Approximate Gibbs Sampler for Efficient Inference of Hierarchical Bayesian Models for Grouped Count Data

Jin-Zhu Yü[a,b,*] and Hiba Baroud[c,d]

[a]Department of Civil Engineering, University of Texas at Arlington, Arlington, TX, USA; [b]Department of Industrial, Manufacturing, and Systems Engineering, University of Texas at Arlington, Arlington, TX, USA; [c]Department of Civil and Environmental Engineering, Vanderbilt University, Nashville, TN, USA; [d]Department of Computer Science, Vanderbilt University, Nashville, TN, USA

ARTICLE HISTORY
Compiled August 19, 2024

Total Words: 7268

ABSTRACT
Hierarchical Bayesian Poisson regression models (HBPRMs) provide a flexible modeling approach of the relationship between predictors and count response variables. The applications of HBPRMs to large-scale datasets require efficient inference algorithms due to the high computational cost of inferring many model parameters based on random sampling. Although Markov Chain Monte Carlo (MCMC) algorithms have been widely used for Bayesian inference, sampling using this class of algorithms is time-consuming for applications with large-scale data and time-sensitive decision-making, partially due to the non-conjugacy of many models. To overcome this limitation, this research develops an approximate Gibbs sampler (AGS) to efficiently learn the HBPRMs while maintaining the inference accuracy. In the proposed sampler, the data likelihood is approximated with Gaussian distribution such that the conditional posterior of the coefficients has a closed-form solution. Numerical experiments using real and synthetic datasets with small and large counts demonstrate the superior performance of AGS in comparison to the state-of-the-art sampling algorithm, especially for large datasets.

KEYWORDS
Conditional conjugacy; Approximate MCMC; Gaussian approximation; Intractable likelihood

1. Introduction

Count data are frequently encountered in a wide range of applications, such as finance, epidemiology, sociology, and operations, among others [1]. For example, in epidemiological studies, the occurrences of a disease are often recorded as counts on a regular basis [2]. Death counts, classified by various demographic variables, are regularly recorded by government agencies [3]. In customer service centers, the service level is often measured based on the number of customers served during a given period of time. More recently, data-driven disaster management approaches have used

*CONTACT Jin-Zhu Yü. Email: yujinzhu88@gmail.com.

count data to analyze the impact of disasters (e.g., number of power outages [4] and pipe breaks [5]) and the recovery process (e.g., recovery rate [6]). Understanding the features that can influence the occurrence of such events is critical to inform future decisions and policies. Therefore, statistical models have been developed to accommodate the complexity of count data, among which are Hierarchical Bayesian Poisson regression models (HBPRMs) that have been widely employed to analyze count data under uncertainty [7–10]. The wide applicability of this class of models is due to the fact that the hierarchical Bayesian approach offers the flexibility to capture the complex hierarchical structure of count data and predictors by estimating different parameters for different data groups, thereby improving the estimation accuracy of parameters for each group. The data can be grouped based on geographical areas, types of experiments in clinical studies, or different hazard types and intensities in disaster studies. The hierarchical structure assumes that the parameters of the prior distribution are uncertain and characterized by their own probability distribution with corresponding parameters referred to as hyperparameters. Therefore, this class of models can account for the individual- and group-level variations in estimating the parameters of interest and the uncertainty around the estimation of hyperparameters [11].

The flexibility of hierarchical models in capturing the complex interactions in the data comes with a high computational expense since all the model parameters need to be estimated jointly [12]. Furthermore, large-scale data may be structured in many levels or groups [13], resulting in a large number of parameters to learn for a hierarchical model, further increasing the computational load. Given that many of the applications involving count data have recently benefited from technological advances in data collection and storage, there is a critical need to ensure the applicability of HBPRMs. As a result, efficient inference algorithms are needed to support the use of statistical learning models such as HBPRMs in risk-based decision-making, especially for time-sensitive applications such as resource allocation during emergency response and disaster recovery.

The most popular algorithms for parameter inference in hierarchical Bayesian models (and generally for Bayesian inference) are Markov Chain Monte Carlo (MCMC) algorithms. MCMC algorithms obtain samples from a target distribution by constructing a Markov chain (irreducible and aperiodic) in the parameter space that has precisely the target distribution as its stationary distribution [14]. This class of algorithms provide a powerful tool to obtain posterior samples and then estimate the parameters of interest when the exact full posterior distributions are only known up to a constant and direct sampling is not possible [14]. However, a major drawback of standard MCMC algorithms, such as the Metropolis-Hastings algorithm (MH), is that they suffer from slow mixing, requiring numerous Monte Carlo samples that grow with the dimension and complexity of the dataset [15,16]. In some applications of Bayesian approaches (e.g., emergency response), decisions relying on outcomes of the model cannot afford to wait days for running MCMC chains to collect a sufficiently large number of posterior samples. As such, the application of standard MCMC algorithms to learn Bayesian models such as HBPRMs or other hierarchical Bayesian models for large datasets is significantly limited and a fast approximate MCMC is needed.

The key idea of approximate MCMC is to replace complex distributions that lead to a computational bottleneck with an approximation that is simpler or faster to sample from than the original [17,18]. Several studies have applied analytical approximation techniques by exploiting conjugacy to accelerate MCMC-based inference in hierarchical Bayesian models [19–22]. More specifically, an approximate Gibbs sampling algorithm to is used to enable the inference of the rate parameter in the hierarchical Poisson re-

gression model in [19]. The conditional posterior of the rate parameter, which does not have a closed-form expression due to non-conjugacy between Poisson likelihood and log-normal prior distribution, is approximated as a mixture of Gaussian and Gamma distributions using the moment matching method. The exact conditional moments are obtained by minimizing the Kullback-Liebler divergence between the original and the approximate conditional posterior distributions. Conjugacy is also employed to improve inference efficiency in large and more complex hierarchical models in [21]. It is shown that the approximation using conjugacy can be utilized even though the original hierarchical model is not fully conjugate [21]. As an example in their study, the approximate full conditional distributions are derived when the likelihood function follows a gamma distribution while the prior for the parameters are assumed to be multivariate normal and inverse Wishart distribution. In [22], a Gaussian approximation to the conditional distribution of the normal random effects in the hierarchical Bayesian binomial model (HBBM) is derived using Taylor series expansion, such that Gibbs sampling can be applied to infer the HBBM more efficiently. A similar approach that approximates the data likelihood with a Gaussian distribution to allow for faster inference of parameters is used for parameter inference in a Bayesian Poisson model [20]. With regard to count data, a fast approximate Bayesian inference method is proposed to infer a negative binomial model (NB) in [23]. The non-conjugacy of the NB likelihood is addressed by the Pólya-Gamma data augmentation. This technique is first developed in [24] and is employed to approximate the likelihood as a Gaussian distribution. Consequently, the conditional posteriors of all but one parameters have a closed-form solution and a Metropolis-within-Gibbs algorithm is thus developed for the posterior inference.

While approximate MCMC algorithms have been developed for hierarchical and non-hierarchical Poisson models as well as hierarchical Bayesian binomial and negative binomial models, the development of approximate MCMC algorithm for an efficient inference of HBPRMs for grouped count data is still lacking. In this paper, we propose an approximate Gibbs sampler to address this problem. To deal with the non-conjugacy between the likelihood and the prior, we approximate the conditional likelihood as a Gaussian distribution, leading to closed-form conditional posteriors for all model parameters. The contribution lies in the derivation of a closed-form approximation to the complex conditional posterior of the parameters and the development of the Approximate Gibbs sampling (AGS) algorithm. The proposed algorithm allows for an efficient inference of the general HBPRM using the approximate Markov chain without compromising the inference accuracy, enabling the use of HBPRMs in applications with large-scale data and time sensitive decision-making. To demonstrate the performance of the proposed AGS algorithm, we conduct multiple numerical experiments and compare the inference accuracy and computational load to state-of-the-art sampling algorithms. Note that due to the use of Gaussian approximation, the AGS algorithm performs well when the dataset does not contain excessive zero counts.

The rest of this paper is organized as follows. In Sec. 2, a general hierarchical Bayesian Poisson model for grouped count data is presented, and the closed-form solution to the approximate conditional posterior distribution of each regression coefficients is derived, followed by a description of the proposed AGS algorithm. Sec. 3 introduces the datasets used in the numerical experiments along with the comparison of the performance of sampling algorithms. Conclusions and future work are provided in Sec. 4.

2. Methodology

2.1. Hierarchical Bayesian Poisson Regression Model

This section presents the Hierarchical Bayesian Poisson Regression Model (HBPRM) for count data. Without loss of generality, we consider a general HBPRM, the hierarchical version of Poisson log-normal model [19,25–27] for grouped count data, in which the coefficient for each covariate varies across groups (Eq. (1) to Eq. (5)). This model can be applied to count datasets in which the counts can be divided into multiple groups based on the covariates. Let $\mathcal{D} = \{x, y\}$ be the dataset where x represents the covariates and y represents the dependent positive counts. This HBPRM assumes that each count, y_{ij}, follows a Poisson distribution. The log of the mean in the Poisson distribution is a linear function of the covariates. In the hierarchical Bayesian paradigm, each of the parameters (regression coefficients) in the linear function follows a prior distribution with hyperparameter(s) which are in turn specified by a hyperprior distribution. Note that the hyperpriors are shared among the parameters of the same covariate for all groups, thereby resulting in shrinkage of the parameters towards the group-mean and facilitating strength borrowing across groups [11]. When the variance of the hyperprior is decreased to zero, the hierarchical model is reduced to a non-hierarchical model. The mathematical formulation of the HBPRM is provided in Eq. (1) to Eq. (5):

$$y_{ij}|\lambda_{ij} \sim \text{Pois}(\lambda_{ij}),\ \forall\ i = 1, \ldots, n_j,\ j = 1, \ldots, J, \tag{1}$$

$$\ln \lambda_{ij} = \sum_{k=1}^{K} w_{jk} x_{ijk},\ \forall\ i = 1, \ldots, n_j,\ j = 1, \ldots, J,\ k = 1, \ldots, K, \tag{2}$$

$$w_{jk}|\mu_k, \sigma_k^2 \sim \mathcal{N}(\mu_k, \sigma_k^2),\ \forall\ j = 1, \ldots, J,\ k = 1, \ldots, K, \tag{3}$$

$$\mu_k|m, \tau^2 \sim \mathcal{N}(m, \tau^2),\ \forall\ k = 1, \ldots, K, \tag{4}$$

$$\sigma_k^2|a, b \sim \mathcal{IG}(\frac{a}{2}, \frac{b}{2}),\ \forall\ k = 1, \ldots, K. \tag{5}$$

In the HBPRM formulation, y_{ij} is the i-th count within group j with an estimated mean of λ_{ij}, n_j is the number of data points in group j, w_{jk} is the regression coefficient of covariate k, and x_{ijk} is the i-th value in group j of covariate k. The prior for the coefficient of each covariate, μ_k, is assumed to be a Gaussian distribution ($\mathcal{N}$) while the prior for the variance, σ_k^2, is assumed to be an inverse-gamma distribution ($\mathcal{IG}$). The Gaussian and inverse-gamma distributions are specified such that we can exploit conditional conjugacy for analytical and computational convenience. Alternative distributions (such as half-Cauchy and uniform distributions) for the prior of group-level variance σ_k^2 do not have this benefit, which will significantly increase the computational load. According to Ref. [28], when the group-level variance is close to zero, the shape parameter a in the inverse-gamma distribution must be set to a reasonable value. For our model, the estimated group variance is always much larger than zero because the mean of estimated variance is approximately $\frac{a+J}{2}$ (Eq. (11)) where J is the number of groups. Therefore, a can be set to a sufficiently large value, such as 2.

2.2. Inference

Given an observed count dataset structured using multiple groups, $\mathcal{D}$, fitting an HBPRM entails the estimation of the joint posterior density distribution of all the parameters, which is only known up to a constant. If we denote the parameters by $\Theta = \left\{w_{11}, \ldots, w_{jk}, \ldots, w_{JK}; \mu_1, \ldots, \mu_K; \sigma_1^2, \ldots, \sigma_K^2\right\}$, then the joint posterior factorizes as

$$p\left(\Theta|y,x\right) \propto \prod_{j=1}^{J}\prod_{i=1}^{n_j} \text{Pois}\left(y_{ij} \mid \exp\left(\sum_{k=1}^{K} w_{jk}x_{ijk}\right)\right) \times \prod_{k=1}^{K} \mathcal{N}\left(w_{jk}|\mu_k,\sigma_k^2\right)\mathcal{N}\left(\mu_k|m,\tau^2\right)\mathcal{IG}\left(\sigma_k^2|\frac{a}{2},\frac{b}{2}\right). \tag{6}$$

Sampling from the joint posterior becomes a challenging task as it does not admit a closed-form expression. While MCMC algorithms (e.g., the MH) can be used, the need to judiciously tune the step size for the desired acceptance rate often repel users from using this algorithm [29,30]. In comparison, the Gibbs sampler is more efficient and does not require any tuning of the proposal distribution, therefore it has been used for Bayesian inference in a wide range of applications [31,32]. Classical Gibbs sampling requires that one can directly sample from the conditional posterior distribution of each parameter (or block of parameters), such as conditional conjugate posterior distributions. The full conditional posteriors for implementing the Gibbs sampler are

$$p\left(w_{jk}|-\right) \propto \prod_{i=1}^{n_j} \text{Pois}\left(y_{ij} \mid \exp\left(\sum_{k=1}^{K} w_{jk}x_{ijk}\right)\right)\mathcal{N}\left(w_{jk}|\mu_k,\sigma_k^2\right), \tag{7}$$

$$p\left(\mu_k|-\right) \propto \mathcal{N}\left(w_{1k},\ldots,w_{Jk}|\mu_k,\sigma_k^2\right)\mathcal{N}\left(\mu_k|m,\tau^2\right), \tag{8}$$

$$p\left(\sigma_k^2|-\right) \propto \mathcal{N}\left(w_{1k},\ldots,w_{Jk}|\mu_k,\sigma_k^2\right)\mathcal{IG}\left(\sigma_k^2|\frac{a}{2},\frac{b}{2}\right), \tag{9}$$

where $p\left(\cdot|-\right)$ represents the conditional posterior of a parameter of interest given the remaining parameters and the data. Due to the Gaussian-Gaussian and Gaussian-inverse-gamma conjugacy, Eq. (8) and Eq. (9) can be expressed in an analytical form [33]

$$p\left(\mu_k|-\right) \propto \mathcal{N}\left(\mu_k \left| \frac{1}{\frac{1}{\tau^2}+\frac{J}{\sigma_k^2}}\left(\frac{m}{\tau^2}+\frac{\sum_{j=1}^{J} w_{jk}}{\sigma_k^2}\right), \frac{1}{\frac{1}{\tau^2}+\frac{J}{\sigma_k^2}}\right.\right), \tag{10}$$

$$p\left(\sigma_k^2|-\right) \propto \mathcal{IG}\left(\sigma_k^2 \left| \frac{a+J}{2}, \frac{b+\sum_{j=1}^{J}\left(w_{jk}-\mu_k\right)^2}{2}\right.\right). \tag{11}$$

However, Eq. (7) does not admit an analytical solution because the Poisson likelihood is not conjugate to the Gaussian prior. Consequently, it is challenging to sample directly from the conditional posterior to enable the Gibbs sampler. In this case, other

algorithms can be used to obtain $p(w_{jk}|-)$, such as adaptive-rejection sampling [34], and Metropolis-within-Gibbs algorithm [35]. However, these algorithms introduce an additional computational cost due to the need to evaluate the complex conditional distribution. Therefore, we propose to use a Gaussian approximation to the Poisson likelihood given in Eq. (7) to obtain a closed-form solution to the conditional posterior of coefficients. With the closed-form solution, the complex inference of regression coefficients can be simplified to save computational resources. Reducing the computational cost of sampling from $p(w_{jk}|-)$ is critical for datasets with a large number of groups because the number of regression coefficients, $J \times K$, can be significantly larger than the number of prior parameters, $2K$.

2.3. Gaussian Approximation to Log-gamma Distribution

This section introduces the Gaussian approximation to the log-gamma distribution that is used to obtain the closed-form approximate conditional posterior distribution in Section 2.4. Consider a gamma random variable z with probability density function (pdf) given by

$$p(z|\alpha,\beta) = \frac{z^{\alpha-1}e^{-\frac{z}{\beta}}}{\Gamma(\alpha)\,\beta^{\alpha}},\ \alpha>0,\ \beta>0, \tag{12}$$

where $\Gamma(\cdot)$ is the gamma function, and α and β are the location parameter and the scale parameter, respectively. The random variable, $\ln z \in \mathbb{R}$, follows a log-gamma distribution. The mean μ_z and variance σ_z^2 of log-gamma distribution are calculated using Eq. (13) and Eq. (14), respectively [36].

$$\mu_z = \psi_0(\alpha) + \ln\beta \tag{13a}$$

$$= -\gamma + \sum_{n=1}^{\infty}\left(\frac{1}{n} - \frac{1}{n+\alpha-1}\right) + \ln\beta \tag{13b}$$

$$= -\gamma + \sum_{n=1}^{\alpha-1}\frac{1}{n} + \ln\beta \tag{13c}$$

$$\sigma_z^2 = \psi_1(\alpha) \tag{14a}$$

$$= \sum_{n=0}^{\infty}\frac{1}{(\alpha+n)^2} \tag{14b}$$

$$= \frac{\pi^2}{6} - \sum_{n=1}^{\alpha-1}\frac{1}{n^2} \tag{14c}$$

In Eq. (13a) and Eq. (14a), $\psi_0(\cdot)$ and $\psi_1(\cdot)$ are the zeroth and the first order of polygamma functions [37]. In Eq. (13b) and Eq. (13c), γ is the Euler-Mascheroni constant [38].

For large values of α, the pdf of log-gamma distribution can be approximated by that of a Gaussian distribution [20,39], shown in Eq. (15).

$$\text{Log-gamma}(\ln z|\alpha,\beta) \approx \mathcal{N}(\ln z|\psi_0(\alpha)+\ln\beta,\ \psi_1(\alpha)) \tag{15}$$

To apply the approximation in the conditional posterior (Eq. (20) to Eq. (21)), we need to include y in the pdf of log-gamma distribution. Therefore, we let $\alpha = y$ and $\beta = 1$, and Eq. (15) becomes

$$\text{Log-gamma}(\ln z|y, 1) \approx \mathcal{N}\left(\ln z|\psi_0\left(y\right), \psi_1\left(y\right)\right). \tag{16}$$

Note that because $\alpha > 0$ and α is replaced by count data y, the approximation can only be applied to positive counts.

Similarly, plugging in $\alpha = y$, $\beta = 1$, and $\Gamma(n) = (n-1)!$ where $n \in \{1, 2, 3, \dots\}$, Eq. (12) becomes

$$p\left(z|y, 1\right) = \frac{z^{y-1}e^{-z}}{(y-1)!}. \tag{17}$$

Next, we need to relate Eq. (16) and Eq. (17). First, using the "change of variable" method and substituting $\ln z$ with v [20] in Eq. (17), we obtain the pdf of v

$$p\left(v|y, 1\right) = p\left(z = e^v|y, 1\right)\frac{\partial e^v}{\partial v} \tag{18a}$$

$$= \frac{1}{(y-1)!}e^{vy}e^{-e^v}. \tag{18b}$$

Then, using Eq. (16) yields

$$\frac{1}{(y-1)!}e^{vy}e^{-e^v} \approx \frac{1}{\sqrt{2\pi\psi_1(y)}}e^{\frac{(v-\psi_0(y))^2}{-2\psi_1(y)}}. \tag{19}$$

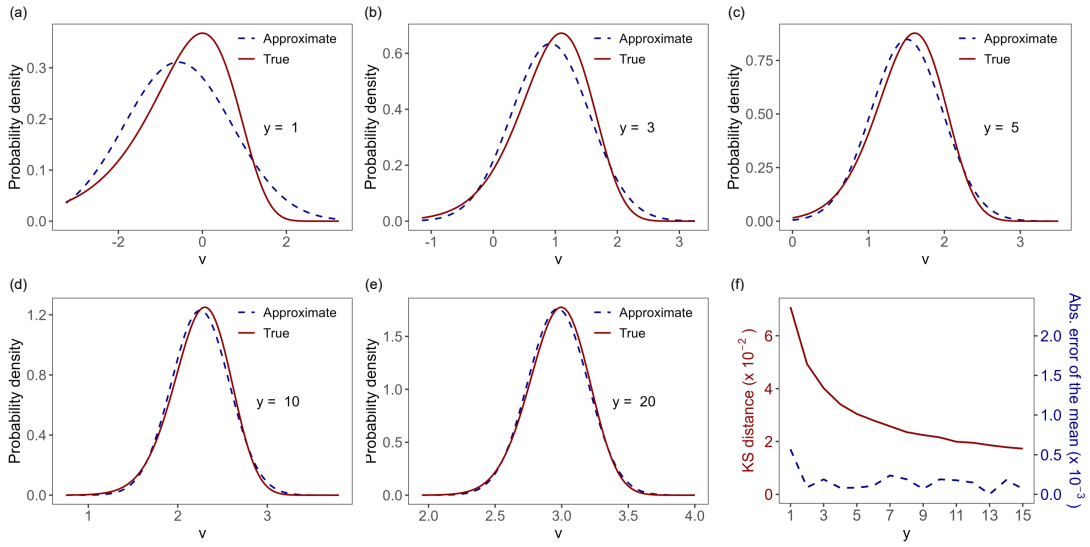


Figure 1. Quality of the Gaussian approximation (dashed blue line) to the true distribution (solid red line) for different values of y: (a) y=1; (b) y=3; (c) y=5; (d) y=10; (e) y=20. (f) Values of KS distance (solid red line) between the approximate distribution and true distribution and values of the absolute error in the mean (dashed blue line) of the approximate distribution as the value of y increases.

The comparison between the true and approximate Gaussian distribution, i.e., left and right-hand sides of Eq. (19) respectively, is shown in Fig. 1. When the counts are small, such as $y \leq 5$ ((a) to (c)), the approximation is not very close to the true distribution. We can also see that the Kolmogorov–Smirnov (KS) distance shown in Fig. 1 (f), defined as the largest absolute difference between the cumulative density functions for the approximate distribution and true distribution, is relatively large. As the value of y increases, the approximate Gaussian distribution is increasingly closer to the true distribution. Also, the absolute error in the mean value of the Gaussian approximation is relatively small when y is greater than 3. Notice again that the approximation given by Eq. (19) is not directly applicable to zero counts. However, when zero counts are present in the dataset, such as in epidemiology studies, one can increase each count by a positive count (e.g., 5). This linear transformation allows one to circumvent the problem arising from zero counts (dependent variable) without compromising the model accuracy, which is because such a transformation does not change the distribution of the error and preserves the relation between the dependent and independent variables.

2.4. Closed-form Approximate Conditional Posterior Distribution

In the conditional posterior of coefficient w_{jk} given by Eq. (7), the likelihood function is

$$\prod_{i=1}^{n_j} \text{Pois}\left(y_{ij} \mid e^{\sum_{k=1}^{K} w_{jk}x_{ijk}}\right) = \prod_{i=1}^{n_j} \frac{1}{y_{ij}(y_{ij}-1)!} e^{\sum_{k=1}^{K} w_{jk}x_{ijk}y_{ij}} e^{-e^{\sum_{k=1}^{K} w_{jk}x_{ijk}}}. \tag{20}$$

Applying the approximation given by Eq. (19) yields

$$\prod_{i=1}^{n_j} \text{Pois}\left(y_{ij} \mid e^{\sum_{k=1}^{K} w_{jk}x_{ijk}}\right) \approx \prod_{i=1}^{n_j} \frac{1}{y_{ij}} \frac{1}{\sqrt{2\pi\psi_1(y_{ij})}} e^{\frac{\left(\sum_{k=1}^{K} w_{jk}x_{ijk} - \psi_0(y_{ij})\right)^2}{-2\psi_1(y_{ij})}}. \tag{21}$$

Plugging Eq. (21) into Eq. (7) we get

$$p\left(w_{jk}|-\right) \propto \exp\left[\frac{\left(w_{jk}-\mu_k\right)^2}{-2\sigma_k^2}\right] \prod_{i=1}^{n_j} \exp\left\{\frac{\left[\sum_{k=1}^{K} w_{jk}x_{ijk} - \psi_0\left(y_{ij}\right)\right]^2}{-2\psi_1\left(y_{ij}\right)}\right\} \tag{22}$$

$$= \exp\left\{\frac{\left(w_{jk}-\mu_k\right)^2}{-2\sigma_k^2} + \sum_{i=1}^{n_j} \frac{\left[\sum_{k=1}^{K} w_{jk}x_{ijk} - \psi_0\left(y_{ij}\right)\right]^2}{-2\psi_1\left(y_{ij}\right)}\right\}. \tag{23}$$

As the product of two Gaussians is still Gaussian, the posterior can also be written as

$$p\left(w_{jk}|-\right) \propto \exp\left[\frac{\left(w_{jk}-\widehat{\mu}_k\right)^2}{-2\widehat{\sigma}_k^2}\right], \tag{24}$$

where $\widehat{\mu}_k$ and $\widehat{\sigma}_k^2$ are the mean and variance of the approximate Gaussian posterior. Completing the squares (see Appendix A for more details) we get

$$\widehat{\mu}_k = \frac{\mu_k + \sigma_k^2 \sum_{i=1}^{n_j} \frac{x_{ijk}}{\psi_1(y_{ij})} \left[\psi_0 \left(y_{ij} \right) - \sum_{h=1, h \neq k}^{K} x_{ijh} w_{jh} \right]}{\sigma_k^2 \sum_{i=1}^{n_j} \frac{x_{ijk}^2}{\psi_1(y_{ij})} + 1}, \forall j = 1, \ldots, J, \tag{25}$$

$$\widehat{\sigma}_k^2 = \frac{\sigma_k^2}{\sigma_k^2 \sum_{i=1}^{n_j} \frac{x_{ijk}^2}{\psi_1(y_{ij})} + 1}, \forall j = 1, \ldots, J. \tag{26}$$

Now that the full conditional posterior distributions can be expressed analytically, we can construct the approximate Gibbs sampler (Algorithm 1) to obtain posterior samples of the parameters in HBPRM efficiently.

Algorithm 1: Approximate Gibbs sampler

Input: x, y, number of samples as warm-ups N_0, number of desired samples N_1.

Output: Desired posterior samples, $\mu_k^{(\ell)}, \sigma_k^{2(\ell)}, w_{jk}^{(\ell)}$, $\ell = N_0 + 1, \ldots, N_0 + N_1$, $k = 1, \ldots, K$, and $j = 1, \ldots, J$.

```
Generate the initial sample μ_k^(0), σ_k^2(0), w_jk^(0).
for ℓ = 1 to (N_0 + N_1) do
    for k = 1 to K do
        Sample hyperparameter μ_k^(ℓ) according to Eq. (10).
        Sample hyperparameter σ_k^2(ℓ) according to Eq. (11).
        for j = 1 to J do
            Sample each parameter w_jk^(ℓ) according to Eq. (24).
        end for
    end for
end for
```

3. Experiments

We evaluate the performance of our proposed AGS algorithm by applying it to several synthetic and real data sets. The performance of AGS is evaluated in terms of the accuracy, efficiency, and computational time. The proposed approach is compared with the state-of-the-art MCMC algorithm, No-U-Turn sampler (NUTS) [40], using the same datasets and performance metrics. NUTS is an extension to Hamiltonian Monte Carlo (HMC) algorithm that exploits Hamiltonian dynamics to propose samples. NUTS can free users from tuning the proposals and has been demonstrated to provide efficient inference of complex hierarchical models [12]. The description of the datasets and the experimental setup is provided in this section. The code and non-confidential data used for the experiments are available on the GitHub account of the corresponding author.

3.1. Data Description

Multiple synthetic and real datasets are used to evaluate the performance of AGS for different data types and sizes. This section describes the approach to generating synthetic data and the characteristics of real datasets which include power outages, Covid-19 positive cases, and bike rentals. A subset of each dataset is provided in tables 1 to 4.

Synthetic data. The synthetic datasets are generated according to the model shown in Eq. (27). This model ensures that the generated datasets contain a specified range of counts and closely mimic the number of emergency incidents during disasters, such as the number of power outages after a severe storm. An example of the synthetic dataset is presented in Table 1. Note that the data for all x_k, $k = 1, ..., 6$, in each group is sorted in ascending order before calculating y to ensure a more consistent relationship between y and x.

$$x_{ij1} \sim U\left(0.1,\, 2\right),\ i = 1, \ldots, n_j,\ j = 1, \ldots, J \tag{27a}$$
$$x_{ij2} \sim U\left(0.1,\, 1\right),\ i = 1, \ldots, n_j,\ j = 1, \ldots, J \tag{27b}$$
$$x_{ij3} \sim U\left(0.1,\, 0.5\right),\ i = 1, \ldots, n_j,\ j = 1, \ldots, J \tag{27c}$$
$$x_{ij4} \sim U\left(1,\, 10\right),\ i = 1, \ldots, n_j,\ j = 1, \ldots, J \tag{27d}$$
$$x_{ij5} \sim U\left(0.5,\, 5\right),\ i = 1, \ldots, n_j,\ j = 1, \ldots, J \tag{27e}$$
$$x_{ij6} \sim U\left(10,\, 100\right),\ i = 1, \ldots, n_j,\ j = 1, \ldots, J \tag{27f}$$
$$x_{\cdot j} \sim U\left(10^4,\, 10^6\right),\ j = 1, \ldots, J \tag{27g}$$
$$w_{jk} \sim \mathcal{N}\left(0.001,\, 0.001\right),\ j = 1, \ldots, J,\ k = 1, \ldots, K \tag{27h}$$
$$y_{ij} = \left(e^{\sum_{k=1}^{K} w_{jk} x_{ijk}}\right)_{\min-\max} x_{\cdot j},\ i = 1, \ldots, n_j,\ j = 1, \ldots, J. \tag{27i}$$

In Eq. (27), the notation $(\cdot)_{\min-\max}$ represents the min-max normalizing function[1]. Each count y_{ij} is rounded to the nearest integer. $x_{\cdot j}$ is the group-level covariate for group j. We generate 15 synthetic datasets (S1, ..., S15) with varying total numbers of data points (N_d) in each dataset and varying numbers of data points in each group n_j (for simplicity, it is assumed the same for each group in the same synthetic dataset), numbers of covariates K ($K \leq 6$), and numbers of groups J to analyze the effect of the size of the data on the performance of AGS and NUTS (Table 5).

Table 1. An example of synthetic dataset

x_1	x_2	x_3	x_4	x_5	x_6	y
0.14	0.11	0.27	1.49	0.66	12.10	42
0.61	0.15	0.27	2.22	1.52	24.76	6440
0.64	0.58	0.27	3.11	1.93	34.67	11424
0.77	0.58	0.30	5.70	2.13	38.71	13535
1.07	0.60	0.34	6.38	2.77	62.73	25064
1.29	0.75	0.42	7.78	3.69	79.38	33917
1.41	0.91	0.47	8.84	4.91	85.56	38272
1.55	0.93	0.49	8.93	4.96	92.86	41806

[1]For an array of real numbers represented by a generic vector $\boldsymbol{x}$. The min-max normalization of $\boldsymbol{x}$ is given by $\boldsymbol{x}_{\min-\max} = \frac{\boldsymbol{x}-\boldsymbol{x}_{\min}}{\boldsymbol{x}_{\max}-\boldsymbol{x}_{\min}}$.

Power outage data. The power outage data includes the number of customers without power in multiple counties following 11 disruptive events (denoted by P1, . . . , P11). The power outage dataset following a particular disruptive event is grouped by county, i.e., power outage counts for the same county after a particular disruptive event fall into the same group. The covariates in each dataset include PS (surface pressure, Pa), TQV (precipitable water vapor, kg·m^{-2}), U10M (10-meter eastward wind speed, m/s), V10M (10-meter northward wind speed, m/s), t (time after the start of an event, hours). The outage datasets were collected from public utility companies during severe weather events and the weather data from the National Oceanic and Atmospheric Administration.

Table 2. A subset of power outage data

Event ID	PS	TQV	U10M	V10M	t	Outage count
1	99691.45	43.18	2.39	4.79	4	66807
1	100917.62	26.75	1.11	1.39	32	18379
1	101041.88	36.29	1.11	-0.18	60	12096
1	101467.45	55.72	-1.73	-0.67	116	14231
1	101155.43	37.50	-0.08	-3.01	144	10155
1	101037.79	32.13	-0.48	-1.53	172	4758
1	101194.86	40.20	-0.90	1.66	200	2699
1	101183.98	46.61	-0.54	1.20	228	2297
1	101136.76	34.68	-1.14	-1.50	256	248
1	101086.31	43.80	-2.19	-2.44	284	43

Covid-19 test data. The Covid-19 test dataset is obtained from Ref. [41], which are originally collected from seven papers (two preprints and five peer-reviewed articles) that provide data on RT-PCR (reverse transcriptase polymerase chain reaction) performance by time since the symptom onset or exposure using samples derived from nasal or throat swabs among patients tested for Covid-19. The number of studies (groups) is 11. Each study includes multiple test cases (Table 3), each of which includes the days, t, after exposure to Covid-19, and the total number of samples tested, N_s. The response variable is the number of patients who tested positive among the samples. The total number of test cases is 379. As the proposed approximation cannot be applied to zero counts, we remove the test cases with zero positive test among the samples tested. The total number of test cases after removing those with zero counts is 298. The test cases are grouped by studies. Following Ref. [41], the exposure is assumed to have occurred five days before the symptom onset and $\log(t)$, $\log(t)^2$, $\log(t)^3$, and N_s are used as the covariates.

Bike sharing data. The bike sharing data include daily bike rental counts for 729 days and the covariates we use include normalized temperature, normalized humidity, and casual bike rentals. The dataset is obtained from the UCI Machine Learning Repository [42]. Bike rental counts are grouped by whether the rental occurs on a working day (Table 4).

To investigate the performance of AGS for small counts, including zero counts, we also simulate datasets with small counts using the following model shown in Eq. (28).

Table 3. A subset of the Covid-19 test data

Study ID	Test case ID	$\log(t)$	$\log(t)^2$	$\log(t)^3$	N_s	Positive count
1	1	1.23	1.51	1.86	35	15
1	2	1.26	1.58	1.98	23	11
1	3	1.28	1.64	2.09	20	6
1	5	1.32	1.75	2.31	20	8
1	6	1.34	1.80	2.42	11	3
1	7	1.36	1.85	2.53	11	5
1	8	1.38	1.90	2.63	9	2
1	9	1.40	1.95	2.73	6	3
1	10	1.41	1.98	2.83	5	2

Table 4. A subset of the bike sharing data. Workingday=1 indicates the rental occurs on a working day, and 0 otherwise.

Workingday	Temperature	Humidity	Casual rental count	Total count
0	0.34	0.81	331	985
0	0.36	0.70	131	801
1	0.20	0.44	120	1349
1	0.20	0.59	108	1562
1	0.23	0.44	82	1600
1	0.20	0.52	88	1606
1	0.20	0.50	148	1510
0	0.17	0.54	68	959
0	0.14	0.43	54	822

$$x_{ij1} \sim U\,(0.1,\, 2)\,,\ i = 1, \ldots, n_j,\ j = 1, \ldots, J \tag{28a}$$
$$x_{ij2} \sim U\,(0.1,\, 1)\,,\ i = 1, \ldots, n_j,\ j = 1, \ldots, J \tag{28b}$$
$$x_{ij3} \sim U\,(0.1,\, 0.5)\,,\ i = 1, \ldots, n_j,\ j = 1, \ldots, J \tag{28c}$$
$$x_{ij4} \sim U\,(1,\, 10)\,,\ i = 1, \ldots, n_j,\ j = 1, \ldots, J \tag{28d}$$
$$x_{ij5} \sim U\,(0.5,\, 5)\,,\ i = 1, \ldots, n_j,\ j = 1, \ldots, J \tag{28e}$$
$$x_{\cdot j} \sim TEXP\,(0.7,\, 1,\, y_{\max})\,,\ j = 1, \ldots, J \tag{28f}$$
$$w_{jk} \sim \mathcal{N}\,(0.1,\, 0.1)\,,\ j = 1, \ldots, J,\ k = 1, \ldots, K \tag{28g}$$
$$y_{ij} = \left\lfloor \left(e^{\sum_{k=1}^{K} w_{jk} x_{ijk}} \right)_{\min-\max} x_{\cdot j} \right\rfloor,\ i = 1, \ldots, n_j,\ j = 1, \ldots, J. \tag{28h}$$

$TEXP$ represents the truncated exponential distribution. The PDF of $TEXP(0.7,\, 1,\, y_{\max})$, where 0.7 is the rate parameter while 1 and $y_{\max}$ are the lower and upper bounds, is given by

$$f(x) = \begin{cases} 0.7e^{-0.7(y_{\max}-x-1)}, & x \in [1,\, y_{\max}], \\ 0, & \text{otherwise.} \end{cases} \tag{29}$$

This particular truncated exponential distribution instead of a uniform distribution is used to ensure that the generated counts will not concentrate on small values. By changing the value of the upper bound, we can generate counts in different ranges. Notice that since the floor function, $\lfloor \cdot \rfloor$, is used in Eq. (28h), the generated counts can have zeros, which is smaller than the lower bound 1.

3.2. Experiment Setup

In the HBPRM for the count datasets listed above, we employ $\mathcal{N}(0,1)$ and $\mathcal{IG}(1,1)$ as a weakly-informative prior [43] for μ_k and σ_k^2 respectively. In the numerical experiments, NUTS is implemented with Stan [43]. Numbers are averaged over 4 runs of 10000 iterations for each algorithm, discarding the first 5000 samples as warm-ups/burn-ins. We compare AGS with NUTS in terms of average sampling time in seconds per 1000 iterations (T_s), sampling efficiency (E_s), R^2, and Root Mean Square Error (RMSE). Sampling efficiency is quantified as the mean effective sampler size ($\hat{n}_{\text{eff}}$) over the average sampling time in seconds per 1000 iterations, i.e., $E_s = \hat{n}_{\text{eff}}/T_s$, where $\hat{n}_{\text{eff}}$ is the effective sample size of multiple sequences of samples [11, Chapter 11]. To make this paper self-contained, we have included the details for calculating $\hat{n}_{\text{eff}}$, R^2, and RMSE in Appendix B. All experiments are implemented with `R` (version 3.6.1) on a Windows 10 desktop computer with a 3.40 GHz Intel Core i7-6700 CPU and 16.0 GB RAM.

3.3. Results

The performances of NUTS and AGS under different datasets are summarized in Table 5 (synthetic datasets) and Table 6 (real datasets). On both the synthetic and real datasets, AGS consistently outperforms NUTS in the average sampling time, especially when the size of the datasets is large. Depending on the dataset, the improvement in the average inference speed can be greater than one order of magnitude. This observation shows that using the Gaussian approximation to avoid the evaluation of complex conditional posterior can significantly boost the sampling speed. However, for all the datasets except for power outage dataset P1, the sampling efficiency of AGS is significantly lower than that of NUTS because the effective sample size obtained from AGS is much lower than that from NUTS. The relatively low inference efficiency of AGS does not compromise the accuracy of parameter estimates. In all the examined datasets, R^2 and RMSE have comparable values for both AGS and NUTS. The inference accuracy is better (higher R^2 and lower RMSE) for AGS across all the synthetic datasets except for S14 where the RMSE of NUTS is marginally smaller than that of AGS. In eight out of the thirteen (about 62%) real datasets, AGS has slightly higher R^2 and lower RMSE. In particular, the results on the Covid test data show that as long as a significant percentage of counts are not all very small counts, then the proposed approximate Gibbs sampler can outperform the NUTS in predictive accuracy. Overall, it can be concluded that AGS significantly decreases the computational load by allowing for faster sampling without compromising the accuracy of the estimates.

Table 5. Performance of NUTS and AGS on synthetic datasets

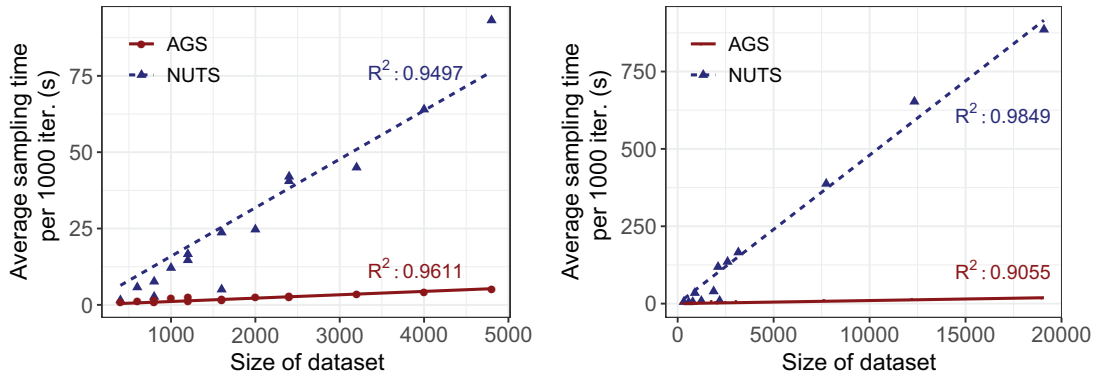
Dataset	Characteristics			T_s (s)		E_s		R^2		RMSE	
	N_d	K	J	NUTS	AGS	NUTS	AGS	NUTS	AGS	NUTS	AGS
S1	200	2	10	1.51	**0.96**	**649.28**	33.46	0.9390	**0.9450**	6500	**6191**
S2	400	2	10	2.66	**0.98**	**373.46**	34.99	0.9430	**0.9500**	5823	**5455**
S3	800	2	20	5.05	**1.63**	**195.21**	18.63	0.9576	**0.9626**	3526	**3310**
S4	200	3	10	5.76	**1.25**	**170.39**	5.28	0.9614	**0.9660**	4235	**3976**
S5	400	3	10	16.63	**1.29**	**59.78**	2.03	0.9683	**0.9710**	3450	**3305**
S6	800	3	20	40.53	**2.44**	**24.30**	1.80	0.9677	**0.9719**	3993	**3728**
S7	200	4	10	7.64	**1.63**	**129.25**	1.59	0.9716	**0.9760**	2955	**2718**
S8	400	4	10	23.72	**1.94**	**41.69**	1.50	0.9639	**0.9701**	4406	**4012**
S9	800	4	20	45.00	**3.31**	**21.99**	0.49	0.9740	**0.9770**	2958	**2790**
S10	200	5	10	12.12	**2.06**	**81.26**	0.70	0.9574	**0.9631**	4941	**4593**
S11	400	5	10	24.70	**2.41**	**39.68**	0.49	0.9782	**0.9826**	2514	**2245**
S12	800	5	20	64.00	**4.12**	**15.44**	0.27	0.9767	**0.9804**	3446	**3157**
S13	200	6	10	14.66	**2.44**	**67.49**	0.23	0.9817	**0.9876**	**2720**	2721
S14	400	6	10	42.01	**2.63**	**20.17**	0.24	0.9922	**0.9948**	1339	**1128**
S15	800	6	20	93.20	**4.97**	**8.44**	0.17	0.9910	**0.9940**	1994	**1629**

We also investigate the scalability of the two algorithms as it is crucial for large-scale hierarchical data. Therefore, we show the average sampling time of both algorithms across different dataset sizes to understand their performance for larger datasets. We conduct an empirical analysis of the average sampling time (seconds) per 1000 iterations for all the synthetic and real datasets shown in Fig. 2. The sampling time of both samplers increases as a function of the size of the dataset. However, when compared to NUTS, the increase in the sampling time of AGS is significantly lower, showing a significantly smaller rate of time increase over the size of datasets and suggesting improved scalability. This observation also indicates that although NUTS can generate samples effectively, it becomes inefficient in the case of large datasets as evaluating the gradient in proposing new samples becomes computationally expensive [44].

Table 6. Performance of NUTS and AGS on real datasets

Dataset	Characteristics			T_s (s)		E_s		R^2		RMSE	
	N_d	K	J	NUTS	AGS	NUTS	AGS	NUTS	AGS	NUTS	AGS
P1	3817	5	56	885.65	**15.76**	1.12	**1.25**	0.9730	**0.9801**	13558	**11621**
P2	2467	5	50	652.87	**13.71**	**1.50**	0.62	0.9873	**0.9884**	1446	**1384**
P3	1548	5	35	387.47	**9.41**	**2.54**	0.92	0.9850	**0.9870**	1974	**1833**
P4	632	5	26	165.73	**6.67**	**5.94**	0.46	**0.9923**	0.9918	**1327**	1373
P5	520	5	16	135.16	**4.31**	**7.27**	1.85	0.9934	**0.9940**	3473	**3312**
P6	421	5	17	118.73	**4.54**	**8.25**	2.68	0.9908	**0.9918**	2526	**2387**
P7	375	5	23	39.86	**5.76**	**24.75**	2.41	**0.9459**	0.9355	**3574**	3903
P8	247	5	10	8.75	**2.78**	**111.91**	2.38	**0.9744**	0.9729	**795**	818
P9	157	5	8	5.48	**2.24**	**179.87**	4.07	0.9964	**0.9967**	803	**766**
P10	115	5	6	4.49	**1.72**	**218.17**	6.75	0.9869	**0.9915**	7715	**6222**
P11	63	5	4	5.49	**1.25**	**177.38**	0.92	**0.9356**	0.9027	**251**	308
Bike share	729	3	2	9.09	**0.98**	**109.54**	24.75	**0.6743**	0.6292	**1101**	1175
Covid test	298	3	11	34.60	**2.59**	**28.57**	18.54	0.8517	**0.8582**	2.53	**2.47**

Results on the inference accuracy of two algorithms on selected Covid-19 test dataset with small counts are summarized in Table 7 where RG represents the range of counts, PCT_0 represents the percentage of zero counts, and $PCT_{1,5}$ represents the percentage of counts in [1, 5]. We can see that AGS can outperform NUTS in R^2 and RMSE when there are no zero counts (subset1 and subset3) in the covid test data. However,



(a) (b)

Figure 2. Scalability of NUTS and AGS on (a) real datasets and (b) synthetic datasets. The size of a dataset, i.e., the total number of count data points, can be calculated by $\sum_{j=1}^{J} n_j$.

Table 7. Inference accuracy of NUTS and AGS on selected Covid-19 test datasets with small counts. The number of covariates $K = 3$ and the number of groups $J = 11$.

Dataset	Characteristics				R^2		RMSE	
	RG	PCT_0 (%)	$PCT_{1,5}$ (%)	N_d	NUTS	AGS	NUTS	AGS
Subset1	[1, 5]	0.0	100.0	168	0.4403	**0.4452**	0.9964	**0.9920**
Subset2	[0, 5]	32.5	67.5	249	**0.6606**	0.5684	**0.9370**	1.0570
Subset3	[1, 38]	0.0	56.4	298	0.8517	**0.8582**	2.5337	**2.4685**
Whole set	[0, 38]	21.4	44.3	379	**0.8692**	0.8546	**2.3492**	2.4769

when zero counts are included (subset2 and the whole set), NUTS performs better than AGS. Comparing the performance of both algorithms on subset 2, we can see that even when all the counts are small but positive, AGS can still outperform NUTS in inference accuracy by a small margin.

We also compare the inference accuracy of both algorithms using synthetic datasets with small counts (SS1 - SS10) (Table 8). It can be observed that for most synthetic datasets with a relatively large percentage of small counts (SS1-SS3 and SS5-SS8), NUTS outperforms AGS in terms of R^2 and RMSE. If the percentage of small counts is not very high (SS9 and SS10), then AGS outperforms NUTS, even when there are zero counts (SS10).

Based on the performance comparison using synthetic datasets with small counts and Covid-19 test datasets with small counts, we can conclude that when there is a large percentage of small counts, particularly zero counts, NUTS tends to outperform AGS. The specific percentage of small counts in a dataset that leads to a better performance of NUTS than AGS varies with the dataset. That is, depending on the particular dataset, AGS may still outperform NUTS when there is a large percentage of small counts.

Table 8. Inference accuracy of NUTS and AGS on synthetic datasets with small counts. The number of covariates $K = 5$ and the number of groups $J = 8$.

Dataset	Characteristics				R^2		RMSE	
	RG	PCT_0 (%)	$PCT_{1,5}$ (%)	N_d	NUTS	AGS	NUTS	AGS
SS1	[1, 5]	0.0	100.0	252	**0.9226**	0.8139	**0.3198**	0.4959
SS2	[0, 5]	21.3	78.8	320	**0.5414**	0.4567	**0.9460**	1.0010
SS3	[1, 10]	0.0	71.2	288	**0.9625**	0.9421	**0.4478**	0.5564
SS4	[0, 10]	10.0	64.1	320	0.6461	**0.6480**	1.4805	**1.4764**
SS5	[1, 15]	0.0	57.3	307	**0.9766**	0.9732	**0.5715**	0.6113
SS6	[0, 15]	4.1	55.0	320	**0.9641**	0.9538	**0.7225**	0.8193
SS7	[1, 20]	0.0	47.0	319	**0.9822**	0.9819	**0.6639**	0.6708
SS8	[0, 20]	3.1	46.9	320	**0.9691**	0.9642	**0.8774**	0.9446
SS9	[1, 30]	0.0	32.2	314	0.9723	**0.9767**	1.3059	**1.1979**
SS10	[0, 30]	1.9	31.6	320	0.9525	**0.9563**	1.7246	**1.6556**

4. Conclusions

This research proposes a scalable approximate Gibbs sampling algorithm for the HBPRM for grouped count data. Our algorithm builds on the approximation of data-likelihood with Gaussian distribution such that the conditional posterior for coefficients have a close-form solution. Empirical examples using synthetic and real datasets demonstrate that the proposed algorithm outperforms the state-of-the-art sampling algorithm, NUTS, in inference efficiency. The improvement in efficiency is greater for larger datasets, suggesting improved scalability. Due in part to the Gibbs updates, the AGS trades off greater accuracy for slower mixing Markov chains, leading to a much lower effective sample size and therefore lower sampling efficiency. However, when sampling time is of great concern to model users (e.g. predicting incidents and demands to allocate resources during a disaster), AGS would be the only feasible option. As the approximation quality improves with larger counts, our algorithm works better for count datasets in which the counts are large. When a large portion of counts in a dataset are zero or very small counts, then NUTS tend to outperform AGS in inference accuracy. Therefore, when there are zero counts and inference accuracy is critical, NUTS is recommended over AGS.

It is worth noting that the approximate conditional distributions of the parameters in the HBPRMs for grouped count data may not be compatible with each other, i.e., there may not exist an implicit joint posterior distribution [17,45] after applying the approximation. However, despite potentially incompatible conditional distributions, the use of such approximate MCMC samplers is suggested due to the computational efficiency and analytical convenience [18,45], especially when the efficiency improvement outweighs the bias introduced by approximation [17].

Future work can explore scalable inference in hierarchical Bayesian models for data with excessive zeros [46–48] as the Poisson regression model is not appropriate for zero-inflated count data.

Acknowledgment

This research was partially funded by the National Science Foundation (grant no. 1635717). We would also like to thank Prof. Qian Qin from the University of Minnesota for his helpful comments on an earlier draft of this paper.

References

[1] Aktekin T, Polson N, Soyer R. Sequential Bayesian analysis of multivariate count data. Bayesian Analysis. 2018;13(2):385–409.
[2] Hay JL, Pettitt AN. Bayesian analysis of a time series of counts with covariates: An application to the control of an infectious disease. Biostatistics. 2001;2(4):433–444.
[3] De Oliveira V. Hierarchical Poisson models for spatial count data. Journal of Multivariate Analysis. 2013;122:393–408.
[4] Han SR, Guikema SD, Quiring SM, et al. Estimating the spatial distribution of power outages during hurricanes in the Gulf coast region. Reliability Engineering & System Safety. 2009;94(2):199–210.
[5] Yu JZ, Whitman M, Kermanshah A, et al. A hierarchical bayesian approach for assessing infrastructure networks serviceability under uncertainty: A case study of water distribution systems. Reliability Engineering & System Safety. 2021;215:107735.
[6] Yu JZ, Baroud H. Quantifying community resilience using hierarchical Bayesian kernel methods: A case study on recovery from power outages. Risk Analysis. 2019;39(9):1930–1948.
[7] Ma J, Kockelman KM, Damien P. A multivariate Poisson-lognormal regression model for prediction of crash counts by severity, using Bayesian methods. Accident Analysis & Prevention. 2008;40(3):964–975.
[8] Baio G, Blangiardo M. Bayesian hierarchical model for the prediction of football results. Journal of Applied Statistics. 2010;37(2):253–264.
[9] Flask T, Schneider IV WH, Lord D. A segment level analysis of multi-vehicle motorcycle crashes in Ohio using Bayesian multi-level mixed effects models. Safety Science. 2014; 66:47–53.
[10] Khazraee SH, Johnson V, Lord D. Bayesian Poisson hierarchical models for crash data analysis: Investigating the impact of model choice on site-specific predictions. Accident Analysis & Prevention. 2018;117:181–195.
[11] Gelman A, Carlin JB, Stern HS, et al. Bayesian data analysis. Chapman and Hall/CRC; 2013.
[12] Betancourt M, Girolami M. Hamiltonian Monte Carlo for hierarchical models. Current Trends in Bayesian Methodology with Applications. 2015;79(30):2–4.
[13] AlJadda K, Korayem M, Ortiz C, et al. PGMHD: A scalable probabilistic graphical model for massive hierarchical data problems. In: 2014 IEEE International Conference on Big Data (Big Data); IEEE; 2014. p. 55–60.
[14] Brooks S. Markov Chain Monte Carlo method and its application. Journal of the Royal Statistical Society: Series D (the Statistician). 1998;47(1):69–100.
[15] Conrad PR, Marzouk YM, Pillai NS, et al. Accelerating asymptotically exact MCMC for computationally intensive models via local approximations. Journal of the American Statistical Association. 2016;111(516):1591–1607.
[16] Robert CP, Elvira V, Tawn N, et al. Accelerating MCMC algorithms. Wiley Interdisciplinary Reviews: Computational Statistics. 2018;10(5):e1435.
[17] Alquier P, Friel N, Everitt R, et al. Noisy Monte Carlo: Convergence of Markov Chains with approximate transition kernels. Statistics and Computing. 2016;26(1-2):29–47.
[18] Johndrow JE, Mattingly JC, Mukherjee S, et al. Optimal approximating Markov Chains for Bayesian inference. arXiv preprint arXiv:150803387. 2015;.

[19] Streftaris G, Worton BJ. Efficient and accurate approximate Bayesian inference with an application to insurance data. Computational Statistics & Data Analysis. 2008; 52(5):2604–2622.
[20] Chan AB, Vasconcelos N. Bayesian Poisson regression for crowd counting. In: 2009 IEEE 12th international conference on computer vision; IEEE; 2009. p. 545–551.
[21] Dutta R, Blomstedt P, Kaski S. Bayesian inference in hierarchical models by combining independent posteriors. arXiv preprint arXiv:160309272. 2016;.
[22] Berman B. Asymptotic posterior approximation and efficient MCMC sampling for generalized linear mixed models [dissertation]. UC Irvine; 2019.
[23] Bansal P, Krueger R, Graham DJ. Fast Bayesian estimation of spatial count data models. Computational Statistics & Data Analysis. 2020;:107152.
[24] Polson NG, Scott JG, Windle J. Bayesian inference for logistic models using pólya–gamma latent variables. Journal of the American Statistical Association. 2013;108(504):1339–1349.
[25] Aguero-Valverde J, Jovanis PP. Bayesian multivariate poisson lognormal models for crash severity modeling and site ranking. Transportation Research Record. 2009;2136(1):82–91.
[26] Montesinos-López OA, Montesinos-López A, Crossa J, et al. A Bayesian Poisson-lognormal model for count data for multiple-trait multiple-environment genomic-enabled prediction. G3: Genes, Genomes, Genetics. 2017;7(5):1595–1606.
[27] Serhiyenko V, Mamun SA, Ivan JN, et al. Fast Bayesian inference for modeling multivariate crash counts. Analytic Methods in Accident Research. 2016;9:44–53.
[28] Gelman A. Prior distributions for variance parameters in hierarchical models (comment on article by browne and draper). Bayesian Analysis. 2006;1(3):515–534.
[29] Graves TL. Automatic step size selection in random walk metropolis algorithms. arXiv preprint arXiv:11035986. 2011;.
[30] Holden L. Mixing of MCMC algorithms. Journal of Statistical Computation and Simulation. 2019;89(12):2261–2279.
[31] Kass RE, Carlin BP, Gelman A, et al. Markov Chain Monte Carlo in practice: A roundtable discussion. The American Statistician. 1998;52(2):93–100.
[32] Pang WK, Forster JJ, Troutt MD. Estimation of wind speed distribution using Markov Chain Monte Carlo techniques. Journal of Applied Meteorology. 2001;40(8):1476–1484.
[33] Murphy KP. Conjugate Bayesian analysis of the Gaussian distribution. Technical Report. 2007;Available from: https://www.cs.ubc.ca/ murphyk/mypapers.html.
[34] Gilks WR, Wild P. Adaptive rejection sampling for Gibbs sampling. Journal of the Royal Statistical Society: Series C (Applied Statistics). 1992;41(2):337–348.
[35] Geweke J, Tanizaki H. Bayesian estimation of state-space models using the Metropolis–Hastings algorithm within Gibbs sampling. Computational Statistics & Data Analysis. 2001;37(2):151–170.
[36] Halliwell LJ. The log-gamma distribution and non-normal error. Variance: Advancing the Science of Risk. 2018;.
[37] Batir N. On some properties of digamma and polygamma functions. Journal of Mathematical Analysis and Applications. 2007;328(1):452–465.
[38] Mačys J. On the Euler-Mascheroni constant. Mathematical Notes. 2013;94.
[39] Prentice RL. A log-gamma model and its maximum likelihood estimation. Biometrika. 1974;61(3):539–544.
[40] Hoffman MD, Gelman A. The No-U-Turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. Journal of Machine Learning Research. 2014;15(1):1593–1623.
[41] Kucirka LM, Lauer SA, Laeyendecker O, et al. Variation in false-negative rate of reverse transcriptase polymerase chain reaction–based SARS-CoV-2 tests by time since exposure. Annals of Internal Medicine. 2020;.
[42] Fanaee-T H. Bike sharing dataset. UCI Machine Learning Repository. 2013;Available from: https://doi.org/10.24432/C5W894.
[43] Stan Development Team. Stan modeling language users guide and reference manual; 2016.
[44] Nishio M, Arakawa A. Performance of Hamiltonian Monte Carlo and No-U-Turn sampler

for estimating genetic parameters and breeding values. Genetics Selection Evolution. 2019; 51(1):73.
[45] Gelman A. Parameterization and Bayesian modeling. Journal of the American Statistical Association. 2004;99(466):537–545.
[46] Gholiabad SG, Moghimbeigi A, Faradmal J, et al. A multilevel zero-inflated Conway–Maxwell type negative binomial model for analysing clustered count data. Journal of Statistical Computation and Simulation. 2021;91(9):1762–1781.
[47] Liu X, Winter B, Tang L, et al. Simulating comparisons of different computing algorithms fitting zero-inflated poisson models for zero abundant counts. Journal of Statistical Computation and Simulation. 2017;87(13):2609–2621.
[48] Brown S, Duncan A, Harris MN, et al. A zero-inflated regression model for grouped data. Oxford Bulletin of Economics and Statistics. 2015;77(6):822–831.

5. Appendices

Appendix A. Derivation of the approximate conditional posterior

Derivation of the approximate posterior distribution in the posterior distribution is presented below. Terms that do not impact w_{jk} are regarded as a constant, i.e. C_i $(i = 1, \ldots, 5)$ in the following equations.

The conditional posterior of regression coefficient w_{jk} can be written as

$$p\left(w_{jk}|-\right) \propto \exp\left\{ \frac{\left(w_{jk}-\mu_k\right)^2}{-2\sigma_k^2} + \sum_{i=1}^{n_j} \frac{\left[x_{ijk}w_{jk} + \sum\limits_{h=1,h\neq k}^{K} x_{ijh}w_{jh} - \psi_0\left(y_{ij}\right)\right]^2}{-2\psi_1\left(y_{ij}\right)} \right\}. \tag{A1}$$

Let A be the exponent in Eq. (A1) and expand the square terms, then we have

$$A = \frac{\left(w_{jk}-\mu_k\right)^2}{-2\sigma_k^2} + \sum_{i=1}^{n_j} \frac{\left(w_{jk}x_{ijk} + \sum\limits_{h=1,h\neq k}^{K} w_{jh}x_{ijh} - \psi_0\left(y_{ij}\right)\right)^2}{-2\psi_1\left(y_{ij}\right)} \tag{A2}$$

$$= \frac{w_{jk}^2 - 2\mu_k w_{jk} + C_1}{-2\sigma_k^2} + \sum_{i=1}^{n_j} \frac{x_{ijk}^2 w_{jk}^2 + 2\left(\sum\limits_{h=1,h\neq k}^{K} w_{jh}x_{ijh} - \psi_0\left(y_{ij}\right)\right) x_{ijk}w_{jk} + C_2}{-2\psi_1\left(y_{ij}\right)} \tag{A3}$$

$$= -\frac{1}{2}\left(\frac{1}{\sigma_k^2} + \sum_{i=1}^{n_j} \frac{x_{ijk}^2}{\psi_1\left(y_{ij}\right)}\right) w_{jk}^2 + \left(\frac{\mu_k}{\sigma_k^2} + \sum_{i=1}^{n_j} \frac{\left(\sum\limits_{h=1,h\neq k}^{K} w_{jh}x_{ijh} - \psi_0\left(y_{ij}\right)\right) x_{ijk}}{-\psi_1\left(y_{ij}\right)} \right) w_{jk} + C_3. \tag{A4}$$

Dividing the numerator and denominator by the coefficient of the quadratic term, we

get

$$A \propto \frac{w_{jk}^2 + \left(\frac{\frac{\mu_k}{\sigma_k^2} + \sum_{i=1}^{n_j} \frac{\left(\sum_{h=1,h\neq k}^{K} w_{jh} x_{ijh} - \psi_0(y_{ij}) \right) x_{ijk}}{-\psi_1(y_{ij})}}{-\frac{1}{2}\left(\frac{1}{\sigma_k^2} + \sum_{i=1}^{n_j} \frac{x_{ijk}^2}{\psi_1(y_{ij})} \right)} \right) w_{jk}}{\frac{1}{-\frac{1}{2}\left(\frac{1}{\sigma_k^2} + \sum_{i=1}^{n_j} \frac{x_{ijk}^2}{\psi_1(y_{ij})} \right)}} + C_4 \tag{A5}$$

$$\propto \frac{\left(w_{jk} - \frac{\frac{\mu_k}{\sigma_k^2} - \sum_{i=1}^{n_j} \frac{\left(\sum_{h=1,h\neq k}^{K} w_{jh} x_{ijh} - \psi_0(y_{ij}) \right) x_{ijk}}{\psi_1(y_{ij})}}{\frac{1}{\sigma_k^2} + \sum_{i=1}^{n_j} \frac{x_{ijk}^2}{\psi_1(y_{ij})}} \right)^2}{-2 \cdot \frac{1}{\frac{1}{\sigma_k^2} + \sum_{i=1}^{n_j} \frac{x_{ijk}^2}{\psi_1(y_{ij})}}} + C_5. \tag{A6}$$

Therefore, we obtain the mean and variance of the the approximate Gaussian posterior

$$\widehat{\mu}_k = \frac{\frac{\mu_k}{\sigma_k^2} - \sum_{i=1}^{n_j} \frac{\left(\sum_{h=1,h\neq k}^{K} w_{jh} x_{ijh} - \psi_0(y_{ij}) \right) x_{ijk}}{\psi_1(y_{ij})}}{\frac{1}{\sigma_k^2} + \sum_{i=1}^{n_j} \frac{x_{ijk}^2}{\psi_1(y_{ij})}} = \frac{\mu_k + \sigma_k^2 \sum_{i=1}^{n_j} \frac{x_{ijk}}{\psi_1(y_{ij})} \left(\psi_0(y_{ij}) - \sum_{h=1,h\neq k}^{K} w_{jh} x_{ijh} \right)}{\sigma_k^2 \sum_{i=1}^{n_j} \frac{x_{ijk}^2}{\psi_1(y_{ij})} + 1}, \tag{A7}$$

$\forall j = 1, \ldots, J,$

$$\widehat{\sigma}_k^2 = \frac{1}{\frac{1}{\sigma_k^2} + \sum_{i=1}^{n_j} \frac{x_{ijk}^2}{\psi_1(y_{ij})}} = \frac{\sigma_k^2}{\sigma_k^2 \sum_{i=1}^{n_j} \frac{x_{ijk}^2}{\psi_1(y_{ij})} + 1}, \forall j = 1, \ldots, J. \tag{A8}$$

Appendix B. Metrics used for comparing samplers

• ***Effective sample size*** ($\hat{n}_{\text{eff}}$)

For each scalar estimand ψ, the simulations/samples are labeled as ψ_{ij} ($i = 1, \ldots, n; j = 1, \ldots, m$) where n is the number of samples in each chain (sequence) and m is the number of chains. The effective sample size is calculated according to Ref. [11, Chapter 11]

$$\hat{n}_{\text{eff}} = \frac{mn}{1 + 2\sum_{t=1}^{T} \widehat{\rho}_t}, \tag{B1}$$

where the estimated auto-correlations $\widehat{\rho}_t$ are computed as

$$\widehat{\rho}_t = 1 - \frac{V_t}{2\,\widehat{\text{var}}^+} \tag{B2}$$

and T is the first odd positive integer for which $\widehat{\rho}_{T+1} + \widehat{\rho}_{T+2}$ is negative. In Eq. (B2), V_t, the variogram at each lag t, is given by

$$V_t = \frac{1}{m\,(n-t)} \sum_{j=1}^{m} \sum_{i=t+1}^{n} (\psi_{i,j} - \psi_{i-t,j})^2, \tag{B3}$$

and $\widehat{\text{var}}^+$, the marginal posterior variance of the estimand, is given by

$$\widehat{\text{var}}^+ = \frac{n-1}{mn} \sum_{j=1}^{m} s_j^2 + \frac{1}{m-1} \sum_{j=1}^{m} \left(\bar{\psi}_{.j} - \bar{\psi}_{..} \right)^2, \tag{B4}$$

where $s_j^2 = \frac{1}{n-1} \sum_{i=1}^{n} \left(\psi_{ij} - \bar{\psi}_{.j} \right)^2$, $\bar{\psi}_{.j} = \frac{1}{n} \sum_{i=1}^{n} \psi_{ij}$, and $\bar{\psi}_{..} = \frac{1}{m} \sum_{j=1}^{m} \bar{\psi}_{.j}$.

- R^2

The R^2 of generic predicted values $\hat{y}_i, i = 1, \ldots, N$ of the dependent variables $y_i, i = 1, \ldots, N$ is expressed as

$$R^2 = 1 - \frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{N} (y_i - \bar{y})^2}, \tag{B5}$$

where $\bar{y}$ is the average value of $y_i, i = 1, \ldots, N$.

- ***RMSE***

The RMSE of predicted values is given by $\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N} (\hat{y}_i - y_i)^2}{N}}$.