JOURNAL OF STATISTICAL COMPUTATION AND SIMULATION
2023, VOL. 93, NO. 5, 753-774
https://doi.org/10.1080/00949655.2022.2119972

The taxicab sampler: MCMC for discrete spaces with application to tree models

Vincent Geels, Matthew T. Pratola, and Radu Herbei

Department of Statistics, The Ohio State University, Columbus, OH, USA

ABSTRACT
Motivated by the problem of exploring discrete but very complex state spaces in Bayesian models, we propose a novel Markov Chain Monte Carlo search algorithm: the *taxicab sampler*. We describe the construction of this sampler and discuss how its interpretation and usage differs from that of standard Metropolis-Hastings as well as the related Hamming ball sampler. The proposed sampling algorithm is then shown to demonstrate substantial improvement in computation time without any loss of efficiency relative to a naïve Metropolis–Hastings search in a motivating Bayesian regression tree count model, in which we leverage the discrete state space assumption to construct a novel likelihood function that allows for flexibly describing different mean-variance relationships while preserving parameter interpretability compared to existing likelihood functions for count data.

ARTICLE HISTORY
Received 1 March 2022
Accepted 29 August 2022

KEYWORDS
Discrete state spaces, Markov Chain Monte Carlo, regression trees, count data, data augmentation

CONTACT Vincent Geels geels.1@osu.edu

1. Introduction

Bayesian models often use continuous latent variable formulations in problems involving discrete spaces. This approach stems from the work of [1], who handled the problem of analysing binary and polychotomous data within a

Bayesian framework via the introduction of Gaussian latent variables; the resulting closed-form conditional posterior distributions facilitated fast parameter updating. More recent examples of analogous continuous latent variable approaches in other discrete settings (namely, modelling count response data) may be found in [2], who describe a latent approach for parameter updating involving power series likelihoods, and [3], who utilizes Pólya–Gamma mixtures to obtain closed-form updates in the zero-inflated (ZI) negative binomial regression setting. While powerful, viable implementations of a continuous latent variable mechanism are not always straightforward in certain types of discrete problems. Additionally, these formulations often suffer from the fact that the latent variable dimension grows with the sample size n.

An alternative approach is to work with discrete spaces directly; a popular example of this may be seen in Bayesian regression tree models (BRTs), where, for instance, the split rules for continuous predictors are modelled using a discrete candidate set. Still, this choice comes with its own difficulties, as the posterior tree space in such models is discrete, complex, and large. The resulting issue of poor mixing in BRTs is a known one in the literature [4–6], although recent progress has been made [7,8].

With these opposing perspectives in mind, we consider the general approach of approximating continuous spaces via discretization, a strategy that allows for the replacement of integrals with sums when marginalization is required and often simplifies the problem of identifying modes in probability distributions over compact sets. O'Hagan et al. [9] leverage this discretization strategy, demonstrating a simple yet practical approach to fitting a Bayesian logistic GLM using improper uniform priors. The conditional posterior distributions of interest are subsequently discretized over a finite grid in order to obtain a closed-form posterior that allows for approximate but straightforward sampling. A more recent example may be found in the machine learning literature for continuous control problems, where [10] propose discretizing high-dimensional continuous action spaces and sequentially obtain actions that conditionally maximize a target action-value function one dimension at a time.

In this paper, we argue for *working with discrete spaces directly* and propose an MCMC algorithm, hereafter referred to as the *taxicab sampler* (TC sampler), a nod to the way in which the sampler traverses the state space. The TC sampler extends the Hamming ball sampler (HBS) previously formulated by Titsias and Yau [11] and can be applied in scenarios where the latter algorithm cannot. Our proposed algorithm is readily applicable to high-dimensional discrete spaces where a natural distance exists, e.g. the L_∞ distance in our example count model application. Importantly, as with the Hamming ball sampler, the taxicab sampler allows for marginalization over 'slices' of the model space, in effect resulting in Gibbs sampling from conditional posterior distributions of interest. A key advantage of the sampler formulation lies in its ability to leverage a computationally advantageous local search approach over a discrete space compared to a more costly global search strategy or an inefficient Metropolis–Hastings (MH) proposal scheme. As shall be seen, the TC sampler delivers improved computational efficiency over a naïve MH implementation without compromising inference.

Note that while other 'locally-informed' MCMC samplers have recently been proposed for searching over discrete state spaces [12–14], their intended usage is similar to that of the Hamming ball sampler – e.g. variable selection, weighted permutations, and/or Ising models. As such they differ substantially from our focus on count models, which typically contain an intrinsic ordering with respect to a model dimension not found in these other discrete space problems.

The remaining sections of this paper are organized as follows. In Section 2, we detail the TC sampler mechanism, its role in a model update algorithm, and discuss considerations for dimension-changing proposals when utilizing the sampler. In Section 3, we describe a single-tree model designed to handle count data involving a novel count likelihood function utilizing discrete parameter spaces. In Section 4, we demonstrate the TC sampler's statistical and computational efficiency in simulation experiments when fitting this single-tree model compared to a straightforward MH approach. Section 5 offers discussion and directions for future work.

2. The taxicab sampler

For a sample of size n, we work with observed responses $\mathbf{Y} = \{Y_i\}_{i=1}^n \in \mathbb{R}^n$, along with observed covariate matrix $\mathbf{X} = (x_{iv}) \in \mathbb{R}^{n \times p}$, where each $\mathbf{X}_i = (x_{i1}, \ldots, x_{ip})$ is the row vector of covariates measured along with the ith response Y_i. We reserve the use of the notation $\mathbf{x}_v$, $v = 1, \ldots, p$, to exclusively describe the vth covariate dimension of $\mathbf{X}$ when discussing certain aspects of tree structures in Section 3. In general, we also use the notation $\mathbf{Y}_{-h} = (Y_1, \ldots, Y_{h-1}, Y_{h+1}, \ldots, Y_n)$ for some vector $\mathbf{Y} \in \mathbb{R}^n$, and we use the square bracket notation $[Z]$ to denote the distribution of a generic random variable Z.

As described in Section 1, our focus is on developing technology with which we may fit models depending (in part or in whole) on parameter spaces restricted to countable subsets of $\mathbb{R}$. We assume a Bayesian setup where inference is desired on some discrete parameter vector $\boldsymbol{\lambda} \in \mathbb{Z}^B$. Choices other than $\mathbb{Z}^B$ are possible. All other model parameters are collected in the vector $\boldsymbol{\theta} \in \Theta$. For the purpose of exposition, we assume we work with a joint probability distribution $p(\mathbf{Y}, \boldsymbol{\theta}, \boldsymbol{\lambda} \,|\, \mathbf{X})$, where $\mathbf{X}$ is treated as fixed. The unnormalized joint posterior distribution for $(\boldsymbol{\theta}, \boldsymbol{\lambda})$ factorizes as

$$\pi(\boldsymbol{\theta}, \boldsymbol{\lambda} \,|\, \mathbf{Y}, \mathbf{X}) \propto p(\mathbf{Y} \,|\, \boldsymbol{\theta}, \boldsymbol{\lambda}, \mathbf{X}) \pi(\boldsymbol{\theta}, \boldsymbol{\lambda}). \tag{1}$$

To explore the discrete parameter spaces associated with $\boldsymbol{\lambda}$ during model-fitting, we introduce a fundamental tool called the taxicab sampler. Our sampler takes motivation from the Hamming ball sampler [11] a generic MCMC sampling procedure for high-dimensional discrete-state models that allows for iterative sampling from 'slices' of the model space that are generated via Hamming balls. These Hamming balls are constructed using the Hamming distance, which is defined between two vectors $\mathbf{w}, \mathbf{u}$ as

$$d_H(\mathbf{w}, \mathbf{u}) = \textstyle\sum_b \mathbb{I}\{w_b \neq u_b\}. \tag{2}$$

These balls have finite cardinality in settings where $\mathbf{w}$ and $\mathbf{u}$ take on a finite number of configurations, and in such cases they allow for tractable computation of normalizing constants in order to sample from slices of the target conditional posterior distributions. The result is a powerful yet simple technology that allows for Gibbs step updates on parameters or latent variables in problems that would otherwise require more cumbersome MH updates.

As constructed, the HBS is ideal for performing Bayesian inference via MCMC on binary sequences or matrices [11]. However, the Hamming distance is less suitable in other discrete-valued discrete state space settings, since enumerating all set elements generated by a Hamming ball becomes significantly more expensive in non-binary settings and intractable when the probability distribution of one or more random variables in the matrix or sequence places positive probability on infinitely many values.

This point is made plain using a simple univariate example: suppose we have a probabilty mass function (pmf) $p(\mathbf{Y} \mid \lambda)$ and prior distribution $\pi(\lambda)$ with support on $\mathbb{Z}$, such that $\pi(\lambda \mid \mathbf{Y})$ is unavailable in closed form. Introducing a latent variable U to facilitate Hamming ball sampling on λ requires construction of an auxiliary distribution $p(U \mid \lambda) = \frac{\mathbb{I}\{U \in \mathscr{H}_m(\lambda)\}}{|\mathscr{H}_m(\lambda)|}$, where $|\cdot|$ denotes set cardinality. Now $\mathscr{H}_m(\lambda) = \{U \in \mathbb{Z} : d_H(U, \lambda) \leq m\} = \{U \in \mathbb{Z} : U \neq \lambda\}$, so that $p(U \mid \lambda)$ amounts to an improper uniform distribution on $\mathbb{Z} \setminus \lambda$ and no longer provides an efficient means to aid in sampling the target posterior distribution $\pi(\lambda \mid \mathbf{Y})$.

2.1. The taxicab sampler algorithm

For discrete state spaces of large or infinite cardinality, the L_∞ distance allows for straightforward generation of L_∞ neighbourhoods of the desired radius, so that marginalization of target distributions over all set elements contained within the neighbourhood is tractable. With this idea in mind, we construct in this section a TC sampler that relies on use of sets generated within an L_∞ neighbourhood, centred at a vector $\mathbf{w}$ with some specified radius *m*, i.e.

$$\mathscr{N}_m(\mathbf{w}) = \{\mathbf{u} : \sup_b |u_b - w_b| \leq m\}, \quad m \geq 0, \tag{3}$$

to construct a transition kernel for an augmented bivariate chain. Algorithm 1 outlines the general TC sampler update algorithm. At its core the TC sampler is a data augmentation scheme requiring the injection of a vector of auxiliary variables $\mathbf{U}$ (possessing the same dimension as $\boldsymbol{\lambda}$) into the joint probability model in order to explore slices of the conditional posterior distribution for $\boldsymbol{\lambda}$; the sampler construction assumes a distribution for $\mathbf{U}$ that depends only on $\boldsymbol{\lambda}$, $p(\mathbf{U} \mid \boldsymbol{\lambda})$. This allows us to factorize the augmented joint probability model as

$$p(\mathbf{Y}, \boldsymbol{\theta}, \boldsymbol{\lambda}, \mathbf{U} \mid \mathbf{X}) = p(\mathbf{Y}, \boldsymbol{\theta}, \boldsymbol{\lambda} \mid \mathbf{X}) p(\mathbf{U} \mid \boldsymbol{\lambda}). \tag{4}$$

In this manuscript, we assume a uniform distribution

$$p(\mathbf{U} \mid \boldsymbol{\lambda}) = \frac{\mathbb{I}\{\mathbf{U} \in \mathscr{N}_{m_\lambda}(\boldsymbol{\lambda})\}}{Z_{m_\lambda}}, \tag{5}$$

where Z_{m_λ} is the cardinality of $\mathscr{N}_{m_\lambda}(\boldsymbol{\lambda})$; however, this choice of distribution is not critical.

```
Algorithm 1: The general taxicab sampler update algorithm.
  Data: Realized observations $(Y_1, \mathbf{X}_1), \ldots, (Y_n, \mathbf{X}_n)$, starting values $(\boldsymbol{\theta}^{(0)}, \boldsymbol{\lambda}^{(0)}, \mathbf{U}^{(0)})$
  Result: Approximate posterior samples drawn from
          $\pi(\boldsymbol{\theta}, \boldsymbol{\lambda}, \mathbf{U} \mid (Y_1, \mathbf{X}_1), \ldots, (Y_n, \mathbf{X}_n))$
1 for t in $1 : N_{mcmc}$ iterations do
2     Update $\boldsymbol{\theta}^{(t)} \leftarrow \boldsymbol{\theta}^{(t-1)}$ using existing MCMC technology
3     Draw $\mathbf{U}^{(t)} \sim p(\mathbf{U}^{(t)} \mid \mathcal{N}_{m_\lambda}(\boldsymbol{\lambda}^{(t-1)})) := p(\mathbf{U}^{(t)} \mid \boldsymbol{\lambda}^{(t-1)})$
4     Draw $\boldsymbol{\lambda}^{(t)} \sim q_{m_\lambda}(\boldsymbol{\lambda}^{(t)} \mid \mathbf{Y}, \boldsymbol{\theta}^{(t)}, \mathcal{N}_{m_\lambda}(\mathbf{U}^{(t)}), \mathbf{X}) := q_{m_\lambda}(\boldsymbol{\lambda}^{(t)} \mid \mathbf{Y}, \boldsymbol{\theta}^{(t)}, \mathbf{U}^{(t)}, \mathbf{X})$
```

The key idea of the auxiliary vector $\mathbf{U}$ is that it facilitates a speedier and more efficient exploration of the conditional posterior distribution of $\boldsymbol{\lambda}$: given some realization $\mathbf{U} = \mathbf{u}$, we may now compute in closed form the slice of the conditional posterior distribution of $\boldsymbol{\lambda}$ captured by $\mathscr{N}_{m_\lambda}(\mathbf{u})$ via

$$q_{m_\lambda}(\boldsymbol{\lambda} \mid \mathbf{Y}, \boldsymbol{\theta}, \mathbf{U} = \mathbf{u}, \mathbf{X}) = \frac{\pi(\boldsymbol{\lambda} \mid \mathbf{Y}, \boldsymbol{\theta}, \mathbf{X})\mathbb{I}\{\boldsymbol{\lambda} \in \mathcal{N}_{m_\lambda}(\mathbf{u})}{\sum_{\boldsymbol{\lambda}^* \in \mathbb{Z}^B} \pi(\boldsymbol{\lambda}^* \mid \mathbf{Y}, \boldsymbol{\theta}, \mathbf{X})\mathbb{I}\{\boldsymbol{\lambda}^* \in \mathcal{N}_{m}} \tag{6}$$

$$= \frac{p(\mathbf{Y}, \boldsymbol{\theta}, \boldsymbol{\lambda} \mid \mathbf{X})\mathbb{I}\{\boldsymbol{\lambda} \in \mathcal{N}_{m_\lambda}(\mathbf{u})\}}{\sum_{\boldsymbol{\lambda}^* \in \mathcal{N}_{m_\lambda}(\mathbf{u})} p(\mathbf{Y}, \boldsymbol{\theta}, \boldsymbol{\lambda}^* \mid \mathbf{X})}. \tag{7}$$

It is the tractable normalization constant in (7) that allows for calculation of these slices of the conditional posterior distribution of $\boldsymbol{\lambda}$, as it only requires summation over Z_{m_λ} total terms. Thus, in contrast to a MH sampler, there is no 'accept-reject' mechanism; rather, $\boldsymbol{\lambda}$ is *drawn* from q_{m_λ} above. By construction, higher-probability regions in these conditional posterior slices are visited more frequently by q_{m_λ}, while the auxiliary proposal distribution (5) encourages random exploration of the augmented space via uniform moves within the generated neighbourhood. From this sampler construction, we see that the choice of m_λ controls the trade-off between computational speed and degree of exploration of the target space, with larger choices of radius facilitating the better exploration of the target space of interest and smaller choices of radii allowing for faster computation speed.

The complete TC sampler may now be assembled according to the following (ordered) two-part Gibbs step:

$$\text{1. draw } \mathbf{U} \sim p(\mathbf{U} \mid \boldsymbol{\lambda}). \tag{8}$$

$$\text{2. draw } \boldsymbol{\lambda} \sim q_{m_\lambda}(\boldsymbol{\lambda} \mid \mathbf{Y}, \boldsymbol{\theta}, \mathbf{U}, \mathbf{X}). \tag{9}$$

In this two-part procedure, we observe that the TC sampler replaces individual draws of $\boldsymbol{\lambda}$ with draws of $(\mathbf{U}, \boldsymbol{\lambda})$ from the augmented chain

$$\ldots, \begin{pmatrix} \mathbf{U}^{(t)} \\ \boldsymbol{\lambda}^{(t)} \end{pmatrix}, \begin{pmatrix} \mathbf{U}^{(t+1)} \\ \boldsymbol{\lambda}^{(t+1)} \end{pmatrix}, \begin{pmatrix} \mathbf{U}^{(t+2)} \\ \boldsymbol{\lambda}^{(t+2)} \end{pmatrix}, \ldots,$$

where the states are traversed via the joint transition kernel

$$[\boldsymbol{\lambda}^{(t+1)}, \mathbf{U}^{(t+1)} \mid \boldsymbol{\lambda}^{(t)}, \mathbf{U}^{(t)}] = [\mathbf{U}^{(t+1)} \mid \boldsymbol{\lambda}^{(t)}][\boldsymbol{\lambda}^{(t+1)} \mid \mathbf{Y}, \boldsymbol{\theta}, \mathbf{U}^{(t+1)}, \tag{10}$$

Figure 1 illustrates how (10) traverses the augmented state space in the simple case where $B = 1$ and $m_\lambda = 1$.

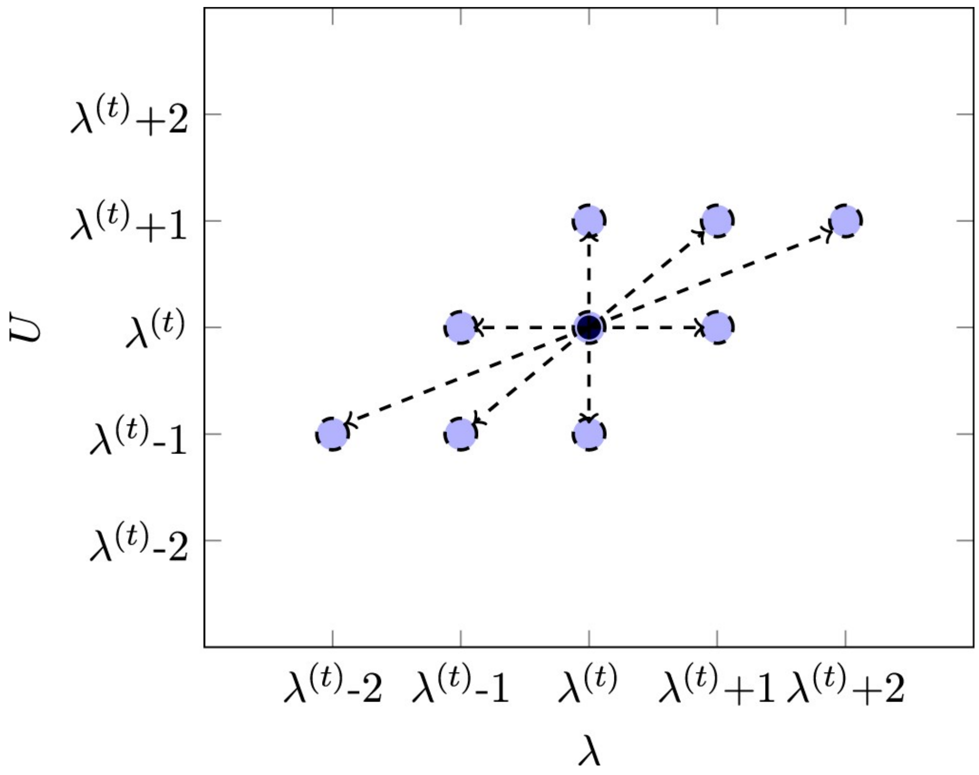


Figure 1. Representation of valid bivariate proposal states $(U^{(t+1)}, \lambda^{(t+1)})$ (dashed blue circles) from state originating at $(U^{(t)}, \lambda^{(t)}) = (\lambda^{(t)}, \lambda^{(t)})$ (solid black circle) via the bivariate kernel (10) with radius $m_\lambda = 1$ when $B = 1$. Dashed directional arrows connect the origin to each proposed state. Note that $(U^{(t+1)}, \lambda^{(t+1)}) = (\lambda^{(t)}, \lambda^{(t)})$ is a valid proposal state.

From the above construction, we see the TC sampler confers certain advantages over MH proposals and direct draws from $\pi(\boldsymbol{\lambda} \mid \mathbf{Y}, \boldsymbol{\theta}, \mathbf{X})$ when $\boldsymbol{\lambda}$ is high-dimensional or requires an infinite sum to obtain the normalization constant, since q_{m_λ} is supported on a (typically) smaller set. As a result, the TC sampler provides a flexible balance between local and global search strategies via a choice of m_λ.

As mentioned above, in the TC sampler, the accept/reject mechanism is replaced by a draw from the distribution q_{m_λ} displayed in (7). This strategy allows the TC sampler to *move all the time* within the appropriate neighbourhood. In our simulations, we find that the local probability renormalization mechanism leads this sampler to visit low-probability states far more efficiently than the MH sampler. Although m_λ plays the role of a tuning parameter, its effects on the efficiency of the proposed sampler are far less extreme than the tuning parameter in a MH sampler. To illustrate these ideas, we include in Section B of the Supplementary Materials a challenging example scenario involving an infinite but countable state space under a highly multimodal distribution. The experimental results show how the TC sampler explores the state space faster and with superior recovery of the underlying target distribution relative to a MH sampler. Figure 2 summarizes this scenario, showing that the TC sampler quickly converges under the Hellinger distance, whereas the MH sampler does so at a much slower rate. We include a similar comparison in terms of the total variation distance in Section B of the Supplementary Materials.

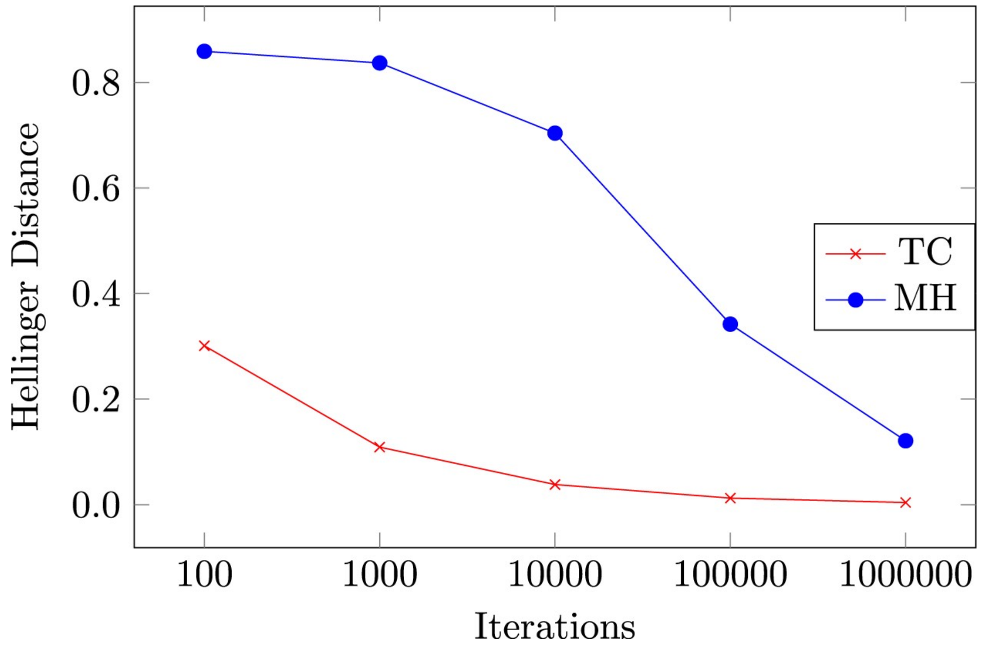


Figure 2. Mean Hellinger distance (based on 100 samples) at selected iterations between estimated and target distributions under the TC and MH samplers for the example scenario presented in Supplementary Materials Appendix B.

The TC sampler also allows for sequential updating of $\boldsymbol{\lambda}$: if $\boldsymbol{\lambda}$ is organized into blocks $\boldsymbol{\lambda}_p$, $p = 1, \ldots, P$, the ordered two-part Gibbs steps are now

$$1.\ \text{draw } \mathbf{U}_p \sim p(\mathbf{U}_p \,|\, \boldsymbol{\lambda}_p), \tag{11}$$

$$2.\ \text{draw } \boldsymbol{\lambda}_p \sim q_{m_\lambda}(\boldsymbol{\lambda}_p \,|\, \mathbf{Y}, \boldsymbol{\theta}, \mathbf{U}_p, \boldsymbol{\lambda}_{-p}, \mathbf{X}), \tag{12}$$

for all p, where (11) depends on $Z_{m_\lambda,p} = |\mathscr{N}_{m_\lambda}(\boldsymbol{\lambda}_p)|$.

Finally, we note that integrating out $\mathbf{U}$ in (4) trivially recovers the non-augmented joint probability distribution. Further, choosing $m_\lambda > 0$ is necessary for ergodicity of the induced Markov chain under the TC sampler. With these results, the constructed TC sampler guarantees that the corresponding *marginal* sampler on $\boldsymbol{\lambda}$ has as its stationary distribution $\pi(\boldsymbol{\lambda} \,|\, \mathbf{Y}, \boldsymbol{\theta}, \mathbf{X})$ under reasonable conditions; we refer the reader to the Supplementary Materials for the associated proofs.

2.2. Considerations for dimension-changing proposals using the taxicab sampler

The strength of the TC sampler lies in its simplicity, and it affords the ability to circumvent larger marginalization problems when performing inference on discrete model parameters; however, the sampler introduces a different set of issues when we consider more complicated models and model-fitting algorithms that involve dimension-changing proposals. In such settings, we must contend with the auxiliary vector $\mathbf{U}$ injected into the model through the TC sampler, along with the fact that the model specification may lack a conjugate relationship between the likelihood function and discrete parameter prior distributions.

We point out that the problem of dimension-changing proposals in models using the TC sampler has a different interpretation than in models using the HBS: the latter naturally allows for jumping between models encoded into binary vectors during the fitting procedure, such that the HBS proposes jumps between (potentially much) smaller- and (potentially much) larger-dimensional models with equal probability during the $\mathbf{U}$-sampling step and with no additional tools required in the proposal construction. By contrast, natural use cases for the TC sampler are more likely to involve an intrinsic ordering associated with models of differing dimension (e.g. we prefer proposing jumps in model dimension within ± 3 of the current state before proposing jumps in model dimension within ± 10). As such, we view dimension-jumping proposals using the TC sampler as an additional focal point worthy of discussion.

Fortunately, the construction of reasonable proposal distributions in these types of problems is mediated to a degree by the fact that we work with discrete parameter spaces instead of continuous ones, and as a result the dimension-change in these kinds of augmented chains may be handled by standard discrete-state Markov chain concepts [15]. Still, proposal construction may require more care in model frameworks where traversing models of differing dimension is non-trivial, e.g. in regression trees, and we construct one possible proposal function for tree birth and death moves within the BRT framework in Section 4.

When non-conjugacy rules out the ability to efficiently marginalize out the dimension-changing parameters, we propose handling these jump proposals via a MH step. To illustrate the construction of this type of move, we consider a simple setting in which the model dimension B in (4) is assigned a prior distribution $\pi(B)$ such that $\boldsymbol{\lambda} \sim \pi(\boldsymbol{\lambda} \mid B)$, so that our augmented joint probability model is now written as

$$p(\mathbf{Y}, \boldsymbol{\theta}, \boldsymbol{\lambda}, \mathbf{U}, B \mid \mathbf{X}) = p(\mathbf{Y} \mid \boldsymbol{\theta}, \boldsymbol{\lambda}, B, \mathbf{X})\pi(\boldsymbol{\theta})\pi(\boldsymbol{\lambda} \mid B)\pi(B)p(\mathbf{U} \tag{13}$$

where $\boldsymbol{\theta}$ may or may not depend on B. In this context, we are interested in proposing a move from the current model state associated with B to a new model state associated with dimension B'. Within the overall transition kernel $(\boldsymbol{\theta}, \boldsymbol{\lambda}, \mathbf{U}, B) \to (\boldsymbol{\theta}', \boldsymbol{\lambda}', \mathbf{U}', B')$, we are primarily concerned with construction of the conditional proposal density $(\boldsymbol{\lambda}, \mathbf{U} \mid B', \boldsymbol{\theta}') \to (\boldsymbol{\lambda}', \mathbf{U}' \mid B', \boldsymbol{\theta}')$ and so focus our attention on this proposal component.

In particular, taking $(\boldsymbol{\lambda}, \mathbf{U} \mid B', \boldsymbol{\theta}') \rightarrow (\boldsymbol{\lambda}', \mathbf{U}' \mid B', \boldsymbol{\theta}')$ to be (10), suitably modified to handle the change in dimension, leads to an interesting connection with the common strategy of marginalizing over continuous dimension-changing parameters in reversible-jump MCMC (RJMCMC). To generate the proposed state $(\boldsymbol{\lambda}', \mathbf{U}')$, we utilize the following order-dependent sequence of steps:

1. Draw $\mathbf{a} \sim p(\mathbf{a})$.
2. Generate $\mathbf{U}' = \delta(\boldsymbol{\lambda}, \mathbf{a})$.
3. Draw $\boldsymbol{\lambda}' \sim q_{m_\lambda}(\boldsymbol{\lambda}' \mid \mathbf{Y}, \boldsymbol{\theta}', \mathbf{U}', B', \mathbf{X})$.

Here, $\mathbf{a}$ is a random vector of dimension $B' - B$ distributed according to $p(\mathbf{a})$ that satisfies the dimension-matching requirement of the jump proposal [16, hereafter referred to as GR95]. After drawing $\mathbf{a}$, we may deterministically generate $\mathbf{U}'$ from $(\boldsymbol{\lambda}, \mathbf{a})$ via a map $\delta : \mathbb{Z}^{B'} \rightarrow \mathbb{Z}^{B'}$ such that $\delta(\boldsymbol{\lambda}, \mathbf{a}) = \mathbf{U}'$; we require δ be invertible in order to ensure reversibility in the resulting Markov chain. While this choice of conditional transition function depends on $\mathbf{Y}$, it leads to useful cancellations in the resulting acceptance probability calculation, which simplifies to

$$\begin{aligned}\alpha[(\boldsymbol{\theta}, \boldsymbol{\lambda}, \mathbf{U}, B), (\boldsymbol{\theta}', \boldsymbol{\lambda}', \mathbf{U}', B')] =& \min\{1, A\},\\ A = \frac{\left[\sum_{\boldsymbol{\lambda}^* \in \mathscr{N}_{m_\lambda}(\mathbf{U}')} p(\mathbf{Y} \mid \boldsymbol{\theta}', \boldsymbol{\lambda}^*, B') \pi(\boldsymbol{\theta}', \boldsymbol{\lambda}^*, B')\right] p(\mathbf{U}' \mid \boldsymbol{\lambda}') q}{\left[\sum_{\tilde{\boldsymbol{\lambda}} \in \mathscr{N}_{m_\lambda}(\mathbf{U})} p(\mathbf{Y} \mid \boldsymbol{\theta}, \tilde{\boldsymbol{\lambda}}, B) \pi(\boldsymbol{\theta}, \tilde{\boldsymbol{\lambda}}, B)\right] p(\mathbf{U} \mid \boldsymbol{\lambda}) p(\mathbf{a}) q(}\end{aligned} \tag{14}$$

and we see via (14) that our choice of proposal function leads to an acceptance probability ratio that weighs moving into the proposed model slice captured by $\mathscr{N}_{m_\lambda}(\mathbf{U}')$ from the current model slice captured by $\mathscr{N}_{m_\lambda}(\mathbf{U})$. It is in (14) that we see a correspondence with the GR95 approach of integrating out continuous dimension-changing variables in reversible-jump moves: here, if we allow $m_\lambda \rightarrow \infty$, in the limit we obtain $\mathscr{N}_{m_\lambda}(\mathbf{U}) = \mathbb{Z}^B$ and the numerator and denominator marginal likelihood functions in (14) are now fully marginalized over the discrete dimension-changing vector $\boldsymbol{\lambda}$.

We do not claim this choice of transition function is optimal in some sense for a dimension-changing proposal, and for brevity also do not consider more flexible non-deterministic reverse transitions in this subsection. We refer the interested reader to [17] for an overview of strategies to construct proposal functions in RJMCMC.

3. A single-tree model for count data

In this section, we introduce a single-tree model for count data using a novel data distribution to serve as the likelihood function. Its advantages include easily-interpretable parameters along with the ability to handle data that displays under-, equi-, and over-dispersion. The corresponding priors encode sensible and desirable regularization and a zero-inflated extension is straightforward. Notably, this likelihood function is not in the exponential family and the selected prior distributions are not conjugate.

To place our proposed model in context, we begin with brief reviews of Bayesian regression tree models and models designed to handle count data in Sections 3.1 and 3.2, respectively. We introduce our proposed model in Section 3.3.

3.1. A review of Bayesian regression tree models

3.1.1. Regression tree models

Each binary decision tree $\mathscr{T}$ is comprised of vertices (which we refer to as nodes or η's), edges, and splitting rules. Nodes positioned at the terminus of a tree branch are labelled *terminal* nodes and equipped with parameters that describe the data associated with them; non-terminal nodes are labelled *internal* nodes and equipped with binary rules. An internal node η may also be referred to as a *parent* node; the connected nodes below it are referred to as left and right *child* nodes based on their relative position to the parent. The left panel of Figure 3 visualizes an example tree $\mathscr{T}$ with parent and child nodes labelled.

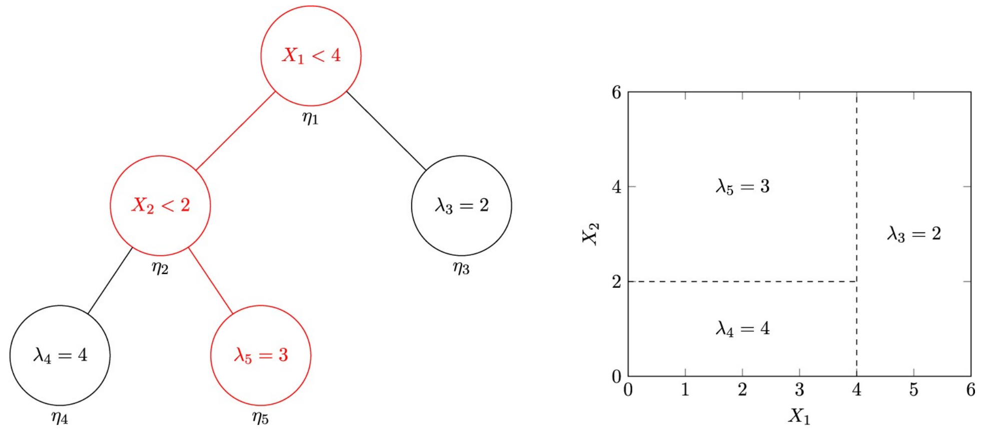


Figure 3. Left: An example tree $(\mathscr{T}, \mathbf{M})$ containing two internal nodes (η_1, η_2) with rules and three terminal nodes (η_3, η_4, η_5) with associated parameters. The example vector $(x_1, x_2) = (3, 3)$ would be sorted along the path highlighted by the red nodes and collected in η_5; the corresponding path from root node to η_5 is represented by $\mathscr{X}_5 = \{x_1 < 4 \cap x_2 \geq 2\}$. Right: an equivalent representation of $(\mathscr{T}, \mathbf{M})$ demonstrating how the tree structure partitions the covariate space (X_1, X_2) into hyperrectangles and assigns parameter values to each. Here $\mathscr{X}_5$ corresponds to the top-left hyperrectangle with parameter value $\lambda_5 = 3$.

A regression tree $\mathscr{T}$ partitions the covariate space according to the binary internal node rules that comprise its interior. These internal node rules are often equivalently referred to as *decision* or *splitting* rules and may be compactly described using the notation (v, c_v), where $v \in \{1, \ldots, p\}$ corresponds to the *v*th dimension of the covariate space, and $c_v \in \mathscr{C}_v$ corresponds to a comparison value or set in the *v*th dimension, with the relation $(v, c_v) = \{x_{iv} < c_v\}$ or $(v, c_v) = \{x_{iv} \in c_v\}$.

In tree models, $|\mathscr{C}_v|$ is finite since each covariate dimension is typically discretized over an equally spaced grid according to some resolution in order to ensure the tree space does not become prohibitively difficult to explore. This grid resolution is specified via a user-selected number of 'cutpoints' ζ_v for any numeric covariate $\mathbf{x}_v$, and otherwise $\zeta_v = \dim(\mathbf{x}_v)$ for categorical covariates (however, other encodings are possible – see, for instance, [18]). The discretization is based on the observed range of covariate values so that in the *v*th dimension, each cutpoint c_v is such that $c_v \in (\min_i\ (x_{iv}), \max_i\ (x_{iv}))$ if $\mathbf{x}_v$ is numeric, or else a subset of categories $c_v \subset \mathbf{x}_v$ if $\mathbf{x}_v$ is categorical.

Specifically, a decision rule (v, c_v) operates on a continuous covariate vector $\mathbf{X}_i$ as follows. The internal node rule (v, c_v) evaluates the event $\{x_{iv} < c_v\}$: if $\{x_{iv} < c_v\}$ is true, $\mathbf{X}_i$ is deterministically assigned to the left child node; otherwise, $\mathbf{x}_i$ is assigned to the right child node. Each $\mathbf{X}_i$ is sorted along a resultant path of internal nodes in this fashion until a terminal node η is reached, at which point it is collected in η. To each terminal node, we also assign a value λ_b where $b = 1, \ldots, B$ indexes a terminal node. In this framework, the tree structure defines a function $g : \mathbb{R}^p \to \mathbb{R}$. If covariate $\mathbf{X}_i$ is assigned to terminal node η_b, then we set

$$g(\mathbf{X}_i) = \lambda_b.$$

In a probabilistic tree model, $\mathscr{T}$ may be used to flexibly describe statistical models for data $(Y_i, \mathbf{X}_i)$. Using the notation described above, let $g(\cdot)$ denote the function described by the tree $\mathscr{T}$. The values $\mathbf{M} = (\lambda_1, \ldots, \lambda_B)$ assigned to the terminal nodes in $\mathscr{T}$ are viewed as parameters and $g(\cdot)$ is the regression function. This model is summarized as

$$\begin{aligned} &\mathbf{Y} \mid \mathbf{X}, \mathbf{M}, \boldsymbol{\theta} \sim f(\cdot \mid \mathbf{X}, \mathbf{M}, \boldsymbol{\theta}), \quad \text{where} \\ &\mathbb{E}(Y_i \mid \mathbf{X}_i, \mathbf{M}) = g(\mathbf{X}_i) = \sum_{b=1}^{B} \lambda_b \mathbb{I}(\mathbf{X}_i \in \mathscr{X}_b), \end{aligned} \tag{15}$$

where *f* is a generic notation for a likelihood function selected by the user and $\mathscr{X}_b$ represents the intersection of the regions described by the internal nodes in $\mathscr{T}$ that form the path from the root node to terminal node η_b. The right panel of Figure 3 shows how such a tree $(\mathscr{T}, \mathbf{M})$ partitions and describes the covariate space according to (15). Further assumptions (conditional independence, for example) will allow us to write the model (15) in a simplified, more manageable form, as seen in the sections to come. In addition to the parameter vector $\boldsymbol{\lambda}$ that is used to describe the mean function, we allow for other parameters, collected in $\boldsymbol{\theta}$, to be a part of the probabilistic model (15). Such parameters may model other components of the model, such as the variance if using a Gaussian likelihood.

3.1.2. Bayesian CART

Chipman et al. [4] (hereafter referred to as CGM98) introduce the Bayesian analogue of the CART model along with the MH analogue of [19]'s CART algorithm for fitting tree models.

3.1.2.1 Likelihood function and prior distribution specification in Bayesian CART In the Bayesian CART model, the joint prior distribution $\pi(\mathscr{T}, \mathbf{M}, \boldsymbol{\theta})$ is assumed to factorize as

$$\pi(\mathscr{T}, \mathbf{M}, \boldsymbol{\theta}) = \pi(\mathbf{M} \mid \mathscr{T})\pi(\mathscr{T})\pi(\boldsymbol{\theta}), \tag{16}$$

$$\pi(\mathbf{M} \mid \mathscr{T}) = \prod_{b=1}^{B} \pi(\lambda_b \mid \mathscr{T}), \tag{17}$$

for the terminal node parameters λ_b, allowing for a prior distribution on the tree topology $\mathscr{T}$ that does not depend on $\mathbf{M}$ and codifying the assumption of a priori independence of terminal node parameters given $\mathscr{T}$. Typically, conjugate prior distributions are assigned to the λ_b's and $\boldsymbol{\theta}$.

Notably, CGM98 specify $\pi(\mathscr{T})$ implicitly. Tree complexity is controlled via the prior probability that a node η will be internal/non-terminal, defined as

$$\begin{aligned} \pi(\eta \text{ is internal}) &= \alpha(1 + d(\eta))^{-\beta}, \quad \alpha \in (0,1),\, \beta > 0, \\ \pi(\eta \text{ is terminal}) &= 1 - \pi(\eta \text{ is internal}). \end{aligned}$$

Each splitting rule (v, c_v) is determined by the choice of available covariates $\mathbf{x}_v$, where v is selected uniformly among all available predictors, and the choice of cutpoint value (or category) c_v is also selected uniformly among the available generated cutpoints given v and $\mathscr{T} \setminus \eta$.

3.1.2.2 Fitting the Bayesian CART model Fitting a Bayesian CART model requires mechanisms for exploring the conditional posterior distributions for $\mathscr{T}$, $\mathbf{M}$, and $\boldsymbol{\theta}$. Conveniently, the choice of conjugate prior distributions for the terminal node parameters collected in $\mathbf{M}$ as well as $\boldsymbol{\theta}$ results in straightforward Gibbs updates for these model parameters.

CGM98 propose searching over the tree space via the use of a top-down stochastic tree-generating process. Broadly, trees (beginning with a single root node) are grown by first proposing a *birth* move (selected with probability $\pi(\text{birth})$) and then uniformly selecting a terminal node to be split; tree pruning occurs via *death* moves (selected with probability $\pi(\text{death})$) performed on a uniformly-selected next-to-terminal node. These moves are illustrated in Figure 4. A successful birth move converts the selected terminal node η into an internal node, which is then assigned a splitting rule and left and right child (terminal) nodes. A successful death move reverses this birth process. Exploration of the posterior tree space in the CGM98 model formulation is also facilitated by two additional proposals, swap and change; proposals affecting $\mathscr{T}$ are handled via MH steps after integrating out the dimension-changing mean vector $\mathbf{M}$, made possible due to the choice of conjugate prior distributions on the λ_b's.

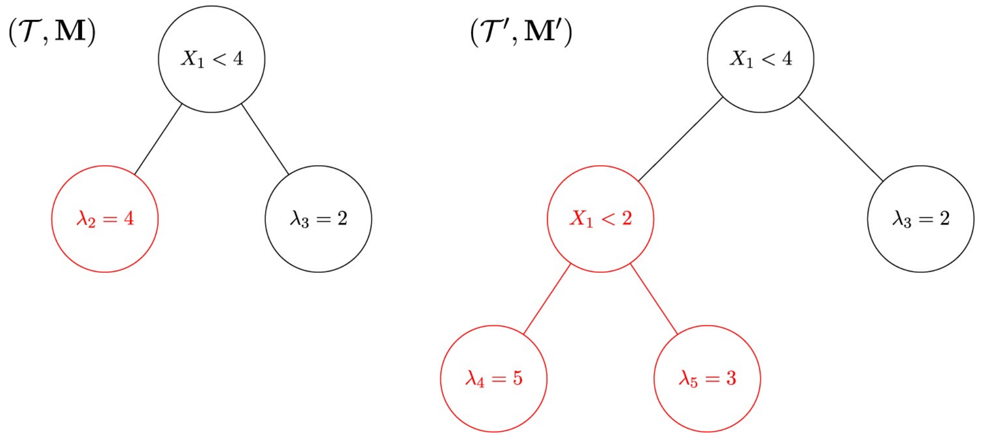


Figure 4. Left: a tree $(\mathscr{T}, \mathbf{M})$ with red-highlighted terminal node selected for birth. Right: an updated tree $(\mathscr{T}', \mathbf{M}')$ after birth, with the updated tree structure again highlighted in red. Here the node η_2 has been converted to an internal node with rule $X_1 < 2$ and assigned two child terminal nodes. Note that the choice of new rule leads to a coherent re-partitioning of the

covariate space. The reverse death move is represented by the transformation $(\mathscr{T}', \mathbf{M}') \to (\mathscr{T}, \mathbf{M})$.

3.2. Count models

For count models, typical likelihood choices include the Poisson and negative binomial distributions, with the latter often desirable due to its ability to model overdispersion. Other likelihood choices for count data models have been introduced that allow for even more flexible modelling of mean-variance relationships (i.e. underdispersion), including the COM-Poisson [20] and double Poisson [21], but these have historically suffered from computational and parameter interpretation issues that have prevented more widespread adoption [22].

In the tree literature, Murray [23] extends the BART framework to accommodate multinomial logistic and count regression models, utilizing the sum-of-trees approach to build up appropriate transformations of mean functions of interest. Key to their model is a sampling algorithm that allows for blocked MCMC updating of each tree $\mathscr{T}_h$, $h = 1, \ldots, m$, and its parameters $\mathbf{M}_h$ while holding $(\boldsymbol{\mathcal{T}}_{-h}, \mathbf{M}_{-h})$ fixed. In the case of count data, Murray achieves this by augmenting a Poisson or negative binomial likelihood with additional latent variables to allow for tractable integrated marginal likelihoods and introduce new conjugate prior distributions. The resulting closed-form marginal likelihoods are thus available to update individual trees via MH, and updating each tree's terminal node parameters may be performed using Gibbs steps after latent variable updates, yielding a non-backfitting update algorithm that still resembles BART's fast-update scheme; however, a key disadvantage of Murray's count model formulation is the number of data-dependent latent variable updates required when modelling zero-inflation and overdispersion.

3.3. The likelihood function

Our proposed likelihood function is fully specified by three parameters: a location parameter $\lambda \in \mathbb{Z}_{\geq 0}$; a scale parameter $k \in \mathbb{Z}$; and a tail mass parameter $t \in [0, 0.5)$. The tail mass parameter controls the probability assigned to each tail of the resulting distribution. The definition of this distribution follows a piece-wise construction and is written as follows:

$$p_t(Y|\lambda, k) = \begin{cases} (1-2t)\dfrac{k+1-|Y-\lambda|}{(k+1)^2}, & |Y-\lambda| \leq k, \\ tp^*(1-p^*)^{|Y-\lambda|-k-1}, & |Y-\lambda| > k, \end{cases} \tag{18}$$

where we define

$$p^* = \min\left\{0.99, \tfrac{1-2t}{t(k+1)^2}\right\}, \quad t > 0. \tag{19}$$

This distribution, which we term a *tent* pmf, is unimodal and symmetric about λ. The parameter k controls the range of Y values centred about λ in the distribution, in that the range of values contained in the middle $100(1-2t)\%$ of the distribution (the 'tent') is always $2k$. The tails follow a geometric distribution and each contains t total mass. Figure 5 visualizes this distribution for several settings of λ, k, and t.

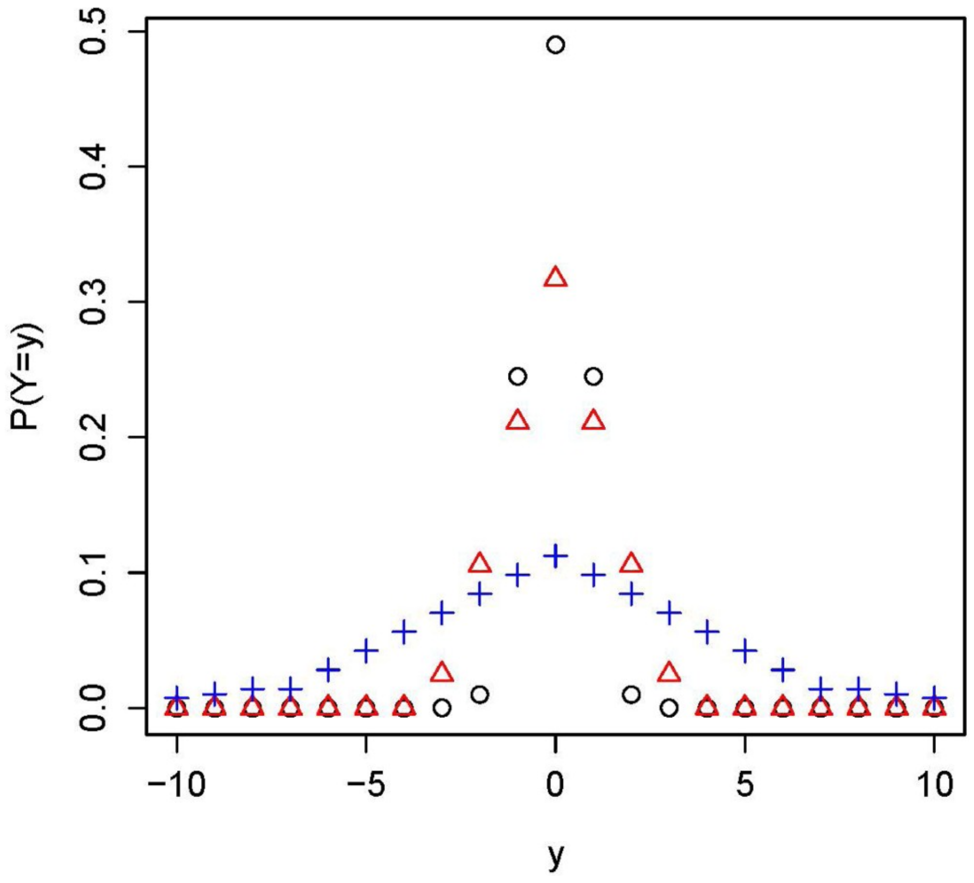


Figure 5. Several examples of the tent distribution using the tent and tail definitions given in (18) and (19). $P_{0.01}(0,1)$ is represented by the black circles, $P_{0.025}(0,2)$ is represented by the red triangles, and $P_{0.05}(0,7)$ is represented by the blue crosses.

We treat t as a hyperparameter to be fixed prior to the outset of any modelling problem. The choice of including separate scale and tail parameters in this pmf allows for additional flexibility in modelling data: the tail parameter t may be calibrated to describe relevant features of interest at the outset of the problem, whereas the behaviour of data closer to the mode can be described separately using k. Moving forward, we refer to the pmf parameterized by (λ, k) for a given t using $P_t(\lambda, k)$. Probabilities according to $Y \sim P_t(\lambda, k)$ will be referred to using $p_t(Y \mid \lambda, k)$, with the conditioning suppressed in cases where the parameterization is clear.

3.4. Prior distributions on λ and k

Since we utilize a single-tree approach for our proposed model, we maintain the implicit prior formulation on $\mathscr{T}$ as specified in CGM98, along with the prior factorization assumptions in (16) and (17). For simplicity, given the tree $\mathscr{T}$ we define a discrete uniform prior distribution on the terminal node mean parameters, i.e.

$$\lambda_b \mid \mathscr{T}, d_1, d_2 \overset{\text{ind.}}{\sim} \mathrm{DU}\{d_1, d_2\}, \quad d_1, d_2 \in \mathbb{N}_{\geq 0},\, d_1 < d_2, \tag{20}$$

in effect allowing the likelihood to guide our search algorithm toward good choices of λ_b. In application, we have taken $d_1 = \min\,(y)$ and $d_2 = \max\,(y)$ as reasonable but vague default hyperparameter values.

We propose using a similar function to the one defined in (18) and (19) as the prior distribution on each k_i. Since the tent pmf places positive mass on $\mathbb{Z}$, we adjust the likelihood function accordingly to ensure appropriate behaviour of the likelihood scale parameter; in particular, we work with the exponentiated likelihood parameterization

$$Y_i \mid \lambda_i, k_i, t \overset{\text{ind.}}{\sim} P_t(\lambda_i, \lfloor \mathrm{e}^{k_i} \rfloor), \quad i = 1, \ldots, n, \tag{21}$$

so that $\lfloor \mathrm{e}^{k_i} \rfloor \in \mathbb{Z}_{\geq 0}$ as previously required. Moving forward, we will use (21) in all instances involving the likelihood function unless otherwise noted. We specify the prior model on k_b associated to terminal node η_b as

$$k_b \mid \lambda_b, \kappa, \beta_k, t_k, \mathscr{T} \overset{\text{ind.}}{\sim} P_{t_k}\left(\left\lfloor \frac{\kappa}{2^{d(\eta_b)}} \right\rfloor, \left\lfloor \frac{\widetilde{\log}(\lambda_b)}{(1+d(\eta_b))^{\beta_k}} \right\rfloor\right), \quad \kappa \in \mathbb{Z}_{\geq 0} \tag{22}$$

where

$$\widetilde{\log}(\lambda) = \begin{cases} 0, & \lambda \leq 1, \\ \log(\lambda), & \lambda > 1. \end{cases}$$

The prior mode in (22) is a function of a hyperparameter value κ, which may be chosen to reflect some belief in the true underlying degree of dispersion if prior information is available, or otherwise chosen to provide an initial guide for the likelihood fit at the grand mean model level. The mode is also a function of node depth with respect to its terminal node η_b and codifies the belief that terminal nodes at deeper levels of a tree, reflecting more complex regions of the response surface, should accordingly seek to explain less variability in the data.

The prior scale also encodes desirable behaviour. The denominator $(1+d(\eta_b))^{\beta_k}$ utilizes the node depth penalization function in the CGM98 tree prior, where in this setting β_k controls the rate at which the prior dispersion increases or decreases; allowing $\beta_k \to \infty$ has the effect of concentrating the scale prior on its current location parameter, while letting $\beta_k \to 0$ allows the current λ_b to fully dictate the dispersion of the scale prior distribution.

From above, the full hierarchical model is specified as follows:

$$\begin{aligned} Y_i \mid t, \mathscr{T}, \lambda_i, k_i, X_i &\overset{\text{ind.}}{\sim} P_t(\lambda_i, \lfloor \exp\{k_i\} \rfloor), \\ \lambda_b \mid d_1, d_2, \mathscr{T} &\overset{\text{iid}}{\sim} \mathrm{DU}\{d_1, \ldots, d_2\}, \quad b = 1, \ldots, B, \\ k_b \mid \lambda_b, \kappa, \beta_k, t_k, \mathscr{T} &\overset{\text{ind.}}{\sim} P_{t_k}\left(\left\lfloor \frac{\kappa}{2^{d(\eta_b)}} \right\rfloor, \left\lfloor \frac{\widetilde{\log}(\lambda_b)}{(1+d(\eta_b))^{\beta_k}} \right\rfloor \right), \quad b \\ \mathscr{T} \mid \alpha, \beta, \zeta_v &\sim \pi(\mathscr{T} \mid \alpha, \beta, \zeta_v). \end{aligned} \tag{23}$$

4. Comparing the algorithms for the single-tree count data example

In this example, we generate observations from a true underlying model and attempt to recover the correct tree structure and terminal node parameters via our single-tree count model (see Section 3) using the taxicab update algorithm, and compare to a naïve MH algorithm in order to illustrate the gains in computational speed under the proposed approach. We also considered other example settings, including data exhibiting excess zeroes, for which we developed a ZI extension to our TC-augmented model. These results are available in the Supplementary Materials and were similar to what is reported here. In the present example, we considered the following two-covariate true data-generating model:

$$Y_i \mid \mathbf{X}_i \overset{\text{ind.}}{\sim} P_t(g(\mathbf{X}_i), \lfloor e^k \rfloor = \lfloor e^2 \rfloor), \tag{24}$$

where

$$g(\mathbf{X}_i) = \begin{cases} 10, & x_{i1} \leq 5, x_{i2} \leq 5, \\ 20, & x_{i1} \leq 5, x_{i2} > 5, \\ 30, & x_{i1} > 5, x_{i2} \leq 5, \\ 40, & x_{i1} > 5, x_{i2} > 5, \end{cases} \tag{25}$$

with $X_{ij} \sim \text{Unif}(0, 10)$ for $i = 1, \ldots, n$ and $j = 1, 2$. For simplicity, we took $t = 0$ in this true model so that (24) places positive probability on a finite set of values.

We proceeded to generate $n = 10, 100$, and 1000 observations according to this true model for use in our fitting procedures.

4.1. Setup

In the non-ZI model, we begin by injecting auxiliary vectors $\mathbf{U} = (U_1, \ldots, U_B) \in \mathbb{Z}^B$ and $\mathbf{R} = (R_1, \ldots, R_B) \in \mathbb{Z}^B$ into our joint distribution, so that we now work with an augmented joint distribution $p(\mathbf{Y}, \mathscr{T}, \boldsymbol{\lambda}, \boldsymbol{k}, \mathbf{U}, \mathbf{R})$. We note, in particular, that the auxiliary vectors in this model are only of dimension B, whereas in other approaches they are typically of dimension n (e.g. [1,23]). We further require the ability to factorize this joint distribution as

$$p(\mathbf{Y}, \mathscr{T}, \boldsymbol{\lambda}, \mathbf{k}, \mathbf{U}, \mathbf{R}) = p(\mathbf{Y}, \mathscr{T}, \boldsymbol{\lambda}, \mathbf{k}) p(\mathbf{U} \mid \boldsymbol{\lambda}) p(\mathbf{R} \mid \mathbf{k}), \tag{26}$$

where

$$p(\mathbf{Y}, \mathscr{T}, \boldsymbol{\lambda}, \mathbf{k}) = \left\{ \prod_{b=1}^{B} \mathscr{L}(\lambda_b, k_b \mid \mathbf{Y}, \cdot) \pi(\lambda_b \mid \mathscr{T}) \pi(k_b \mid \lambda_b, \mathscr{T}) \right\} \pi$$
$$p(\mathbf{U} \mid \boldsymbol{\lambda}) p(\mathbf{R} \mid \mathbf{k}) = \prod_{b=1}^{B} p(U_b \mid \lambda_b) p(R_b \mid k_b). \tag{27}$$

Utilizing B total blocks for updating both $\boldsymbol{\lambda}$ and $\boldsymbol{k}$, the complete TC sampler in this problem is now constructed as

$$\begin{aligned} &U_b' \sim p(U_b' \mid \lambda_b), \\ &\lambda_b' \sim q_{m_\lambda}(\lambda_b' \mid \mathbf{Y}, \mathscr{T}, k_b, U_b') \propto p(\mathbf{Y}, \mathscr{T}, \lambda_b', k_b)\mathbb{I}\{\lambda_b' \in \mathscr{N}_{m_\lambda}(U_b')\}, \end{aligned} \tag{28}$$

$$\begin{aligned} &R_b' \sim p(R_b' \mid k_b), \\ &k_b' \sim q_{m_k}(k_b' \mid \mathbf{Y}, \mathscr{T}, \lambda_b, R_b') \propto p(\mathbf{Y}, \mathscr{T}, \lambda_b', k_b')\mathbb{I}\{k_b' \in \mathscr{N}_{m_k}(R_b')\}, \end{aligned} \tag{29}$$

for $b = 1, \ldots, B$. Note that several cancellations follow in the numerator and denominator of both (28) and (29) according to (27), including $\pi(\mathscr{T})$, $\pi(\lambda_b \mid \mathscr{T})$, and all likelihood contributions from terminal nodes besides η_b; the auxiliary distribution for the fixed terminal node parameter also naturally cancels from these kernels ($p(R_b \mid k_b)$ for the update of λ_b, $p(U_b' \mid \lambda_b')$ for the update of k_b).

For the naïve MH algorithm, we utilized simple MH moves with discrete uniform proposal distributions to update (λ_b, k_b) values, along with tree birth and death moves using the marginal likelihood approach described in CGM98. In both search approaches, we utilized birth and death moves along with [7]'s cutpoint perturb proposal as an added way to facilitate exploration of the posterior tree space.

Here, the tree birth and death moves require accounting for a change of dimension in both $\boldsymbol{\lambda}$ and $\boldsymbol{k}$. Since the repertoire of tree moves we work with are local perturbations by design, we proceed to describe the construction of a dimension-changing move for tree birth proposals with respect to the local change about a terminal node η_b that has been selected for birth, with its proposed left and right child nodes denoted by $\eta_{b(l)}$ and $\eta_{b(r)}$, respectively, and the proposed rule (v', c') assigned to η_b in the new structure $\mathscr{T}'$. Further, let $\tilde{\boldsymbol{\lambda}} = (\lambda_1, \ldots, \lambda_{b-1}, \lambda_{b(l)}, \lambda_{b(r)}, \lambda_{b+1}, \ldots, \lambda_B)$, with equivalent definitions for $\tilde{\boldsymbol{k}}$, $\tilde{\mathbf{U}}$, and $\tilde{\mathbf{R}}$. In this joint setting, we also let $\mathbf{m} = (m_\lambda, m_k)$. Conveniently, the transition proposals between $\mathscr{T}$ and $\mathscr{T}'$ in our model are still handled as in CGM98, and so the immediate problem is that of the conditional transitions

$$\lambda_b \to (U_{b(l)}, U_{b(r)}), \tag{30}$$

$$(U_{b(l)}, U_{b(r)}) \to (\lambda_{b(l)}, \lambda_{b(r)}), \tag{31}$$

$$k_b \to (R_{b(l)}, R_{b(r)}), \tag{32}$$

$$(R_{b(l)}, R_{b(r)}) \to (k_{b(l)}, k_{b(l)}) \tag{33}$$

given $\mathscr{T}$ and $\mathscr{T}'$.

Per GR95, one useful way to resolve the transitions (30) and (32) is through dimension-matching. Specifically, we generate discrete random variables a_λ and a_k and subsequently define an invertible and deterministic mapping $\delta : \mathbb{Z}^2 \to \mathbb{Z}^2$ to match the dimension of our current state to that of the proposed state: for some $\theta, a \in \mathbb{Z}$ we define

$$\delta(\theta, a) = (\delta_1[\theta, a], \delta_2[\theta, a]) \tag{34}$$

$$= \left(\theta - \left\lfloor \frac{a}{2} \right\rfloor, \theta + \left\lceil \frac{a}{2} \right\rceil\right), \tag{35}$$

with inverse $\delta^{-1} : \mathbb{Z}^2 \to \mathbb{Z}^2$ given by

$$\delta^{-1}(x, y) = (\delta_1^{-1}[x, y], \delta_2^{-1}[x, y]) \tag{36}$$

$$= \left(\left\lfloor \frac{x+y}{2} \right\rfloor, y - x\right), \quad x, y \in \mathbb{Z}, \tag{37}$$

so that we obtain the desired transitions in this problem via the following sequence of operations:

1. Generate random scalars $a_\lambda \sim \mathrm{DU}\{-2m_\lambda, 2m_\lambda\}$, $a_k \sim \mathrm{DU}\{-2m_k, 2m_k\}$.
2. Given a_λ, generate the auxiliary variables $(u_{b(l)}, u_{b(r)}) = \delta(\lambda_b, a_\lambda) = (\lambda_b - \lfloor \frac{a_\lambda}{2} \rfloor, \lambda_b + \lceil \frac{a_\lambda}{2} \rceil)$.
3. Given a_k, generate the auxiliary variables $(r_{b(l)}, r_{b(r)}) = \delta(k_b, a_k) = (k_b - \lfloor \frac{a_k}{2} \rfloor, k_b + \lceil \frac{a_k}{2} \rceil)$.

Note that the choice of support for the random scalars a_λ and a_k, in tandem with the definition of the function δ, ensures that the newly proposed auxiliary variables in steps #2 and #3 of the above operation are contained respectively within the neighbourhoods centred at the current parameter values λ_b and k_b in birth proposals, as required under the TC sampler framework.

From here, the transitions (31) and (33) are proposed according to

$$(\lambda_{b(l)}, k_{b(l)}) \sim q_{\mathbf{m}}(\lambda_{b(l)}, k_{b(l)} \,|\, \mathbf{Y}, \mathscr{T}', U_{b(l)}, R_{b(l)}), \tag{38}$$

$$(\lambda_{b(r)}, k_{b(r)}) \sim q_{\mathbf{m}}(\lambda_{b(r)}, k_{b(r)} \,|\, \mathbf{Y}, \mathscr{T}', U_{b(r)}, R_{b(r)}), \tag{39}$$

where (38) and (39) follow the general definition

$$q_{\mathbf{m}}(\lambda_b, k_b \,|\, \mathbf{Y}, \mathscr{T}, U_b, R_b) \propto p(\mathbf{Y}, \mathscr{T}, \lambda_b, k_b)\mathbb{I}\{\lambda_b, k_b \in \mathscr{N}_m^2(U_b, R_b)\},$$
$$\mathbb{I}\{\lambda_b, k_b \in \mathscr{N}_m^2(U_b, R_b)\} = \mathbb{I}\{(\lambda_b, k_b) \in \mathscr{N}_{m_\lambda}(U_b) \times \mathscr{N}_{m_k}(R_b)\}.$$

The full reverse transition is then generated according to

$$U_b = \delta_1^{-1}(\lambda_{b(l)}, \lambda_{b(r)}) = \left\lfloor \frac{1}{2}(\lambda_{b(l)} + \lambda_{b(r)}) \right\rfloor, \tag{40}$$

$$R_b = \delta_1^{-1}(k_{b(l)}, k_{b(r)}) = \left\lfloor \frac{1}{2}(k_{b(l)} + k_{b(r)}) \right\rfloor, \tag{41}$$

$$(\lambda_b, k_b) \sim q_{\mathbf{m}}(\lambda_b, k_b \,|\, \mathbf{Y}, \mathscr{T}, U_b, R_b), \tag{42}$$

where (40) and (41) again allow for a deterministic mapping from the proposed higher-dimensional model to the lower-dimensional one, and (42) simply calculates the joint probability of the reverse transitions $(\lambda_{b(l)}, \lambda_{b(r)}) \to (\lambda_b, U_b)$ and $(k_{b(l)}, k_{b(r)}) \to (k_b, R_b)$.

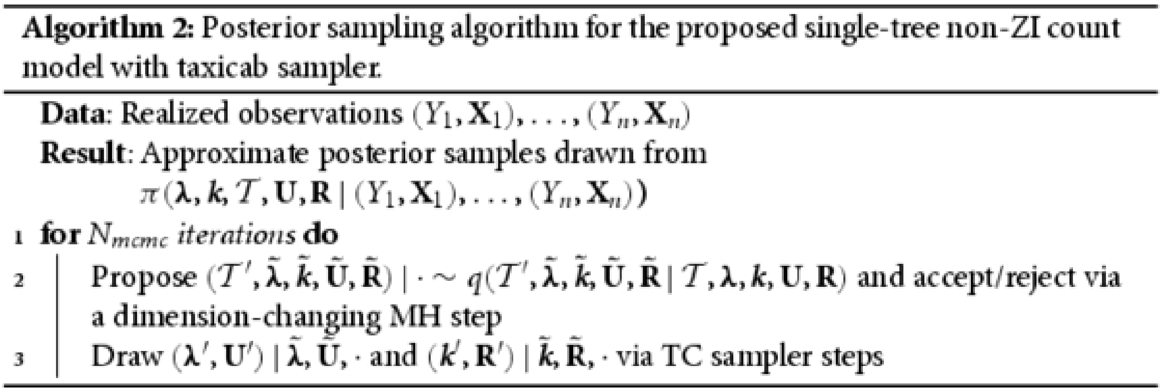
The acceptance probability for this birth move is calculated as

$$\alpha[(\mathscr{T}, \boldsymbol{\lambda}, \boldsymbol{k}, \mathbf{U}, \mathbf{R}), (\mathscr{T}', \tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{k}}, \tilde{\mathbf{U}}, \tilde{\mathbf{R}})] \tag{43}$$

$$= \min\left\{1, \frac{\pi(\mathscr{T}', \tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{k}}, \tilde{\mathbf{U}}, \tilde{\mathbf{R}} \,|\, \mathbf{Y}, \cdot) q(\mathscr{T}, \boldsymbol{\lambda}, \boldsymbol{k}, \mathbf{U}, \mathbf{R} \,|\, \mathscr{T}', \tilde{\boldsymbol{\lambda}},}{\pi(\mathscr{T}, \boldsymbol{\lambda}, \boldsymbol{k}, \mathbf{U}, \mathbf{R} \,|\, \mathbf{Y}, \cdot) q(\mathscr{T}', \tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{k}}, \tilde{\mathbf{U}}, \tilde{\mathbf{R}} \,|\, \mathscr{T}, \boldsymbol{\lambda},}\right. \tag{44}$$

In the case of a death move, the acceptance probability is calculated as the inverse of (44).

The moves described and constructed in this section give rise to the fully specified updating algorithm for our proposed single-tree non-ZI count model using the TC sampler, detailed in Algorithm 2.



Algorithm 2: Posterior sampling algorithm for the proposed single-tree non-ZI count model with taxicab sampler.

```
Data: Realized observations $(Y_1, \mathbf{X}_1), \ldots, (Y_n, \mathbf{X}_n)$
Result: Approximate posterior samples drawn from
        $\pi(\boldsymbol{\lambda}, \boldsymbol{k}, \mathcal{T}, \mathbf{U}, \mathbf{R} \mid (Y_1, \mathbf{X}_1), \ldots, (Y_n, \mathbf{X}_n))$
1 for $N_{mcmc}$ iterations do
2     Propose $(\mathcal{T}', \tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{k}}, \tilde{\mathbf{U}}, \tilde{\mathbf{R}}) \mid \cdot \sim q(\mathcal{T}', \tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{k}}, \tilde{\mathbf{U}}, \tilde{\mathbf{R}} \mid \mathcal{T}, \boldsymbol{\lambda}, \boldsymbol{k}, \mathbf{U}, \mathbf{R})$ and accept/reject via
      a dimension-changing MH step
3     Draw $(\boldsymbol{\lambda}', \mathbf{U}') \mid \tilde{\boldsymbol{\lambda}}, \tilde{\mathbf{U}}, \cdot$ and $(\boldsymbol{k}', \mathbf{R}') \mid \tilde{\boldsymbol{k}}, \tilde{\mathbf{R}}, \cdot$ via TC sampler steps
```

We followed the ‘restart’ strategy described in CGM98, running 20 individual chains for 3000 iterations each and restarting each new chain from a single-node tree. 500 burn-in iterations were used for each run and discarded prior to analysis. We utilized $\zeta = 50$ cuts to discretize each covariate dimension. Hyperparameter settings for this set of comparison simulations are detailed in Table 1.

Table 1. Hyperparameter settings for runtime comparison. (Table view)

Method	Parameters	Values considered
Naïve MH	*k* prior: (κ, β_k, t_k) combinations	(4,1,0.025)
	Tree depth prior: (α, β) combinations	(0.95,4)
	MH proposal radii: (λ, k, c) combinations	(4,2,25), (6,2,25)
	Tent pmf tail mass parameter: *t*	0.025
Taxicab	*k* prior: (κ, β_k, t_k) combinations	(4,1,0.025)
	Tree depth prior: (α, β) combinations	(0.95,4)
	$\mathscr{N}(\cdot)$ radii: (m_λ, m_k) combinations	(2,1), (3,1), (4,2), (5,2)
	MH proposal radius: *c*	25
	Tent pmf tail mass parameter: *t*	0.025

The choice of MH proposal radius for cutpoints c was selected so that perturbation proposal corresponding to an existing cut $c = 24$ or 25 could conceivably propose any other available cutpoint value in the corresponding covariate dimension. The choices of ball and MH proposal radii for each (λ_b, k_b) pair were intended to highlight any potential differences or variability in fit and computation time. For simplicity we fixed $t = 0.025$ and $t_k = 0.025$ for all runs. Assessment of fit was based on mean absolute error (MAE) and L_2 norm, both averaged over the 20 runs at each combination of hyperparameter settings.

We took 1000 posterior samples to compute both the L_2 norm and MAE quantities, along with their standard deviation (SD) and standard error (SE) respectively. Total runtime was also recorded at each combination of model settings, measuring the length of time elapsed to execute the model-fitting algorithm for all 20 runs. The results of this comparison are presented in Tables 2 and 3, corresponding to the respective outcomes for the naïve MH and TC sampler approaches.

Table 2. Comparison of runtime results for models fit with naïve MH and TC sampler approaches. (Table view)

Method	*n*	(λ, k, c) radii	Runtime (s)
Naïve MH	10	(4,2,25)	101.85
	10	(6,2,25)	101.81
	100	(4,2,25)	546.45
	100	(6,2,25)	544.52
	1000	(4,2,25)	3847.76
	1000	(6,2,25)	4072.09
Method	*n*	(m_λ, m_k, c) radii	Runtime (s)
Taxicab	10	(2,1,25)	5.11
	10	(3,1,25)	5.52
	10	(4,2,25)	7.84
	10	(5,2,25)	8.68
	100	(2,1,25)	24.67
	100	(3,1,25)	28.93
	100	(4,2,25)	41.03
	100	(5,2,25)	53.81

Method	n	(λ, k, c) radii	Runtime (s)
	1000	(2,1,25)	216.39
	1000	(3,1,25)	242.84
	1000	(4,2,25)	337.98
	1000	(5,2,25)	443.24

All reported values are rounded to the nearest hundredths place.

Table 3. Comparison of MAE and L_2 norm results for models fit with naïve MH and TC sampler approaches. (Table view)

Method	n	(λ, k, c) radii	MAE (SE)	L_2 norm (SD)
Naïve MH	10	(4,2,25)	6.12 (0.02)	189.11 (36.79)
	10	(6,2,25)	6.15 (0.02)	189.39 (41.17)
	100	(4,2,25)	2.73 (0.00)	52.30 (1.88)
	100	(6,2,25)	2.73 (0.00)	52.57 (1.97)
	1000	(4,2,25)	2.73 (0.00)	54.22 (3.10)
	1000	(6,2,25)	2.75 (0.01)	55.84 (11.88)
Method	n	(m_λ, m_k, c) radii	MAE (SE)	L_2 norm (SD)
Taxicab	10	(2,1,25)	6.23 (0.04)	195.68 (23.43)
	10	(3,1,25)	6.17 (0.03)	193.60 (21.77)
	10	(4,2,25)	6.05 (0.03)	192.16 (27.89)
	10	(5,2,25)	6.08 (0.02)	191.33 (21.80)
	100	(2,1,25)	3.98 (0.42)	84.00 (48.17)
	100	(3,1,25)	3.49 (0.26)	76.63 (42.65)
	100	(4,2,25)	2.76 (0.23)	52.84 (5.12)
	100	(5,2,25)	2.74 (0.00)	52.33 (2.81)
	1000	(2,1,25)	2.75 (0.01)	54.77 (1.82)
	1000	(3,1,25)	2.75 (0.01)	54.72 (2.03)
	1000	(4,2,25)	2.73 (0.00)	54.80 (2.47)
	1000	(5,2,25)	2.74 (0.01)	54.28 (1.25)

All reported values are rounded to the nearest hundredths place.

4.2. Performance comparison

Performance between the two methods with respect to L_2 norm was comparable across sample size, with improved recovery of the true underlying $(\mathscr{T}, \mathbf{M})$ for larger n. As expected, computation time increased in both methods with sample size and larger choice of $\mathscr{N}(\cdot)$ radii in the TC sampler approach. Depending on sample size, the TC algorithm was anywhere between 8 to 20 times faster than the naïve MH algorithm for 'similar' $\mathscr{N}(\cdot)$ and MH proposal radii settings. The runtime improvements were on the larger end of this range for $(m_\lambda, m_k) \in \{(2,1),(3,1)\}$, though the reported MAE and L_2 norm values at these settings were suboptimal compared to the $(m_\lambda, m_k) = (4,2)$ and $(5,2)$ settings at the $n = 100$ sample size. With respect to TC sampler results for the $n = 1000$ sample size, the most-probable tree configurations were frequently close to the ground truth tree structure, with some runs identifying somewhat larger trees due to the inclusion of extraneous internal node rules that were unable to be pruned away; we note that similar behaviour occurred with most probable tree configurations at this sample size setting under the naïve MH sampler with comparable frequency, indicating the 'excess' estimated tree structure is an effect of the underlying stochastic-search mechanism used to explore posterior trees in these kinds of models, as opposed to an inherent issue with the TC sampler itself.

Though the calculated MAEs were in-sample, they are included in both summary tables as a simple way to screen any potentially noticeable differences in mean parameter fits both within and between model fits according to the two algorithms. The MAEs for the $(m_\lambda, m_k) = (2,1)$ and $(m_\lambda, m_k) = (3,1)$ settings at $n = 100$ are somewhat higher than their counterparts under the naïve MH sampler, suggesting that chains involving combinations of smaller (m_λ, m_k) values require longer mixing time for intermediate sample sizes due to constraints imposed in the construction of our dimension-changing proposals (see Figure 6); however, the MAE in the $(m_\lambda, m_k) = (4,2)$ and $(5,2)$ settings at $n = 100$ is in line with the $n = 100$ results in the naïve MH sampler, indicating that, in medium sample size cases, the TC sampler is able to identify good tree structures and terminal node parameter values in shorter chains at other reasonable $\mathscr{N}(\cdot)$ radii settings. Otherwise, MAEs did not appear appreciably different between the two approaches in the reported results, showing that the TC sampler performs comparably to the naïve MH sampler in a large number of cases at a fraction of the computation time.

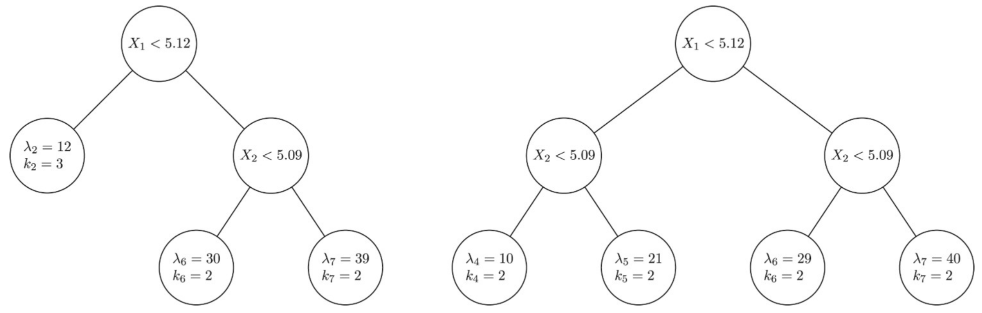

$X_1 < 5.12$
$\lambda_2 = 12$ $k_2 = 3$
$X_2 < 5.09$
$\lambda_6 = 30$ $k_6 = 2$
$\lambda_7 = 39$ $k_7 = 2$
$X_1 < 5.12$
$X_2 < 5.09$
$X_2 < 5.09$
$\lambda_4 = 10$ $k_4 = 2$
$\lambda_5 = 21$ $k_5 = 2$
$\lambda_6 = 29$ $k_6 = 2$
$\lambda_7 = 40$ $k_7 = 2$

Figure 6. Left: most-probable tree configuration identified on run #5 (with 26.13% within-run posterior probability) using TC sampler algorithm at $n = 100$ with $(m_\lambda, m_k) = (2, 1)$ and L_2 norm = 137.397; the displayed terminal node parameter values are from a single saved iteration associated with this tree structure. In this configuration the sampler has not accepted a birth proposal with internal rule involving X_2 at node η_2 to recover the optimal tree structure given the generated cutpoints. Right: most-probable tree configuration identified on run #1 (with 31.17% within-run posterior probability) using TC sampler algorithm at $n = 100$ with $(m_\lambda, m_k) = (2, 1)$ and L_2 norm = 50.00.

5. Discussion

The taxicab sampler presented here builds on the ideas presented in [11], offering a flexible, natural and useful extension to operations on non-binary discrete state spaces in Bayesian models. We have demonstrated the improved efficiency and inferential capabilities of the TC sampler relative to a MH sampler in a challenging univariate setting involving a complicated multimodal distribution, suggesting that even equipping the TC sampler with relatively small choices of radius parameter m can result in tangible performance improvements over a MH sampler with comparable choice of random walk proposal radius value.

Here, we have also shown the ability of the TC sampler to aid in performing efficient inference in a Bayesian regression tree count model setting with comparable performance to that of a naïve MH sampler implementation at a fraction of the computational cost; further gains in speed could be achieved by parallelizing TC sampler computations. Further, while exotic, our proposed single-tree count model offers a number of advantages, including interpretability of model parameters and the ability to readily model under-, equi-, and over-dispersion over different regions of covariate space, showing how the use of discrete parameter spaces in tandem with a non-conjugate, non-exponential-family-based model specification can serve as an interesting alternative compared to a more traditional modelling approach relying on continuous latent state spaces.

The TC sampler may also offer additional benefits in the context of efficiently searching over the posterior tree space in Bayesian regression tree models. Whereas [8] use a full marginalization strategy to perform advantageous tree updates in these types of models, the ability to marginalize over a subset of tree structures to sidestep traversing low-probability regions of the posterior tree space is desirable in its own right, and the local marginalization approach presented in this paper offers interesting considerations for this tree-mixing problem if a suitable distance can be identified.

Disclosure statement

The authors report there are no competing interests to declare(s).

Funding

The work of Matthew T. Pratola was supported in part by the National Science Foundation (NSF) under Agreements DMS-1564395 and DMS-1916231 and in part by the King Abdullah University of Science and Technology (KAUST) Office of Sponsored Research (OSR) under Award No. OSR-2018-CRG7-3800.3.

References

[1] Albert JH, Chib S. Bayesian analysis of binary and polychotomous response data. *J Am Stat Assoc*. 1993;88(422):669–679. Crossref. Web of Science.

[2] Ghosh SK, Mukhopadhyay P, Lu JC. Bayesian analysis of zero-inflated regression models. *J Stat Plan Inference*. 2006;136:1360–1375. Crossref. Web of Science.

[3] Neelon B. Bayesian zero-inflated negative binomial regression based on Pólya-Gamma mixtures. *Bayesian Anal*. 2019;14(3):829–855. Crossref. PubMed.

[4] Chipman HA, George EI, McCulloch RE. Bayesian CART model search. *J Am Stat Assoc*. 1998;93(443):935–948. Crossref. Web of Science.

[5] Denison DGT, Mallick BK, Smith AFM. A Bayesian CART algorithm. *Biometrika*. 1998;85(2):363–377. Crossref. Web of Science.

[6] Chipman HA, George EI, McCulloch RE. BART: Bayesian additive regression trees. *Ann Appl Stat*. 2010;4(1):266–298. Crossref. Web of Science.

[7] Pratola MT. Efficient Metropolis–Hastings proposal mechanisms for Bayesian regression tree models. *Bayesian Anal*. 2016;11(3):885–911. Crossref.

[8] Mohammadi R, Pratola M, Kaptein M. Continuous-time birth-death MCMC for Bayesian regression tree models. *J Mach Learn Res*. 2020;21(201):1–26. PubMed.

[9] O'Hagan A, Woodward EG, Moodaley LC. Practical Bayesian analysis of a simple logistic regression: predicting corneal transplants. *Stat Med*. 1990;9(9):1091–1101. Crossref. PubMed.

[10] Metz L, Ibarz J, Jaitly N, et al. Discrete sequential prediction of continuous actions for deep RL; 2019. Available from: arXiv:1705.05035v3 [cs.LG].

[11] Titsias MK, Yau C. The hamming ball sampler. *J Am Stat Assoc*. 2017;112(520):1598–1611. Crossref. PubMed. Web of Science.

[12] Zanella G. Informed proposals for local MCMC in discrete spaces. *J Am Stat Assoc*. 2020;115(530):852–865. Crossref. Web of Science.

[13] Zhou Q, Yang J, Vats D, et al. Dimension-free mixing for high-dimensional Bayesian variable selection; 2021. Available from: arXiv:2105.05719 [stat.ME].

[14] Grathwohl W, Swersky K, Hashemi M, et al. Oops I took a gradient: Scalable sampling for discrete distributions; 2021. Available from: arXiv:2102.04509 [cs.LG].

[15] Hastie DI, Green PJ. Model choice using reversible jump Markov Chain Monte Carlo. *Stat Neerl*. 2012;66(3):309–338. Crossref.

[16] Green PJ. Reversible jump Markov Chain Monte Carlo computation and Bayesian model determination. *Biometrika*. 1995;82(4):711–732. Crossref. Web of Science.

[17] Brooks S, Gelman A, Jones G, et al. *Handbook of Markov Chain Monte Carlo*. CRC Press; 2011. Crossref.

[18] Dorogush AV, Ershov V, Gulin A. CatBoost: gradient boosting with categorical features support; 2018. Available from: arXiv:1810.11363 [cs.LG].

[19] Breiman L, Friedman JH, Olshen RA, et al. *Classification and regression trees*. 2nd ed. Chapman and Hall; 1984.

[20] Conway RW, Maxwell WL. A queuing model with state dependent service rates. *J Ind Eng*. 1962;12:132–136.

[21] Efron B. Double exponential families and their use in generalized linear regression. *J Am Stat Assoc*. 1986;81(395):709–721. Crossref. Web of Science.

[22] Sellers KF, Shmueli G. A flexible regression model for count data. *Ann Appl Stat*. 2010;4(2):943–961. Crossref. Web of Science.

[23] Murray JS. Log-linear Bayesian additive regression trees for multinomial logistic and count regression models. *J Am Stat Assoc*. 2021;116(534):756–769. Crossref.

Sections

List of Illustrations

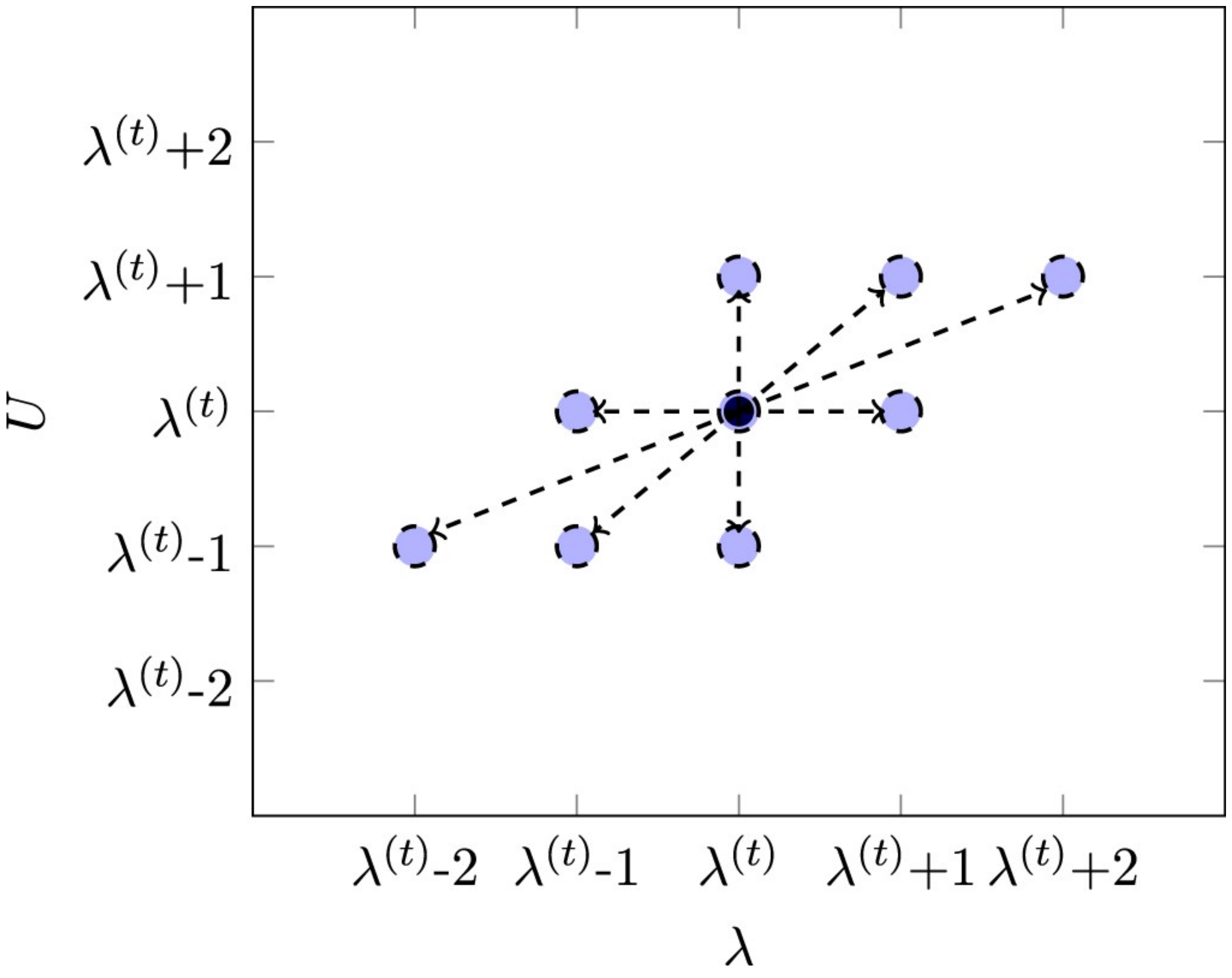


Figure 1. Representation of valid bivariate proposal states $(U^{(t+1)}, \lambda^{(t+1)})$ (dashed blue circles) from state originating at $(U^{(t)}, \lambda^{(t)}) = (\lambda^{(t)}, \lambda^{(t)})$ (solid black circle) via the bivariate kernel (10) with radius $m_\lambda = 1$ when *B* = 1. Dashed directional arrows connect the origin to each proposed state. Note that $(U^{(t+1)}, \lambda^{(t+1)}) = (\lambda^{(t)}, \lambda^{(t)})$ is a valid proposal state.

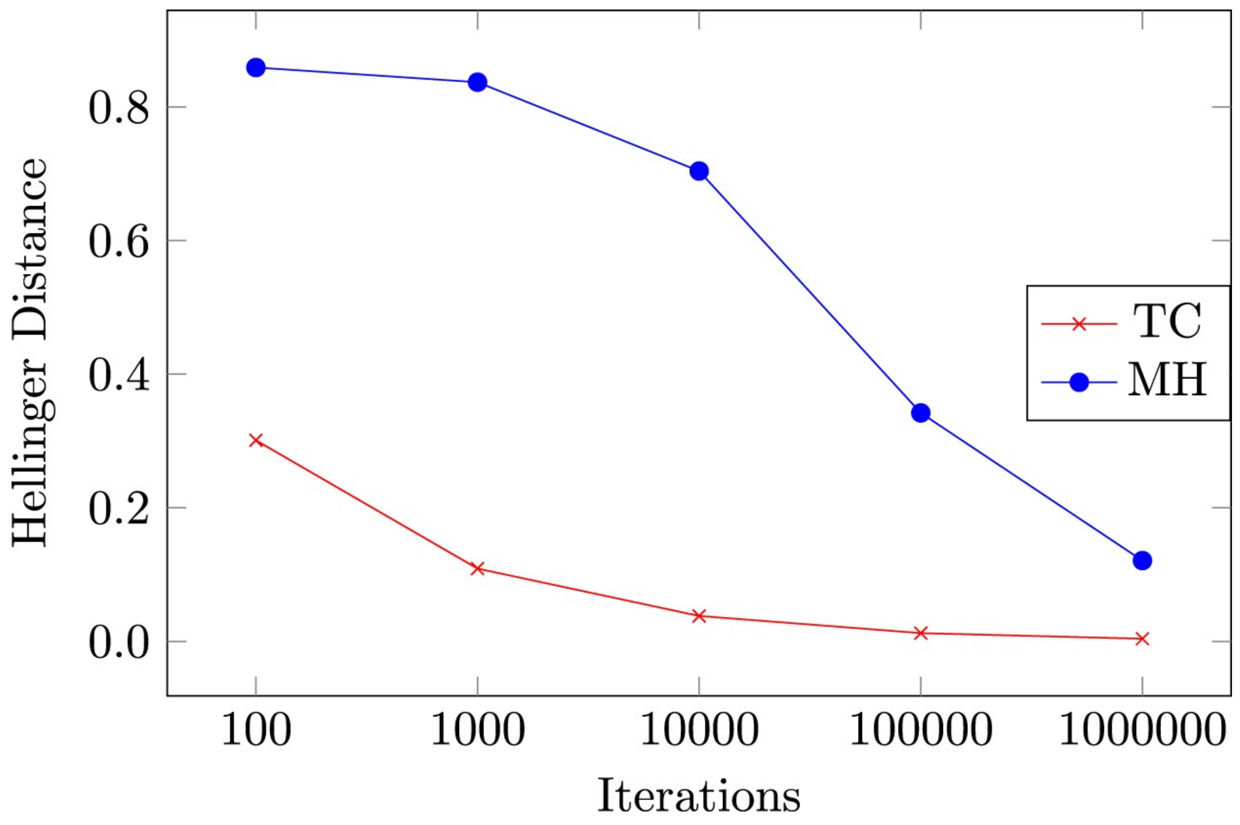


Figure 2. Mean Hellinger distance (based on 100 samples) at selected iterations between estimated and target distributions under the TC and MH samplers for the example scenario presented in Supplementary Materials Appendix B.

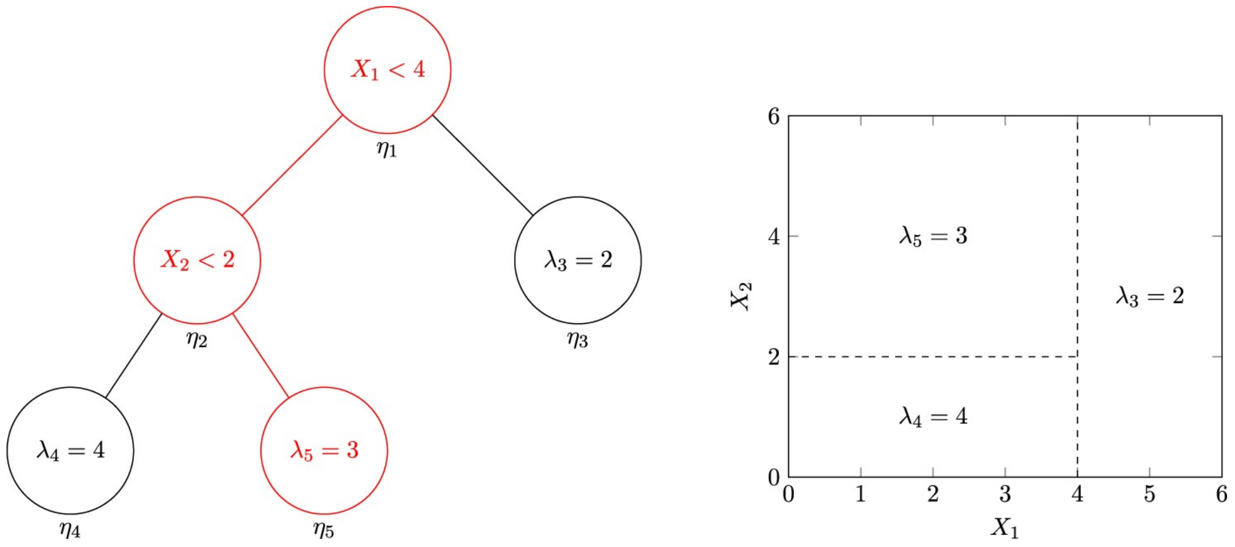


Figure 3. Left: An example tree $(\mathcal{T}, \mathcal{M})$ containing two internal nodes (η_1, η_2) with rules and three terminal nodes (η_3, η_4, η_5) with associated parameters. The example vector $(x_1, x_2) = (3, 3)$ would be sorted along the path highlighted by the red nodes and collected in η_5; the corresponding path from root node to η_5 is represented by $\mathcal{X}_5 = \{x_1 < 4 \cap x_2 \geq 2\}$. Right: an equivalent representation of $(\mathcal{T}, \mathcal{M})$ demonstrating how the tree structure partitions the covariate space (X_1, X_2) into hyperrectangles and assigns parameter values to each. Here $\mathcal{X}_5$ corresponds to the top-left hyperrectangle with parameter value $\lambda_5 = 3$.

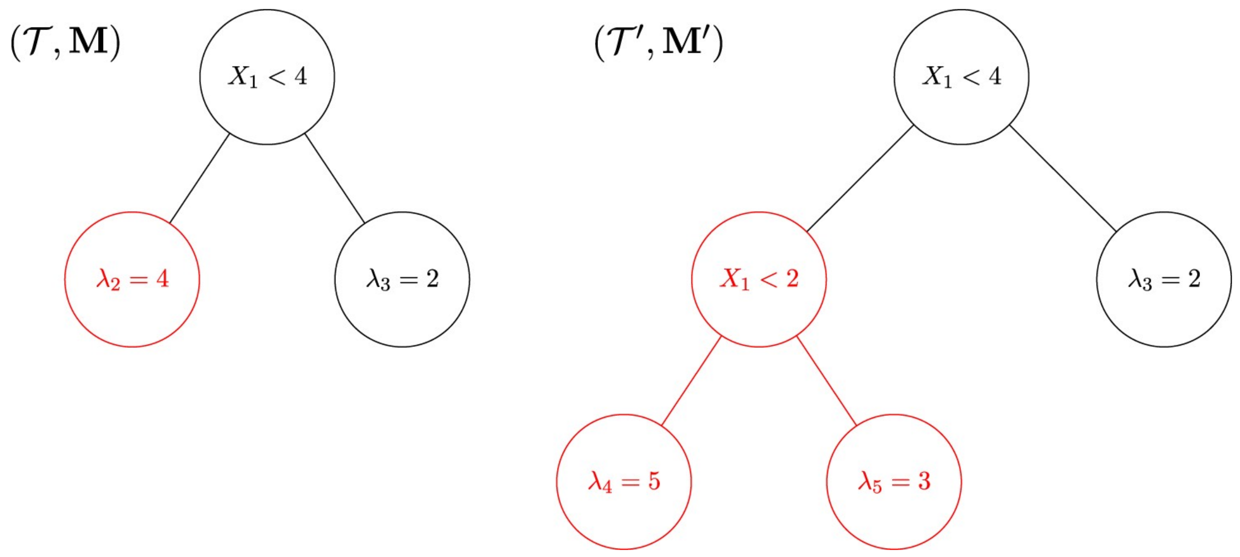


Figure 4. Left: a tree $(\mathcal{T}, \mathbf{M})$ with red-highlighted terminal node selected for birth. Right: an updated tree $(\mathcal{T}', \mathbf{M}')$ after birth, with the updated tree structure again highlighted in red. Here the node η_2 has been converted to an internal node with rule $X_1 < 2$ and assigned two child terminal nodes. Note that the choice of new rule leads to a coherent re-partitioning of the covariate space. The reverse death move is represented by the transformation $(\mathcal{T}', \mathbf{M}') \rightarrow (\mathcal{T}, \mathbf{M})$.

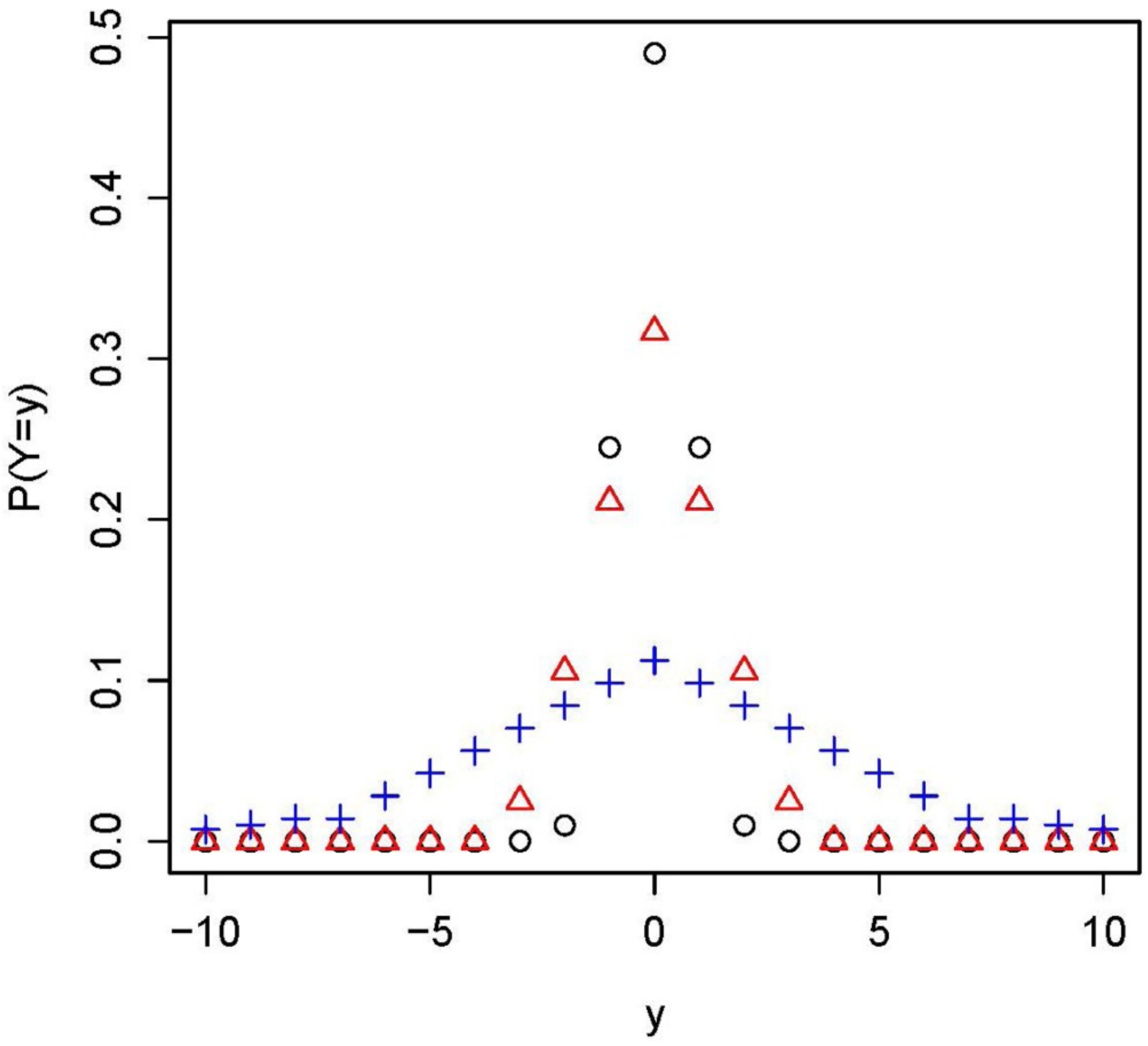


Figure 5. Several examples of the tent distribution using the tent and tail definitions given in (18) and (19). $\mathrm{P}_{0.01}(0, 1)$ is represented by the black circles, $\mathrm{P}_{0.025}(0, 2)$ is represented by the red triangles, and $\mathrm{P}_{0.05}(0, 7)$ is represented by the blue crosses.

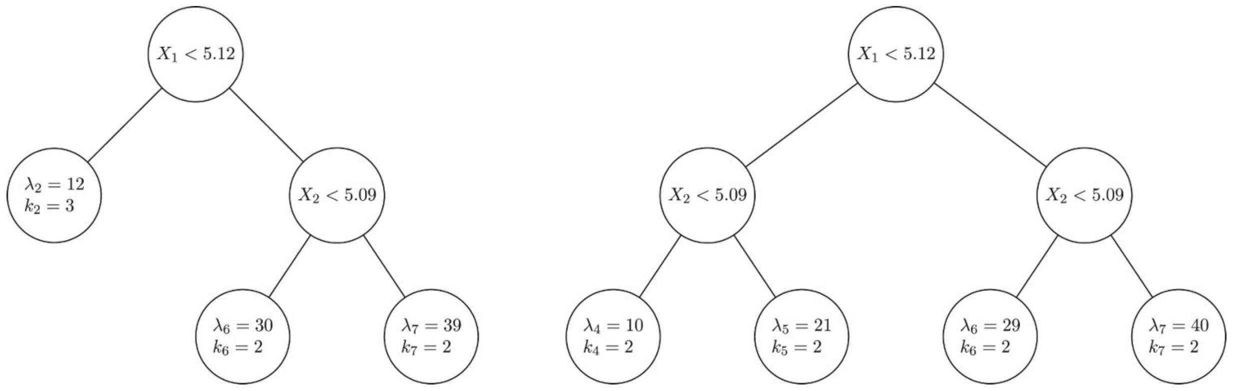

$X_1 < 5.12$
$\lambda_2 = 12$ $k_2 = 3$
$X_2 < 5.09$
$\lambda_6 = 30$ $k_6 = 2$
$\lambda_7 = 39$ $k_7 = 2$
$X_1 < 5.12$
$X_2 < 5.09$
$X_2 < 5.09$
$\lambda_4 = 10$ $k_4 = 2$
$\lambda_5 = 21$ $k_5 = 2$
$\lambda_6 = 29$ $k_6 = 2$
$\lambda_7 = 40$ $k_7 = 2$

Figure 6. Left: most-probable tree configuration identified on run #5 (with 26.13% within-run posterior probability) using TC sampler algorithm at *n* = 100 with $(m_\lambda, m_k) = (2, 1)$ and L_2 norm = 137.397; the displayed terminal node parameter values are from a single saved iteration associated with this tree structure. In this configuration the sampler has not accepted a birth proposal with internal rule involving X_2 at node η_2 to recover the optimal tree structure given the generated cutpoints. Right: most-probable tree configuration identified on run #1 (with 31.17% within-run posterior probability) using TC sampler algorithm at *n* = 100 with $(m_\lambda, m_k) = (2, 1)$ and L_2 norm = 50.00.

Table 1. Hyperparameter settings for runtime comparison.

Method	Parameters	Values considered
Naïve MH	*k* prior: (κ, β_k, t_k) combinations	(4,1,0.025)
	Tree depth prior: (α, β) combinations	(0.95,4)
	MH proposal radii: (λ, k, c) combinations	(4,2,25), (6,2,25)
	Tent pmf tail mass parameter: *t*	0.025
Taxicab	*k* prior: (κ, β_k, t_k) combinations	(4,1,0.025)
	Tree depth prior: (α, β) combinations	(0.95,4)
	$\mathcal{N}(\cdot)$ radii: (m_λ, m_k) combinations	(2,1), (3,1), (4,2), (5,2)
	MH proposal radius: *c*	25
	Tent pmf tail mass parameter: *t*	0.025

Table 2. Comparison of runtime results for models fit with naïve MH and TC sampler approaches.

Method	*n*	(λ, k, c) radii	Runtime (s)
Naïve MH	10	(4,2,25)	101.85
	10	(6,2,25)	101.81
	100	(4,2,25)	546.45
	100	(6,2,25)	544.52
	1000	(4,2,25)	3847.76
	1000	(6,2,25)	4072.09
Method	*n*	(m_λ, m_k, c) radii	Runtime (s)
Taxicab	10	(2,1,25)	5.11
	10	(3,1,25)	5.52
	10	(4,2,25)	7.84
	10	(5,2,25)	8.68
	100	(2,1,25)	24.67
	100	(3,1,25)	28.93
	100	(4,2,25)	41.03
	100	(5,2,25)	53.81
	1000	(2,1,25)	216.39
	1000	(3,1,25)	242.84
	1000	(4,2,25)	337.98
	1000	(5,2,25)	443.24

All reported values are rounded to the nearest hundredths place.

Table 3. Comparison of MAE and L_2 norm results for models fit with naïve MH and TC sampler approaches.

Method	*n*	(λ, k, c) radii	MAE (SE)	L_2 norm (SD)
Naïve MH	10	(4,2,25)	6.12 (0.02)	189.11 (36.79)
	10	(6,2,25)	6.15 (0.02)	189.39 (41.17)
	100	(4,2,25)	2.73 (0.00)	52.30 (1.88)
	100	(6,2,25)	2.73 (0.00)	52.57 (1.97)
	1000	(4,2,25)	2.73 (0.00)	54.22 (3.10)
	1000	(6,2,25)	2.75 (0.01)	55.84 (11.88)
Method	*n*	(m_λ, m_k, c) radii	MAE (SE)	L_2 norm (SD)
Taxicab	10	(2,1,25)	6.23 (0.04)	195.68 (23.43)
	10	(3,1,25)	6.17 (0.03)	193.60 (21.77)
	10	(4,2,25)	6.05 (0.03)	192.16 (27.89)
	10	(5,2,25)	6.08 (0.02)	191.33 (21.80)
	100	(2,1,25)	3.98 (0.42)	84.00 (48.17)
	100	(3,1,25)	3.49 (0.26)	76.63 (42.65)
	100	(4,2,25)	2.76 (0.23)	52.84 (5.12)
	100	(5,2,25)	2.74 (0.00)	52.33 (2.81)
	1000	(2,1,25)	2.75 (0.01)	54.77 (1.82)
	1000	(3,1,25)	2.75 (0.01)	54.72 (2.03)
	1000	(4,2,25)	2.73 (0.00)	54.80 (2.47)
	1000	(5,2,25)	2.74 (0.01)	54.28 (1.25)

All reported values are rounded to the nearest hundredths place.