1142.5632	1676.1026
Python_hp	**7.2505**	**3.5440**	**5.2948**	**8.7025**

Notes: This table compares the **MSE** for Gamma GLM implementations of Algorithm 1 in Section 4.1 (with the *half-power* LF in (2.8) with $\gamma = -2$) to its six benchmarks (with *half-power* LF in (2.8) with $\gamma = -2$ and *log* LF). All implementations use the packages' default starting value with a maximum of 10,000 iterations and 10^{-6} tolerance level. The starting values for Algorithm 1 are the same as those in **MATLAB** *fitglm*.

The second dataset is a rich flood insurance data that accounts for flood related claim payments

[23] "US Health Insurance" dataset is available at github.com/stedy/Machine-Learning-with-R-datasets or www.kaggle.com/datasets/teertha/ushealthinsurancedataset.

across the 50 US states[24]. The data include National Flood Insurance Program (NFIP) claim transactions, which are derived from the NFIP system but the claims details are redacted to protect the policyholders' personally identifiable information. We extract the flood claims from three US states, namely, Texas, Louisiana and California which are ranked the highest for the potential of losses triggered by riverine flooding events.

To test and compare the Gamma GLM prediction methods, we extract data about one dependent variable (coded as "amountPaidOnBuildingClaim") and 16 covariates/features; these are coded as "totalBuildingInsuranceCoverage", "buildingPropertyValue", "elevatedBuildingIndicator", "numberOfFloorsInTheInsuredBuilding", "originalConstructionYear", "primaryResidenceIndicator", "waterDepth", "buildingDeductibleCode", "causeOfDamage", "occupancyType", "rateMethod", "countyCode", "crsClassificationCode", "floodEvent", "floodZoneCurrent" and "ratedFloodZone". Amongst these features, all categorical variables are one-hot encoded.

We predict the flood claims in these three states from year 2011 to 2023. We rely on the previous year data (e.g., year 2010) as the training set to estimate the parameters and predict the claims in the next year (as the testing set); we repeat the same process to predict the claims from 2011 to 2023. Table 5 contains the out-of-sample performances across the three US states over the 13 years. The overall conclusion is favourable to Algorithm 1 that outperforms all its six competitors. Recall that the *log* LF fails to show good predictions in Table 4, which is no longer true for the flood data where both LFs show relatively good results though once again, *half-power* often outperforms its equivalent implementation with *log* LF. Further, when counting the "best" implementations (as listed in the bottom panel), our Algorithm 1 shows the highest number of best performance with 11 "wins" (out of 39 instances) if comparisons are made with all six competitors; Algorithm 1 shows the highest number of best performance with 15 "wins" (out of 39 instances) if comparisons are made with the three GLM implementations equipped with *half-power* LF.

It is very interesting that Table 5 matches the lack of convergence pattern observed in Table 3. That is, "$Python\_hp$" is more likely to not converge, but when it converges, it has reasonable performance but it does not (overall) outperform Algorithm 1. "$Python\_hp$" does not converge in 12 (out of 39) instances, while "$MATLAB\_log$" does not converge in one (out of 39) instance; as before, lack of convergence is indicated as "NA". Once again, our Algorithm 1 converges in all 39 instances. The overall picture is that Algorithm 1 shows very good performance when comparing to its six competitors.

[24] "OpenFEMA" dataset is available at www.fema.gov/openfema-data-page/fima-nfip-redacted-claims-v2.

Table 5: Ratios of MSEs for Gamma GLM (Out-of-sample Performance)

year	"best" method	R_log	R_hp	MATLAB_log	MATLAB_hp	Python_log	Python_hp
				Texas			
2011	MATLAB_hp	7.46E+01	9.42E-01	7.14E+01	9.12E-01	7.14E+01	NA
2012	**Alg 1**	1.21E+24	1.94E+01	4.67E+23	6.96E+00	3.22E+08	6.96E+00
2013	**Alg 1**	8.18E+16	5.51E+07	2.48E+17	1.31E+02	2.48E+17	NA
2014	Python_hp	2.18E-04	1.58E+02	2.18E-04	2.27E-04	2.18E-04	2.15E-04
2015	R_hp	5.34E+15	6.33E-01	6.02E+15	9.97E-01	6.02E+15	1.03E+00
2016	R_hp	5.48E+18	9.79E-01	2.92E+20	9.80E-01	2.92E+20	NA
2017	**Alg 1**	1.70E+180	1.01E+00	5.04E+185	6.36E+00	5.04E+185	NA
2018	MATLAB_hp	8.89E+06	1.01E+00	2.80E+07	9.98E-01	2.80E+07	NA
2019	R_log	5.19E-04	2.82E+00	5.22E-04	2.48E+05	5.22E-04	NA
2020	**Alg 1**	7.21E+19	1.08E+00	1.08E+20	2.58E+00	1.08E+20	NA
2021	R_hp	2.79E+03	9.88E-01	2.80E+03	1.00E+00	2.80E+03	1.00E+00
2022	R_log	6.24E-01	8.62E-01	9.67E-01	1.04E+00	8.45E-01	1.03E+06
2023	R_log	7.74E-01	8.95E-01	7.83E-01	9.04E-01	7.83E-01	9.04E-01
				Louisiana			
2011	Python_hp	4.12E+49	1.02E+00	4.00E+49	1.00E+00	4.00E+49	1.00E+00
2012	**Alg 1**	8.96E+03	4.27E+00	7.65E+03	4.38E+00	7.65E+03	4.38E+00
2013	R_hp	4.85E-05	4.05E-05	NA	1.58E+00	4.88E-05	NA
2014	R_log	9.49E-01	9.68E-01	9.50E-01	9.68E-01	9.50E-01	9.68E-01
2015	R_hp	1.36E+13	9.67E-01	1.36E+13	1.00E+00	1.36E+13	1.00E+00
2016	MATLAB_hp	3.15E+66	1.00E+00	1.96E+66	1.00E+00	1.96E+66	1.00E+00
2017	**Alg 1**	4.79E+11	1.19E+00	1.99E+18	2.68E+00	1.97E+18	NA
2018	MATLAB_log	9.81E-01	1.01E+00	9.81E-01	4.78E+00	9.81E-01	9.86E-01
2019	R_hp	3.66E+56	7.24E-01	3.13E+61	1.00E+00	1.51E+59	1.00E+00
2020	**Alg 1**	1.02E+00	1.01E+00	1.02E+00	1.01E+00	1.02E+00	1.01E+00
2021	Python_hp	4.17E+00	9.82E-01	5.90E+00	2.13E+00	5.90E+00	1.19E-01
2022	**Alg 1**	1.30E+00	1.05E+00	1.31E+00	1.01E+00	1.21E+00	NA
2023	Python_log	1.73E-01	7.58E+01	1.86E-01	7.66E+01	1.73E-01	7.62E+01
				California			
2011	R_log	2.40E-02	1.08E+00	2.41E-02	1.03E+00	2.41E-02	1.03E+00
2012	R_hp	9.49E+07	9.92E-01	9.36E+07	1.00E+00	9.36E+07	1.00E+00
2013	MATLAB_log	9.69E-01	1.02E+00	9.69E-01	1.02E+00	9.69E-01	1.02E+00
2014	Python_hp	7.84E+37	1.08E-03	1.64E+38	1.47E-04	4.86E+36	1.46E-04
2015	Python_log	5.37E-03	9.53E-03	5.36E-03	9.52E-03	5.36E-03	9.52E-03
2016	R_hp	1.15E+00	9.99E-01	1.15E+00	1.00E+00	1.15E+00	1.00E+00
2017	MATLAB_hp	8.75E+16	1.07E+00	9.97E+16	1.00E+00	9.97E+16	1.00E+00
2018	**Alg 1**	1.13E+00	1.00E+00	1.15E+00	1.00E+00	1.15E+00	NA
2019	Python_hp	1.04E+10	2.73E+00	1.03E+10	2.75E+00	1.03E+10	1.00E+00
2020	R_hp	5.48E+01	7.84E-01	5.69E+01	9.14E+09	5.64E+01	NA
2021	**Alg 1**	1.79E+14	1.20E+00	2.65E+13	1.00E+00	1.52E+10	1.00E+00
2022	**Alg 1**	3.96E+02	1.03E+00	4.00E+02	1.03E+00	4.31E+02	1.02E+00
2023	MATLAB_hp	3.98E+00	1.06E+00	3.97E+00	1.00E+00	3.97E+00	1.00E+00
				Summary			
	Averages	4.37E+178	1.41E+06	1.33E+184	2.34E+08	1.29E+184	3.83E+04
	25% quantile	9.75E-01	9.68E-01	9.89E-01	1.00E+00	9.75E-01	9.93E-01
	50% quantile	3.96E+02	1.01E+00	1.60E+03	1.00E+00	4.31E+02	1.00E+00
	75% quantile	4.36E+16	1.08E+00	2.11E+17	2.63E+00	5.28E+16	1.02E+00
				Counting the "best" implementation (all)			
	Alg 1	R_log	R_hp	MATLAB_log	MATLAB_hp	Python_log	Python_hp
2011-2023	11	5	9	2	5	2	5
				Counting the "best" implementation (half-power only)			
	Alg 1	R_log	R_hp	MATLAB_log	MATLAB_hp	Python_log	Python_hp
2011-2023	15	na	11	na	5	na	8

Notes: This table compares the **MSE** for Gamma GLM implementations of Algorithm 1 from Section 4.1 (with the *half-power* LF from (2.8) with $\gamma = -2$) to its six benchmarks (with the *half-power* LF from (2.8) with $\gamma = -2$ and the *log* LF). All implementations use the package's default starting values with a maximum of $10,000$ iterations and 10^{-6} tolerance level. The starting values for Algorithm 1 are the same as those in **MATLAB** *fitglm*. The top three panels refer to 13 out-of-sample performances (between 2011 and 2023) related to the three selected US states. An "NA" entry means that the corresponding method is unable to converge; note that Algorithm 1 converges in all cases. The average and three quantiles corresponding to each set of 39 ratios from each column are shown in the Summary panel. The bottom panel counts the "best" implementation of each method across all settings.

7. Conclusions

This paper makes two important contributions to the GLM literature. First, we provide a general characterisation of proper GLMs for various exponential dispersion models, including the Tweedie family. The main finding is that although most Tweedie GLMs are not proper for *canonical* and *log* LFs, a rich class of proper Tweedie GLMs can be identified for *power* LFs. Second, we propose specialized optimisation algorithms for implementing several instances of Tweedie GLMs under *power* LFs. These algorithms outperform standard methods in terms of accuracy and efficiency, particularly in high-dimensional scenarios, as demonstrated via a thorough comparison with existing libraries like **MATLAB** *fitglm*, **R** *glm2*, and **Python** *sm.GLM.*

References

Antonio, K. and J. Beirlant (2007). Actuarial statistics with generalized linear mixed models. *Insurance: Mathematics and Economics 40*(1), 58–76.

Bach, F. (2010). Self-concordant analysis for logistic regression. *Electronic Journal of Statistics 4*, 384 – 414.

Bickel, P. J. and K. A. Doksum (2015). *Mathematical Statistics: Basic Ideas and Selected Topics, Volume I.* Second ed., Chapman and Hall/CRC.

Boyd, S. and L. Vandenberghe (2004). *Convex optimization.* Cambridge university press.

Boyd, S. P., N. Parikh, E. Chu, B. Peleato, and J. Eckstein (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning 3*(1), 1–122.

Debón, A., F. Montes, and F. Puig (2008). Modelling and forecasting mortality in spain. *European Journal of Operational Research 189*(3), 624–637.

Delong, L., M. Lindholm, and M. V. Wüthrich (2021). Making tweedie's compound poisson model more accessible. *European Actuarial Journal 11*(1), 185–226.

Eling, M. and J. Wirfs (2019). What are the actual costs of cyber risk events? *European Journal of Operational Research 272*(3), 1109–1119.

Hastie, T., R. Tibshirani, and J. Friedman (2001). *The Elements of Statistical Learning.* Springer Series in Statistics. New York, NY, USA: Springer New York Inc.

Jørgensen, B. (1987). Exponential dispersion models. *Journal of the Royal Statistical Society, Series B (Methodology) 49*(2), 127–145.

Kuo, L. and B. Mallick (1998). Variable selection for regression models. *Sankhyā: The Indian Journal of Statistics, Series B 60*(1), 65–81.

Mäkeläinen, T., K. Schmidt, and G. Styan (1981). On the existence and uniqueness of the maximum likelihood estimate of a vector-valued parameter in fixed-size samples. *Annals of Statistics 9*(4), 758–567.

Marschner, I. C. (2011). Glm2: Fitting Generalized Linear Models with Convergence Problems. *The R Journal 3*(2), 12–15.

McCullagh, P., J. Nelder, and R. Wedderburn (1989). *Generalized Linear Models.* Second ed., Chapman and Hall/CRC.

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

Nesterov, Y. E. (2004). *Introductory Lectures on Convex Optimization - A Basic Course*, Volume 87 of *Applied Optimization.* Springer.

Wedderburn, R. (1976). On the existence and uniqueness of the maximum likelihood estimates for certain generalized linear models. *Biometrika 63*(1), 27–32.

Appendix A. Proofs

Appendix A.1. Proof of Propositions 3.1-3.3

The proofs follow easily by verifying the conditions in Definition 2.1 for the Poisson, Gamma and Inverse Gaussian families, respectively.

Appendix A.2. Proof of Theorem 3.4

The identification of the three classes of Tweedie GLM that are well-defined is not difficult, and thus, we only outline some arguments without further details that are quite obvious. Clearly, $b'(\theta) = \big(\theta/(\alpha-1)\big)^{\alpha-1}$ for all $\theta \in \Re$. Since $\alpha < 2$, then setting a) is readily true and we require $\alpha \in (1,2)$, which is equivalent to $p < 0$, whenever $\Theta \in \{\Re_+^*, \Re_+\}$. Setting b) is the mirror case of setting a), and the proof is very similar. Settings c) and d) are similar to the previous ones, and the analysis depends if $\alpha - 1$ is an odd or even negative integer.

Appendix A.3. Proof of Theorem 3.5

First, we investigate parts (i) and (iii) (the *negative power* LF case) together, and therefore assume only *power* or *negative power* LFs. Condition **C2** requires

$$y(\alpha-1)\big(h(\eta)\big)^{\frac{1}{\alpha-1}} - \frac{\alpha-1}{\alpha}\big(h(\eta)\big)^{\frac{\alpha}{\alpha-1}} \quad \text{to be concave in } \eta \text{ on } \Re \text{ for all } y \in \mathcal{Y}. \tag{A.1}$$

Setting a) is first justified, but only for *power* LFs since the image of h is $\Re_+^*$, and in turn, $\gamma = 2k, k \in \mathbb{Z}^*$. Denote $a_1 = y\,(\alpha-1)$, $a_2 = \frac{1-\alpha}{\alpha}$ and $\gamma' = \frac{1}{\alpha-1}$. Equation (A.1) is equivalent to

$$\xi\,(\eta; y) := a_1\eta^{\gamma\gamma'} + a_2\eta^{\gamma(\gamma'+1)} \quad \text{is concave in } \eta \text{ on } \Re \text{ for all } y \in \mathcal{Y}. \tag{A.2}$$

Note that $\mathcal{Y} = \Re_+^*$ is assumed. Since $1 < \alpha < 2$ and $y > 0$, then $a_1 > 0$ and $a_2 < 0$, and in turn, (A.2) holds if and only if $\gamma\gamma' \in [0,1]$ and $\gamma(\gamma'+1) \notin (0,1)$. This is equivalent to having $\gamma \geq 0$, $\gamma\gamma' \leq 1$ and $\gamma(\gamma'+1) \geq 1$, since $\gamma' > 1$ in this case, which is further equivalent to $\frac{\alpha-1}{\alpha} \leq \gamma \leq \alpha - 1$. The latter cannot hold since $\alpha - 1 \in (0,1)$, $\alpha - 1 - \frac{\alpha-1}{\alpha} \in (0, 1/2)$ and $\gamma = 2k, k \in \mathbb{Z}^*$, which concludes that no proper GLM model is possible for setting a).

Setting b) is now justified, but only for *power* LFs since the image of h is $\Re_+^*$, and thus, $\gamma = 2k, k \in \mathbb{Z}^*$. We split this in two subcases, setting b1) and setting b2) for $0 < \alpha < 1$ and $\alpha < 0$, respectively.

Setting b1) holds if and only if $\gamma\gamma' \notin (0,1)$ and $\gamma(\gamma'+1) \in [0,1]$, since $a_1 < 0$ and $a_2 > 0$, which is equivalent to having $\gamma \leq 0$, $\gamma\gamma' \geq 1$ and $\gamma(\gamma'+1) \geq 1$ as $\gamma' < -1$, and in turn, $\frac{\alpha-1}{\alpha} \leq \gamma \leq \alpha - 1$. The later is true if and only if $\gamma = -2k$ for any $k \in \mathbb{N}^*$ and $(1-\gamma)\alpha \leq 1$ since $0 < \alpha < 1$, which concludes setting b1).

Setting b2) implies that $a_1, a_2 < 0$ and $\gamma' \in (-1, 0)$. Therefore, setting b2) holds if and only if $\gamma\gamma' \notin (0,1)$ and $\gamma(\gamma'+1) \notin (0,1)$, which is equivalent to having $\gamma \geq 0$ and $\gamma(\gamma'+1) \geq 1$ or $\gamma \leq 0$ and $\gamma\gamma' \geq 1$, and in turn, $\frac{\alpha-1}{\alpha} \leq \gamma$ or $\gamma \leq \alpha - 1$ must hold, which concludes setting b2).

Setting c) is similar to setting b2), and we thus skip its proof. Setting d) requires for *power* and *negative power* LFs to having $\gamma' \in \mathbb{Z}$ so that the likelihood function is well-defined in (2.4) (and thus, in (A.1)), but also γ to be an odd integer so that the image of h is $\Re^*$. These do not hold since $\gamma' \in (-1, 0)$, which justifies our claim for setting d). This concludes parts (i) and (iii) (the *negative power* LF case).

The proof of parts (ii) and (iii) (the *negative half-power* LF case) follows in a similar way, with one small difference. That is, *half-power* LFs require $\gamma \in \Re^*$ instead of $\gamma = 2k, k \in \mathbb{Z}^*$, but everything else does not significantly change. For these reasons, we do not provide additional details on this proof.

Appendix A.4. Proof of Theorem 3.6

We first show part (i), and assume *canonical* LFs. Note first $h(\eta) = b'(\eta) = \big(\eta/(\alpha-1)\big)^{\alpha-1}$, which implies that $\alpha \in \mathbb{Z}\backslash\{1\}$. This implies that amongst settings a)–c), only setting b2), which was introduced in Appendix A.3, might hold while all other settings are clearly infeasible. The image of h is $\Re_+^*$ and therefore, α is an odd negative integer, which is a *power* LF with an odd parameter γ. This contradicts our findings in the proof of part (i) from Theorem 3.4 for setting b2), and concludes that no *canonical* LF leads to proper GLM in settings a)–c). Setting d) requires α to be an even negative integer and $\gamma' \in \mathbb{Z}$ as explained in the previous proof, which are infeasible conditions. Thus, no *canonical* LF leads to proper GLM in setting d). This concludes part (i).

We now show part (ii) and assume *log* LFs. Using the same notations as in Appendix A.3, Equation (A.1) is equivalent to

$$\xi\left(\eta; y\right) := a_1 e^{\eta\gamma'} + a_2 e^{\eta(\gamma'+1)} \quad \text{is concave in } \eta \text{ on } \Re \text{ for all } y \in \mathcal{Y}, \tag{A.3}$$

which requires $a_1, a_2 \leq 0$ due to the convexity property of $e^{\eta\gamma}$ in η on $\Re$, for any $\gamma \in \Re$. The latter explains that only setting b2) is feasible amongst settings a)–c). Setting d) is infeasible since the the image of h is $\Re^*$, which is impossible for a *log* LF. The proof is now complete.

Appendix A.5. Proof of Theorem 4.2

We proceed by showing part a), but only for (4.1), since (4.2) could be argued similarly. Let

$$f_{i,P}\left(\boldsymbol{\beta}\right) = \left(\frac{1}{2}\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)^2 - y_i \log\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)\right) \quad \text{for all } 1 \leq i \leq n, \tag{A.4}$$

so that $f_P(\boldsymbol{\beta}) = \sum_{i=1}^{n} f_{i,P}(\boldsymbol{\beta})$. First, we show that f_P is a closed convex function on Ω. From (A.4), $f_{i,P}$ is convex (and therefore, continuous) on Ω, and since $dom\,(f_P) = \Omega$ is an open set and $\lim_{\boldsymbol{\beta}\to\boldsymbol{\beta}_0} f_{i,P}(\boldsymbol{\beta}) = \infty$ for all $\boldsymbol{\beta}_0 \in \partial\, dom\,(f_P)$, it follows that $f_{i,P}$ is closed convex on Ω. The closed convex property of f_P follows from the fact that it is a sum of closed convex functions. We next prove that f_P is self-concordant on Ω. For any $t \in \Re$, $\boldsymbol{u} \in \Omega$ and $\boldsymbol{v} \in \Re^d$, such that $\boldsymbol{u} + t\boldsymbol{v} \in \Omega$, we define the function $g_{i,P}(t) = f_{i,P}(\boldsymbol{u} + t\boldsymbol{v})$, or any $i = 1, \ldots, n$, and let $g_P(t) = \sum_{i=1}^{n} g_{i,P}(t)$. Next, we show that

$$\left|g'''_{i,P}(t)\right| \le 2\left(g''_{i,P}(t)\right)^{3/2}. \tag{A.5}$$

Note that

$$g''_{i,P}(t) = \left(\boldsymbol{x}_i^\top \boldsymbol{v}\right)^2 + \frac{y_i\left(\boldsymbol{x}_i^\top \boldsymbol{v}\right)^2}{\left(\boldsymbol{x}_i^\top \boldsymbol{u} + t\,\boldsymbol{x}_i^\top \boldsymbol{v}\right)^2} \quad \text{and} \quad g'''_{i,P}(t) = -\frac{2y_i\left(\boldsymbol{x}_i^\top \boldsymbol{v}\right)^3}{\left(\boldsymbol{x}_i^\top \boldsymbol{u} + t\,\boldsymbol{x}_i^\top \boldsymbol{v}\right)^3}.$$

Clearly, (A.5) holds whenever $\boldsymbol{x}_i^\top \boldsymbol{v} = 0$, and thus, we further assume that $\boldsymbol{x}_i^\top \boldsymbol{v} \ne 0$. Now,

$$\left|g'''_{i,P}(t)\right|\left(g''_{i,P}(t)\right)^{-3/2} = 2y_i\left(y_i + \left(\boldsymbol{x}_i^\top \boldsymbol{u} + t\,\boldsymbol{x}_i^\top \boldsymbol{v}\right)^2\right)^{-3/2} \le 2,$$

since $y_i \le y_i^{3/2} \le (y_i + \epsilon_i)^{3/2}$ for any non-negative integer y_i and any $\epsilon_i \ge 0$ (recall that $y_i \in \mathbb{N}$ as the sampling distribution is Poisson). The self-concordant property of f_P follows from

$$\left|g'''_P(t)\right| = \left|\sum_{i=1}^{n} g'''_{i,P}(t)\right| \le \sum_{i=1}^{n}\left|g'''_{i,P}(t)\right| \le 2\sum_{i=1}^{n}\left(g''_{i,P}(t)\right)^{3/2} \le 2\left(\sum_{i=1}^{n} g''_{i,P}(t)\right)^{3/2} = 2\left(g''_P(t)\right)^{3/2}.$$

Note that the first inequality follows from the triangle inequality, the second from (A.5), and the last one from the fact that the p-norm on $\Re^n$, $||\boldsymbol{x}||_p := \left(\sum_{i=1}^{n}|x_i|^p\right)^{1/p}$ is a decreasing function in p on $\Re_+^*$ for any $\mathbf{x} \in \Re^n$, and thus, $||\boldsymbol{x}||_1 \le ||\boldsymbol{x}||_{2/3}$. This completes the proof for part a). The proof of part b) follows in a similar way, and thus, we only provide the main steps. As before, we only show (4.3) since its proof is very similar to the proof of (4.4). We denote

$$f_{i,G}(\boldsymbol{\beta}) = \left(\frac{y_i}{2}\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)^2 - \log\left(\boldsymbol{x}_i^\top \boldsymbol{\beta}\right)\right) \quad \text{for all } 1 \le i \le n,$$

so that $f_G(\boldsymbol{\beta}) = \sum_{i=1}^{n} f_{i,G}(\boldsymbol{\beta})$. Following the same arguments as in part a), we may show that f_G is a closed convex function on Ω. The proof that f_G is self-concordant on Ω follows in a similar way by defining the function $g_{i,G}(t) = f_{i,G}(\boldsymbol{u} + t\boldsymbol{v})$ and $g_G(t) = \sum_{i=1}^{n} g_{i,G}(t)$ for any $t \in \Re$, $\boldsymbol{u} \in \Omega$ and $\boldsymbol{v} \in \Re^d$, such that $\boldsymbol{u} + t\boldsymbol{v} \in \Omega$, and showing that $\left|g'''_{i,G}(t)\right| \le 2\left(g''_{i,G}(t)\right)^{3/2}$. The second and third order derivatives of $g_{i,G}$ are given by

$$g''_{i,G}(t) = y_i \left(\boldsymbol{x}_i^\top \boldsymbol{v}\right)^2 + \frac{\left(\boldsymbol{x}_i^\top \boldsymbol{v}\right)^2}{\left(\boldsymbol{x}_i^\top \boldsymbol{u} + t\,\boldsymbol{x}_i^\top \boldsymbol{v}\right)^2} \quad \text{and} \quad g'''_{i,G}(t) = -\frac{2\left(\boldsymbol{x}_i^\top \boldsymbol{v}\right)^3}{\left(\boldsymbol{x}_i^\top \boldsymbol{u} + t\,\boldsymbol{x}_i^\top \boldsymbol{v}\right)^3}.$$

Clearly, the required inequality holds if $\boldsymbol{x}_i^\top \boldsymbol{v} = 0$, and thus, $\boldsymbol{x}_i^\top \boldsymbol{v} \neq 0$ is further assumed. Now,

$$\left|g'''_{i,G}(t)\right| \left(g''_{i,G}(t)\right)^{-3/2} = 2\left(y_i \left(\boldsymbol{x}_i^\top \boldsymbol{u} + t\,\boldsymbol{x}_i^\top \boldsymbol{v}\right)^2 + 1\right)^{-3/2} \leq 2,$$

since $(1 + y_i \epsilon_i)^{-3/2} \leq 1$ for any $y_i > 0$ and $\epsilon_i \geq 0$ (recall that $y_i \in \Re_+^*$ as the sampling distribution is Gamma). This completes the proof.

Appendix B. Other special cases of GLMs

Appendix B.1. Linear Regression – Gaussian family

Assume that $Y \sim N(\theta, \phi^2)$ with probability distribution function given by

$$\log\left(f_Y(y;\theta,\phi)\right) = \frac{\theta y - \frac{\theta^2}{2}}{\phi} - \frac{1}{2}\left(\frac{y^2}{\phi} + \log(2\pi\phi)\right), \quad (y,\theta,\phi) \in \Re \times \Re \times \Re_+^*.$$

The above pdf is obtained as a special case of (2.1) by taking

$$a\left(\phi\right) = \phi, \quad b\left(\theta\right) = \frac{\theta^2}{2}, \quad c\left(y,\phi\right) = -\frac{1}{2}\left(\frac{y^2}{\phi} + \log(2\pi\phi)\right).$$

In addition, $b'\left(\Theta\right) = \Re$ and $b'^{-1}\left(\mu\right) = \mu$. Proposition Appendix B.1 provides a characterisation of the LFs under which the Gaussian GLM is properly defined according to Definition 2.1.

Proposition Appendix B.1. *Assume that $Y \sim N(\theta, \phi^2)$. The Gaussian GLM is proper if and only if $h : \Re \to \Re$ and*

$$-yh(\eta) + \frac{h^2(\eta)}{2} \quad \textit{is convex in } \eta \textit{ on } \Re \textit{ for any given } y \in \Re. \tag{B.1}$$

Proof. The proof follows from verifying conditions **C1** and **C2** from Definition 2.1. ■
Corollary Appendix B.2 identifies the only class of LFs which satisfies Equation (B.1).

Corollary Appendix B.2. *The Gaussian GLM is proper if and only if the LF is linear.*

Proof. Since any convex real function defined on a finite open set I is continuous with non-decreasing left (and right) derivatives, then (B.1) implies that

$$h'_+\left(\eta_1\right) h\left(\eta_1\right) - yh'_+\left(\eta_1\right) \leq h'_+\left(\eta_2\right) h\left(\eta_2\right) - yh'_+\left(\eta_2\right) \quad \text{for all } y \in \Re, \tag{B.2}$$

and any reals $\eta_1 < \eta_2$ from I, where h'_+ is the right derivative of h. Assume now that h is not linear on $\Re$, and thus, not linear on I. Then, there exists $\eta_1 < \eta_2$ from I such that $h'_+\left(\eta_2\right) - h'_+\left(\eta_1\right) \neq 0$. The latter contradicts (B.2), and in turn, we must have h linear on $\Re$, and no other possible LF leads to a MLE-based Gaussian GLM. ■

The *canonical* LF for Gaussian GLMs is the *identity* function. Corollary Appendix B.2 implies the *canonical* LF leads to a proper GLM and it is the only *power* function with this property.

Appendix B.2. Logistic Regression – Bernoulli family

Assume that $Y \sim Bernoulli(\theta)$ with probability mass function given by

$$\log\left(f_Y(y;\theta,\phi)\right) = \theta y - \log\left(1+e^{\theta}\right) \quad \text{with } (y,\theta,\phi) \in \{0,1\} \times \Re \times \{1\}.$$

The above function is obtained as a special case of (2.1) by taking

$$a(\phi) = 1, \quad b(\theta) = \log\left(1+e^{\theta}\right), \quad c(y,\phi) = 0.$$

In addition, $b'(\Theta) = (0,1)$ and $b'^{-1}(\mu) = \log\frac{\mu}{1-\mu}$. Proposition Appendix B.3 provides a brief characterisation of a proper Logistic regression model.

Proposition Appendix B.3. *Assume that $Y \sim Bernoulli(\theta)$. The Bernoulli GLM is proper if and only if $h : \Re \to (0,1)$, and*

$$y\log\left(h(\eta)\right) + (1-y)\log\left(1-h(\eta)\right) \quad \textit{is concave in } \eta \textit{ on } \Re \textit{ for any given } y = \{0,1\}. \tag{B.3}$$

Proof. The proof follows easily by verifying the conditions **C1** and **C2** from Definition 2.1. ■

A direct consequence of the above is that the MLE-based Bernoulli GLM is proper if and only if $h(\eta)$ and $h(1-\eta)$ are log-concave functions[25] on $\Re$. Three standard choices for h have been proposed for this family in the literature, and all of them lead to proper GLMs:

(i) *logit* LF, which corresponds to having $h(\eta) = \frac{1}{1+e^{-\eta}}$, which is also the Bernoulli *canonical* LF that satisfies the conditions in Proposition 2.2 since b is strictly convex on $\Re$.

(ii) *probit* LF, which corresponds to having $h(\eta) = \Phi(\eta)$, where Φ is the cdf of a standard Gaussian random variable. In this case, it is not difficult to show that h satisfies the characterisation from Proposition Appendix B.3.

(iii) *complementary log-log* LF, which corresponds to having $h(\eta) = 1 - \exp\left(-\exp\left(-\eta\right)\right)$. It is not difficult to show that h satisfies the conditions in Proposition Appendix B.3.

Finally, it is clear that no power LF satisfies the conditions in Proposition Appendix B.3.

Appendix C. Data Generation Process

This section briefly outlines the DGPs for the Poisson, Gamma and Inverse Gaussian GLMs.

- **Step 1**: Generate the matrix of covariates $\boldsymbol{X} = \{X_{i,j}\}_{i=1,j=1}^{n,d}$, from a Gaussian distribution with mean μ and unit standard deviation, $X_{i,j} \sim \mathcal{N}(\mu, 1)$. Note that for each GLM, we let μ to be a function of d, such that the expected value of the response variable is within reasonable bounds in order to avoid exaggerating the parameter values when generating $\boldsymbol{Y}$ in Step 3, which typically affect the estimation procedure for the benchmarks.[26]

[25] A function $f : A \to B$ is log-concave on A if $\log\left(f\left(\alpha x + (1-\alpha)y\right)\right) \geq \alpha \log\left(f(x)\right) + (1-\alpha)\log\left(f(y)\right)$ for all $x, y \in A$ and $0 < \alpha < 1$.

[26] Note that in such cases all standard benchmarks fail to converge in most scenarios.

- **Step 2**: Generate the regression coefficient $\boldsymbol{\beta} = \{\beta_j\}_{j=0}^{d}$ by setting $\beta_j = j/d$.

- **Step 3**: For any $i = 1, \ldots, n$, let $\theta_i = \beta_0 + \sum_{j=1}^{d} \beta_j x_{i,j}$ and generate the response variable $\boldsymbol{Y} = \{Y_i\}_{i=1}^{n}$ by simulating each Y_i from $Poisson\left(\theta_i^2\right)$ for the Poisson GLM, $Gamma\left(\theta_i^2, 1\right)$ for the Gamma GLM and $IG\left(\theta_i^{-2}, 1\right)$ for the Inverse Gaussian GLM.

Appendix D. Deviance for Poisson, Gamma and Inverse Gaussian GLMs

- Poisson GLM with *half-square-root* LF

$$D\left(\hat{\boldsymbol{\beta}}\right) = \sum_{i=1}^{n} \left(4y_i \log\left(\frac{\sqrt{y_i}}{\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}}\right) + 2\left(\left(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}\right)^2 - y_i\right)\right) \cdot I_{\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} > 0} + 0 \cdot I_{\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} = y_i = 0} + \infty \cdot I_{else}$$

- Gamma GLM with *half-reciprocal-square-root* LF

$$D\left(\hat{\boldsymbol{\beta}}\right) = \sum_{i=1}^{n} \left(2y_i \left(\left(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}\right)^2 - y_i^{-1}\right) - 2\log\left(y_i\right) - 4\log\left(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}\right)\right) \cdot I_{\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} > 0} + \infty \cdot I_{\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} \leq 0}$$

- Inverse Gaussian GLM with *reciprocal-square root* LF

$$D\left(\hat{\boldsymbol{\beta}}\right) = \phi^2 \sum_{i=1}^{n} \left(y_i \left(\left(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}\right)^4 - y_i^{-2}\right) - 2\left(\left(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}\right)^2 - y_i^{-1}\right)\right) \cdot I_{\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} > 0} + \infty \cdot I_{\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} \leq 0}$$