SoftwareX 35 (2026) 102847

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

BayesDensity.jl: Bayesian nonparametric density estimation in Julia

Oskar Høgberg Simensen

Independent researcher, Karlsruhe, Germany

ARTICLE INFO

Keywords:
Julia
Bayesian statistics
Nonparametric density estimation
Uncertainty quantification
Markov chain Monte Carlo
Variational inference

ABSTRACT

BayesDensity.jl is a Julia package for Bayesian nonparametric estimation of smooth univariate densities. The package provides a unified interface to a range of well-established models from the statistical literature, supporting both Markov chain Monte Carlo and variational inference for most implemented procedures. Tailor-made inference algorithms facilitate computational efficiency across all supported models. The practical usage of the package is illustrated through several real-data examples, demonstrating its ability to produce smooth, interpretable density estimates with principled uncertainty quantification. The excellent computational performance of the package is demonstrated in two benchmarks.

Metadata

Code metadata

Nr.	Code metadata description	Metadata
C1	Current code version	v0.5.1
C2	Permanent link to code/repository used for this code version	https://github.com/oskarhs/BayesDensity.jl
C3	Permanent link to Reproducible Capsule	https://doi.org/10.5281/zenodo.20541134
C4	Legal Code License	MIT license
C5	Code versioning system used	git
C6	Software code languages, tools, and services used	Julia
C7	Compilation requirements, operating environments & dependencies	See Project.toml
C8	If available Link to developer documentation/manual	https://oskarhs.github.io/BayesDensity.jl/stable/
C9	Support email for questions	oskarhsimensen@gmail.com

1. Motivation and significance

Nonparametric density estimation is a fundamental problem in statistics concerned with recovering an unknown probability density from observed data without imposing restrictive parametric assumptions. The estimation of a univariate density, in particular, has attracted sustained interest throughout the development of the field. Although nonparametric density estimators are often used to visualize univariate data [1], their role extends far beyond exploratory analysis and they often serve as key components within more complex statistical procedures. Applications of nonparametric univariate density estimation beyond simple visualization are abundant in the scientific literature, including work in finance [2,3], insurance [4], medicine [5,6], and geochemistry [7].

With the growing availability of powerful computational resources, driven by advances in hardware, Bayesian approaches to density estimation have attracted considerable attention in the statistical literature [8]. Bayesian methods not only provide highly accurate point estimates of unknown densities, but also facilitate the simple construction of uncertainty measures [9]. Although Bayesian models are highly appealing from a theoretical perspective, their practical estimation often requires the implementation of sophisticated numerical methods, which appears to have limited their widespread adoption by practitioners.

BayesDensity.jl aims to bridge this gap by offering high-quality implementations of several Bayesian density models under a simple-to-use unified interface in the Julia programming language [10]. The package provides tools for statistical estimation and inference in Bayesian nonparametric models, allowing practitioners to use the implemented

Email address: oskarhsimensen@gmail.com.

https://doi.org/10.1016/j.softx.2026.102847
Received 28 April 2026; Received in revised form 7 June 2026; Accepted 25 June 2026
Available online 29 June 2026

methods for tasks such as data visualization or as building blocks in more elaborate multivariate models.

The remainder of the article is structured as follows. Section 2 describes the software architecture and provides a brief introduction to Bayesian computation. In Section 3, we illustrate the use of the package through two real-life datasets. Section 4 discusses the impact of the package on the density estimation and Julia ecosystems. We conclude the article in Section 5.

2. Software description

2.1. Software architecture

The BayesDensity.jl package provides access to many different density estimators from the Bayesian nonparametrics literature under a unified API. The package is divided into several subpackages, where most subpackages provide the functionality needed to perform statistical inference for a given model. An overview of the models currently available as part of BayesDensity.jl can be found in Table 1. Each subpackage is organized around one or two central structs, which are used to construct a model object in Julia for density estimation. For example, the spline histogram smoother of Wand and Yu [9] is available through the BayesDensityHistSmoother package and the model can be accessed by users through the `HistSmoother` struct. Although the underlying representations of these models can differ substantially, the BayesDensity interface provides a unified way to interact with them. This makes it straightforward to test and compare different models using only a few lines of code.

Every subpackage is hosted in the BayesDensity monorepo and maintains its own test suite that is run automatically through a GitHub Actions workflow. The monorepo approach ensures that end users only need to install the specific estimator(s) they want to use, potentially leading to a significant reduction in the number of downstream dependencies. The BayesDensityCore package defines a common API for all estimators implemented in the library and provides numerous convenience functions for computation of various posterior quantities of interest and data visualization.

2.2. Software functionalities

Let $x_1, x_2, \dots, x_n$ be an independent and identically distributed (iid.) sample from an unknown smooth density f. The aim of density estimation procedures is to construct an accurate estimate $\widehat{f}$ of f based on the observed sample. Nonparametric methods are particularly attractive in this context, as they impose minimal assumptions on f and can adapt to its potentially complex structure. The Bayesian framework in particular is a powerful approach to nonparametric density estimation. Here, the density f is assumed to belong to a given space $\mathcal{F}$, consisting of probability densities of various degrees of smoothness, and is modeled as a random variable through a prior probability distribution $P(f \in \cdot)$. Inference for the density is then based on the posterior distribution,

$$P(f \in \mathcal{A} \mid x_1, \dots, x_n) = \frac{\int_{\mathcal{A}} \prod_{i=1}^{n} f(x_i) \mathrm{d}P(f)}{\int \prod_{i=1}^{n} f(x_i) \mathrm{d}P(f)}, \tag{1}$$

where $\mathcal{A}$ is a measurable set with respect to a suitable σ-algebra on $\mathcal{F}$. One particularly attractive feature of Bayesian methods is that they yield a complete probability distribution over the density f. This enables the construction of both point estimates and uncertainty measures through the posterior.

2.2.1. Markov chain Monte Carlo

The posterior distribution (1) is typically not analytically tractable, and one has to resort to approximate numerical techniques to make inferences about the density f. Markov chain Monte Carlo (MCMC) is a popular class of methods for posterior inference in Bayesian models for which analytic calculation is not possible [11]. MCMC is a powerful simulation-based technique that allows one to estimate a variety of posterior functionals of interest based on approximate samples from the posterior distribution (1). Although the samples from an MCMC procedure are generally dependent, it can be shown under mild regularity conditions that the corresponding sample-based estimators converge to their theoretical counterparts as the number of drawn samples diverges to infinity (e.g. [12]).

In density estimation tasks, MCMC can be used to generate a sample of densities $f^{(1)}, f^{(2)}, \dots, f^{(M)}$ from the posterior (1), facilitating the computation of point estimates and uncertainty measures. A natural choice for point estimation of f is the posterior mean,

$$\widehat{f}(t) = \mathbb{E}\big[f(t) \mid x_1, \dots, x_n\big] \approx \frac{1}{M} \sum_{m=1}^{M} f^{(m)}(t), \qquad t \in \mathbb{R}. \tag{2}$$

To estimate the posterior uncertainty, we may construct a pointwise credible band at level $1 - \alpha \in (0, 1)$, defined as a collection of credible intervals for $f(t)$ at the desired level for each value of t. Mathematically, a pointwise credible band is constructed by finding two functions l, u satisfying

$$P\big(l(t) \le f(t) \le u(t) \mid x_1, \dots, x_n\big) = 1 - \alpha, \qquad \forall t \in \mathbb{R}. \tag{3}$$

In practice, l, u are approximated using the $\alpha/2$ and $1 - \alpha/2$ sample quantiles of $f^{(m)}$, evaluated on a fine grid.

The high- or infinite-dimensional nature of models used for Bayesian nonparametric density estimation makes it difficult to apply general-purpose MCMC samplers to simulate from the posterior. To circumvent this issue, BayesDensity provides a set of efficient tailor-made algorithms for each of the implemented models, allowing end users to perform posterior inference with relative ease. For MCMC sampling from the posterior distribution of the chosen model, BayesDensity exports the `sample` method, which returns an object of type `PosteriorSamples` that can be used directly to compute posterior quantities of interest such as the posterior mean (2), or for data visualization purposes (see Section 3.1). `PosteriorSamples` objects are not limited to estimation of the density f, and can also be used for inferences about the cumulative density $F(x) = \int_{-\infty}^{x} f(t)\mathrm{d}t$ or the quantile function $F^{\leftarrow}(p) = \inf\{x : F(x) \ge p\}$. These quantities are, for example, required in inverse copula models [13].

2.2.2. Variational inference

Variational inference (VI) is a different paradigm for performing posterior inference in Bayesian models, providing a fast, deterministic

Table 1
Overview of the models implemented by BayesDensity.jl as of version v0.5.1, along with the corresponding subpackages and the methods available for posterior inference.

Model	Subpackage	Inference method	
		MCMC	VI
`HistSmoother`	BayesDensityHistSmoother	✓	✓
`BSplineMixture`	BayesDensityBSplineMixture	✓	✓
`PitmanYorMixture`	BayesDensityPitmanYorMixture	✓	✓
`FiniteGaussianMixture`	BayesDensityFiniteGaussianMixture	✓	✓
`RandomFiniteGaussianMixture`	BayesDensityFiniteGaussianMixture	✓	✓
`RandomBernsteinPoly`	BayesDensityRandomBernsteinPoly	✓	✗

alternative to MCMC [14]. Variational approaches seek to approximate the posterior distribution (1) by a distribution Q^* from a tractable family $\mathcal{Q}$, obtained as

$$Q^* = \arg\min_{Q\in\mathcal{Q}} \mathrm{KL}\big(Q \parallel P(\cdot \mid x_1, \ldots, x_n)\big),$$

where $\mathrm{KL}(Q \parallel P)$ denotes the Kullback–Leibler divergence [15]. In practice, variational methods proceed by solving the equivalent problem of maximizing the evidence lower bound (ELBO) [16],

$$\mathrm{ELBO}(Q) = \log \int \prod_{i=1}^{n} f(x_i)\,\mathrm{d}P(f) - \mathrm{KL}\big(Q \parallel P(\cdot \mid x_1, \ldots, x_n)\big) \tag{4}$$

For the models implemented in BayesDensity, the variational family $\mathcal{Q}$ is chosen so that it is straightforward to generate iid. samples from the variational approximation Q^*, making it easy to perform Monte Carlo-based inference via estimators such as (2) for the posterior mean and (3) for pointwise credible bands. Since the parameterizations and variational families vary according to the chosen method, BayesDensity.jl provides tailored optimization routines for efficient fitting of variational approximations across all model classes. A key limitation of the VI approach to posterior inference is its approximate nature, and it may be difficult to quantify how well the variational distribution approximates the true posterior. Still, it provides an economical alternative when the use of MCMC is deemed too slow for a particular application.

BayesDensity exports the `varinf` method for carrying out variational inference, which uses iterative numerical optimization techniques to compute an approximate maximizer of (4). This method returns an object representing the variational posterior distribution Q^* which is a subtype of the abstract type `AbstractVIPosterior`. Similarly to `PosteriorSamples`, the variational posterior distribution object can be used directly for visualization or computation of key functionals of interest via Monte Carlo methods.

2.2.3. Prior elicitation

The prior distribution $P(f \in \cdot)$ associated with a given model depends on a set of hyperparameters, whose values influence the resulting posterior distribution. BayesDensity.jl allows the user to specify these quantities through the use of keyword arguments when creating model objects. The package documentation supports the user in selecting appropriate hyperparameter values by explaining the roles of each prior hyperparameter, with references to relevant scientific literature. BayesDensity.jl also provides default values for the hyperparameters based on recommendations from the aforementioned literature.

2.2.4. Diagnosing convergence

The successful practical application of Bayesian methods involves performing checks of the results of the model fitting procedure itself. As a result, a large literature has emerged on the topic of diagnosing the results of MCMC procedures; see [17] for a modern review. The assessment of convergence of Markov chains in Bayesian nonparametrics is complicated by the fact that the primary object of interest is a function and not a set of easily interpretable scalar parameters [18]. To help practitioners visually inspect the output of MCMC fits, BayesDensity.jl provides the `check_chains` method, which can be used to draw trace and autocorrelation plots of the sampled density $f^{(m)}$ evaluated on a grid of points for one or more chains. Additionally, the package provides access to several commonly used MCMC convergence diagnostics—including the Gelman–Rubin statistic and effective sample size—via an extension of the MCMCDiagnosticTools.jl package [19]. The package documentation also provides a brief, example-based introduction to the use of MCMC diagnostics aimed primarily at practitioners who may be unfamiliar with their use.

Convergence assessment also plays an important role when using VI schemes to approximate the posterior distribution in practical problems. Since the ELBO criterion (4) is generally non-convex and may exhibit multiple local maxima, standard considerations from numerical optimization apply. In particular, the choice of initialization can have a substantial impact on the resulting solution. To facilitate convergence assessment, the `varinf` method returns a `VariationalOptimizationResult` object, which records the value of the ELBO at each iteration along with the final convergence status. This information can be used to monitor the optimization process and to assess the quality of solutions obtained from optimization runs starting from different initial variational distributions.

3. Illustrative examples

3.1. Data visualization

In our first example we show how to use BayesDensity for posterior inference and data visualization. Here, we consider the Old Faithful dataset from [20], which consists of measured waiting times in minutes between the consecutive eruptions of the Old Faithful geyser in Yellowstone National Park. In Julia, this dataset is available as part of the RDatasets.jl package [21], and it can be loaded as follows:

```
using RDatasets
data = dataset("datasets", "faithful").Waiting
```

To analyze the data, we fit a Gaussian mixture model with an unknown number of components [22], using MCMC to generate samples from the posterior distribution. The code snippet shown below illustrates how to create a model object and sample from the corresponding posterior distribution:

```
using BayesDensityFiniteGaussianMixture

# Create a model object
model = RandomFiniteGaussianMixture(data)

# Generate a total of 11_000 samples and discard the first 1_000
mcmc_samples = sample(model, 11_000; n_burnin = 1_000)
```

In the above example, samples from the initial burn-in period are discarded to reduce the influence of the Markov chain's starting point on parameter estimates.

Having generated samples from the posterior distribution, we can use the resulting `mcmc_samples` object directly for inference. For example, we can compute the posterior mean and variance of $f(60)$ as follows:

```
mean(mcmc_samples, 60)
var(mcmc_samples, 60)
```

The MCMC output can also be used to plot estimates of the density. For convenience, BayesDensity provides extensions for the Makie.jl [23] and Plots.jl [24] packages that allow MCMC and VI objects to be used directly as plotting inputs. The code snippet shown below illustrates how to produce a simple plot of the posterior mean, along with a 90% pointwise credible band with Makie.jl. A version of the resulting plot is shown in Fig. 1.

```
using CairoMakie
plot(mcmc_samples; level = 0.90)
```

3.2. Mortality data

In our second example, we study how age moderates the risk of dying from heart disease in an Estonian sample from the year 2 000. The dataset is retrieved from the Human Mortality Database [25] and contains a total of 18 403 deaths grouped by age brackets, of which 7 213 were due to heart-related diseases. To explain our modeling approach, we introduce some mathematical notation. Let D and H be indicator variables denoting whether an individual has died and whether the cause of death was

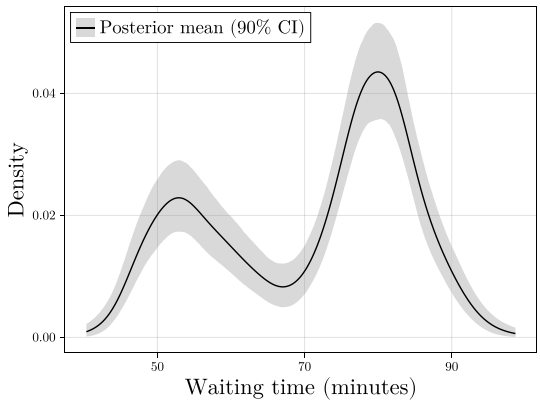

Fig. 1. Density estimate for the Old Faithful waiting times data. The thick line is the posterior mean, while the shaded band demarcates a 90% pointwise credible band.

heart disease. Through an application of Bayes' Theorem, we can write the probability of a death at a given age being due to heart disease as

$$P(H = 1 \mid \text{Age}, D = 1) = \frac{f_1(\text{Age})\, P(H = 1 \mid D = 1)}{f_1(\text{Age})\, P(H = 1 \mid D = 1) + f_0(\text{Age})\left\{1 - P(H = 1 \mid D = 1)\right\}}. \tag{5}$$

Here, f_0 and f_1 denote the densities of age at death for individuals who died of other causes and of heart disease, respectively. The marginal probability $P(H = 1 \mid D = 1)$ can be estimated by the sample proportion of deaths attributable to heart disease, while the densities f_0 and f_1 may be estimated using nonparametric methods. Since the dataset consists of binned observations, many standard methods implemented in common software are not directly applicable, as they require exact (unbinned) data. The BayesDensity.jl package addresses this issue by providing estimators such as `BSplineMixture` and `HistSmoother`, both of which can accommodate binned continuous data. Inference for the posterior probability $P(H = 1 \mid \text{Age}, D = 1)$ can then be performed by drawing M samples $f_0^{(m)}, f_1^{(m)}$ from their joint posterior distribution and evaluating the corresponding curves via (5). This yields a Monte Carlo sample of the target probability, from which point estimates and credible intervals can be constructed.

To analyze the mortality data, we fit a `BSplineMixture` model to the age-at-death data in each subgroup using MCMC. Fig. 2 shows the estimated posterior median of $P(H = 1 \mid \text{Age}, D = 1)$ as well as a 99% pointwise credible band. The resulting curve increases monotonically, indicating that deaths from heart disease become more common with increasing age. This example illustrates how BayesDensity.jl can be used to yield smooth, interpretable estimates along with uncertainty quantification even for binned data.

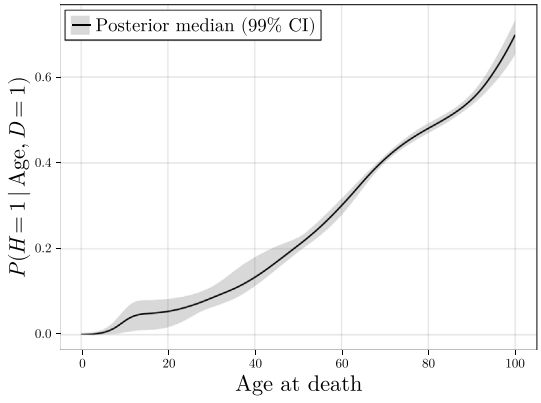

Fig. 2. Probability of a death being due to heart disease by age at death in the Estonian mortality data. The thick line is the posterior median while the shaded band delimits a 99% pointwise credible band.

3.3. Benchmarks

To showcase the computational efficiency of BayesDensity.jl we compared the performance of the package to existing software implementations of Bayesian nonparametric density estimators across three real-life datasets of different sizes. The datasets considered were the Galaxies dataset ($n = 82$), the Old Faithful waiting times considered in the previous subsection ($n = 272$), and a dataset consisting of 7 201 British incomes. We fitted each method to all three datasets and recorded the time to evaluate the resulting posterior means on a regular grid consisting of 10^4 points spanning the range of the data. For each model a total of 10 000 posterior samples were generated of which the first 1 000 were discarded. The remaining 9 000 samples were then used to estimate the posterior mean. All benchmarks were performed on a laptop equipped with an Intel Core i7-1255U (12th Gen) processor running at 1.7 GHz and 16 GB of RAM. The results are shown in Table 2. Our Julia implementation performs better than the competitors in almost all the settings considered, the one exception being the `RandomFiniteGaussianMixture` model and the Old Faithful dataset. The improvement in speed is especially noticeable for the Income dataset, where the runtime of BayesDensity is always less than a third of that of the other implementations. Overall, the benchmark presented here demonstrates the excellent computational performance of BayesDensity.jl.

In a second benchmark, we illustrate the scalability of BayesDensity.jl. To this end, we generated 101 independent synthetic datasets of sizes $n \in \{10^5, 10^6, 10^7\}$ from the first 10 benchmark densities used by Marron and Wand [26] and fit the `HistSmoother` and `BSplineMixture` models using MCMC and VI. For each synthetic dataset, we recorded the total time it took to run each posterior inference algorithm and to evaluate the posterior draws on a regular grid consisting of 10^4 points. Table 3 provides a summary of the simulation results of the second benchmark.

When running the numerical experiment, we found that the variational inference algorithm for the `HistSmoother` approach occasionally encountered numerical problems and failed to converge for around 30% of the simulated datasets as a result. This behavior was also observed in the original methodological work of Wand and Yu [9], who suggest using MCMC instead for posterior inference whenever such a failure is detected. Interestingly, convergence failures were mainly driven by poor performance for the densities numbered 3, 5 and 10 in [26], as the algorithm was able to converge in almost all other cases. We observed no such issues for the `BSplineMixture` model, as it reached convergence for all simulated datasets. For MCMC-based inference, the median inference times of both algorithms were always in the range 5-7 seconds, demonstrating that BayesDensity.jl is suitable for quick data visualization on a modern laptop, even for large-scale datasets. Variational Bayes tended to yield faster estimates than the corresponding MCMC method whenever convergence was reached, showing how VI provides an economical alternative to MCMC. The performance of VI-based inference in the `BSplineMixture` model is particularly impressive, as it typically reaches convergence in well under 1 second across all sample sizes, making it an attractive choice for applications where fast delivery of density estimates is of great importance.

Finally, we demonstrate the ability of the implemented estimators to produce highly accurate point estimates for smooth densities exhibiting a wide variety of shapes. To this end, for a subset of the Marron–Wand densities, we recorded the integrated absolute error (IAE), $\int |\hat{f}(t) - f_0(t)|\, \mathrm{d}t$, incurred when the posterior mean $\hat{f}$ is used as an estimate of the true density f_0 in the experiment described above. To illustrate the typical performance of the four methods, we then identified, for each of the

Table 2
Inference times in seconds for BayesDensity and competitor packages. The Language column refers to the programming language(s) in which the respective package is implemented. The lowest inference time for each model and dataset is highlighted in bold.

Model	Package	Language	Inference time		
			Galaxy	Old Faithful	Income
`HistSmoother`	BayesDensity	Julia	**46.1**	**39.3**	**39.6**
	densEstBayes	R/C++	111.0	129.4	128.0
`PitmanYorMixture`	BayesDensity	Julia	**9.2**	**8.4**	**33.0**
	BNPmix	R/C++	29.5	32.7	4042.5
`RandomFiniteGaussianMixture`	BayesDensity	Julia	**36.9**	46.0	**80.5**
	telescope	R	51.5	**37.0**	470.3

Table 3
Median inference times in seconds for each model and mode of inference, with standard deviations in parentheses. Asterisks denote that the results are conditional on convergence.

Model	Mode of inference	Median inference time (standard deviation)		
		$n = 10^5$	$n = 10^6$	$n = 10^7$
`HistSmoother`	MCMC	5.42 (0.92)	5.04 (0.76)	6.61 (0.95)
	VI	0.88 (0.08)*	1.01 (0.09)*	3.1 (0.39)*
`BSplineMixture`	MCMC	5.18 (0.26)	5.19 (0.16)	5.46 (0.37)
	VI	0.52 (0.11)	0.58 (0.1)	0.7 (0.12)

four method–density combinations shown, the sample of size $n = 10^5$ that attained the median IAE across the replications, and plotted the resulting estimate against the true density in Fig. 3. As indicated by the large degree of overlap between the two curves in each panel, the posterior means and the true densities are in every case in close agreement, showing that the estimators are able to recover markedly different density shapes accurately even from a single representative sample.

Additional comments on both benchmarks can be found in Appendix A.

4. Impact

Bayesian nonparametrics has proven to be an effective approach to density estimation, but the difficult practical implementation of this class of methods seems to have hindered their use by practitioners somewhat. BayesDensity.jl aims to simplify the practical application of Bayesian nonparametric procedures by providing an easy-to-use interface to a wide array of methods, significantly strengthening the density estimation ecosystem in Julia as a result. Given the important role of visualization in exploratory data analysis and in diagnosing output from statistical models, the methods provided by this package are immediately applicable in a number of settings. In addition, the package has been designed with extensibility in mind. Users looking to implement new Bayesian density estimators can rely on the existing functionality for Monte Carlo estimation and plotting in the BayesDensityCore subpackage, significantly reducing the amount of boilerplate code needed to provide a convenient user interface to the new method.

To date, the broader Julia ecosystem for density estimation has been more focused on classical frequentist approaches such as histograms, kernel estimators and finite Gaussian mixtures (e.g. [27–29]), with Bayesian alternatives remaining scarce. Although Julia benefits from a rich ecosystem of general-purpose Bayesian inference tools such as Turing.jl, RxInfer.jl, Gen.jl and BAT.jl [30–33], these are not well-suited to the specific demands of nonparametric modeling, where efficient posterior inference typically requires tailor-made sampling or variational algorithms rather than generic ones. Currently, there are a few Julia packages available for Bayesian nonparametric density estimation [34–36]; see Table 4 for a summary of the functionality provided by these software libraries. Unfortunately, none of the listed packages appear to have been actively maintained in recent years. Moreover, each package is limited to a specific type of exchangeable mixture model, and each package implements only a single mode of posterior inference. In contrast, BayesDensity.jl is more flexible both in the range of supported models and in the available inference methods: it provides implementations of both MCMC and VI for most models, is not restricted to exchangeable mixtures, and includes several methods for which no public Julia implementation was previously available. The availability of alternative methods is a major advantage when faced with larger datasets, as inference in Bayesian nonparametric mixtures is known to scale quite poorly with increasing sample sizes [37]. In particular, the `HistSmoother` and `BSplineMixture` models both provide a binned-data strategy for scalable posterior inference. As we illustrated in Section 3.3, these methods facilitate fully Bayesian density estimation for datasets

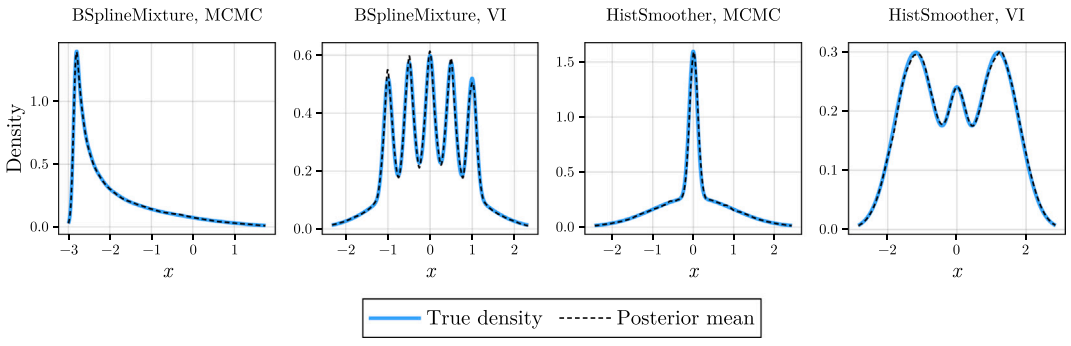

Fig. 3. Posterior-mean estimates (black, dashed) and true densities (blue, solid) for four representative samples of size $n = 10^5$. Each panel shows the results of applying one method to a different Marron–Wand density. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 4
Overview of the Julia ecosystem for Bayesian nonparametric density estimation. Latest commit dates were verified as of June 29, 2026.

Package	Supported models	Inference mode	Latest commit
BayesianMixtures.jl	Mixture of finite mixtures	MCMC	November 2020
BayesianNonparametrics.jl	Dirichlet process mixtures	MCMC	September 2018
DirichletProcessMixtures.jl	Dirichlet process mixtures	VI	July 2015

with millions of observations in just a few seconds on a modern laptop.

4.1. Limitations

As we have demonstrated, BayesDensity.jl offers several improvements over existing packages for Bayesian density estimation; however, it also has certain limitations. A popular use case for Bayesian nonparametric mixture models is semiparametric clustering, where the goal is to group individual observations into one out of several clusters. Although both the `PitmanYorMixture` and `RandomFiniteGaussianMixture` models have been used for this purpose in the statistical literature (e.g. [38, 39]), their internal representations in BayesDensity.jl prioritize memory-efficiency, which precludes their use in this context. Another limitation of the software is that many of the implemented methods may not be directly applicable to data that have been recorded in discrete units rather than continuous ones. This setting is rather common with real-life datasets, where continuous covariates such as age are often reported in whole-year intervals. Most of the implemented methods do not provide any built-in means of handling excessively binned data, and their direct application to such datasets often results in rather uninformative visual summaries. An ad-hoc solution to this problem is to add a small amount of random noise to the observed data points before running the estimation procedures.

5. Conclusion

We have presented BayesDensity.jl, a Julia package for Bayesian nonparametric univariate density estimation. The package is designed to make sophisticated Bayesian methodology accessible to practitioners who may lack the expertise or resources to implement these methods from scratch, and we hope it will encourage broader adoption of Bayesian approaches to density estimation in applied work.

Finally, we present a few possible directions for future work. Currently, BayesDensity.jl implements a single MCMC sampler for each method, yet several alternative samplers exist for some of the models in the package; providing these as options would give users greater flexibility in performing posterior inference. Given the modular design of the BayesDensity.jl package, another natural avenue for future contributions to the software is the implementation of other approaches to Bayesian density estimation such as logistic Gaussian processes [40], Pólya trees [41] and mixture models based on normalized random measures [42]. Another limitation of the package at present is the lack of a convenient user interface for parallel sampling of Markov chains. Future versions of BayesDensity.jl should look to implement the AbstractMCMC.jl [43] interface for parallel sampling. This would make it straightforward to run multiple chains concurrently through multithreading and multiprocessing, resulting in faster posterior inference, particularly for larger problems, and supporting convergence assessment via between-chain diagnostics. The resulting workflow would also be familiar to users of the Turing.jl ecosystem.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors thank the two anonymous referees for their valuable suggestions which helped improve the quality of the software and this manuscript.

Appendix A. Results from benchmarks

This section contains some additional comments on the two benchmarks of Section 3.3.

For our first benchmark, we note that there are multiple MCMC procedures that can be used to sample from the posterior distribution of the `PitmanYorMixture` and `RandomFiniteGaussianMixture` models, see e.g. [44] for a review pertaining to the former and [45] for the latter. As a result, we have restricted our comparison to libraries that implement the same algorithms as BayesDensity.jl for these two procedures, i.e., algorithm 2 of Neal [44] for `PitmanYorMixture` and the telescope sampler [45] for `RandomFiniteGaussianMixture`. The packages used in the comparison are densEstBayes [46], telescope [47] and BNPmix [48]. The Galaxy dataset is available as part of the datasets R package [49], while the British income dataset can be accessed through the densEstBayes library [46]. In all our experiments, we also ensured that the values of all model-specific hyperparameters were kept fixed when performing the comparison, as the values of these may in some cases influence the performance of the sampler. In particular, the hyperparameter values used in the experiment were taken from [46] for the `HistSmoother` model, from [22] for `RandomFiniteGaussianMixture`, and from the defaults of the BNPmix package [48] for `PitmanYorMixture`.

For the second benchmark, we ran both models with the parameter values set to the BayesDensity.jl defaults. For MCMC-based inference we generated a total of 1 100 samples and discarded the first 100 for the `HistSmoother` method, and for the `BSplineMixture` model 4 500 samples were generated with the first 500 discarded. The variational posterior means of both models were calculated using a total of 1 000 Monte Carlo samples from their respective variational posterior distributions.

Data availability

No new data were created or collected in this study. All data analyzed are publicly available from the sources cited in the manuscript.

References

[1] Scott DW. Multivariate density estimation: theory, practice, and visualization. Wiley; 1992. https://doi.org/10.1002/9780470316849
[2] Leitao A, Ortiz-Gracia L. Model-free computation of risk contributions in credit portfolios. Appl Math Comput 2020;382. https://doi.org/10.1016/j.amc.2020.125351
[3] Cui Z, Kirkby JL, Nguyen D. A data-driven framework for consistent financial valuation and risk measurement. Eur J Oper Res 2021;289:381–98. https://doi.org/10.1016/j.ejor.2020.07.011
[4] Bolancé C, Bahraoui Z, Artís M. Quantifying the risk using copulae with nonparametric marginals. Insur Math Econ 2014;58:46–56. https://doi.org/10.1016/j.insmatheco.2014.06.008
[5] Guan Z, Wu B, Zhao H. Nonparametric estimator of false discovery rate based on Bernstein polynomials. Stat Sin 2008;18:905–23.
[6] Simensen OH, Christensen D, Hjort NL. Random irregular histograms. Comput Stat Data Anal 2026;220. https://doi.org/10.1016/j.csda.2026.108367
[7] Colombo I, Nobile F, Porta G, Scotti A, Tamellini L. Uncertainty quantification of geochemical and mechanical compaction in layered sedimentary basins. Comput Methods Appl Mech Eng 2018;328:122–46. https://doi.org/10.1016/j.cma.2017.08.049
[8] Choudhuri N, Ghosal S, Roy A. Bayesian methods for function estimation. In: Bayesian thinking, Vol. 25 handbook of statistics. Elsevier; 2005. p. 373–414. https://doi.org/10.1016/S0169-7161(05)25013-7
[9] Wand MP, Yu JCF. Density estimation via Bayesian inference engines. Asta Adv Stat Anal 2022;106:199–216. https://doi.org/10.1007/s10182-021-00422-8
[10] Bezanson J, Edelman A, Karpinski S, Shah VB. Julia: a fresh approach to numerical computing. SIAM Rev 2017;59:65–98. https://doi.org/10.1137/141000671
[11] Gelfand AE, Smith AFM. Sampling-based approaches to calculating marginal densities. J Am Stat Assoc 1990;85:398–409. https://doi.org/10.1080/01621459.1990.10476213
[12] Robert CP, Casella G. Monte Carlo statistical methods. vol. 2. Springer; 2004. https://doi.org/10.1007/978-1-4757-4145-2
[13] Smith MS. Implicit copulas: an overview. Econom Stat 2023;28:81–104. https://doi.org/10.1016/j.ecosta.2021.12.002

[14] Ormerod JT, Wand MP. Explaining variational approximations. Am Stat 2010;64:140–53. https://doi.org/10.1198/tast.2010.09058
[15] Kullback S, Leibler RA. On information and sufficiency. Ann Math Stat 1951;22:79–86. https://doi.org/10.1214/aoms/1177729694
[16] Blei DM, Kucukelbir A, McAuliffe JD. Variational inference: a review for statisticians. J Am Stat Assoc 2017;112:859–77. https://doi.org/10.1080/01621459.2017.1285773
[17] Roy V. Convergence diagnostics for Markov chain Monte Carlo. Annu Rev Stat Appl 2020;7:387–412. https://doi.org/10.1146/annurev-statistics-031219-041300
[18] Pham TH, Wand MP. Generalised additive mixed models analysis via gammSlice. Aust N Z J Stat 2018;60:279–300. https://doi.org/10.1111/anzs.12241
[19] Müller-Widmann D, Pfiffer C, Axen S, contributors. MCMCDiagnosticTools.jl; 2026. https://github.com/TuringLang/MCMCDiagnosticTools.jl., Software version v0.3.17.
[20] Härdle W. Smoothing techniques: with implementation in S. Springer New York, NY; 1991. https://doi.org/10.1007/978-1-4612-4432-5
[21] White JM, Zwitch R, Bates D, contributors. RDatasets.jl; 2026. https://github.com/JuliaStats/RDatasets.jl., Software version v0.8.1.
[22] Richardson S, Green PJ. On Bayesian analysis of mixtures with an unknown number of components. J R Stat Soc Ser B Methodol 1997;59:731–92. https://doi.org/10.1111/1467-9868.00095
[23] Danisch S, Krumbiegel J. Makie.jl: flexible high-performance data visualization for Julia. J Open Source Softw 2021;6:3349. https://doi.org/10.21105/joss.03349
[24] Christ S, Schwabeneder D, Rackauckas C, Borregaard MK, Breloff T. Plots.jl - a user extendable plotting API for the Julia programming language. J Open Res Softw 2023. https://doi.org/10.5334/jors.431
[25] Human Mortality Database. Human mortality database, Available at:, 2026). https://doi.org/10.4054/HMD.Countries.20260218, https://www.mortality.org [Downloaded 23 March 2026].
[26] Marron JS, Wand MP. Exact mean integrated squared error. Ann Stat 1992;20:712–36. https://doi.org/10.1214/aos/1176348653
[27] Simensen OH. AutoHist.jl: a Julia package for fast and automatic histogram construction. J Open Source Softw 2025;10. https://doi.org/10.21105/joss.08850
[28] Byrne S, Noack A, Lee A, contributors. KernelDensity.jl; 2026. https://github.com/JuliaStats/KernelDensity.jl., Software version v0.6.11.
[29] van der Velde D, McCurry M, Saba E, contributors. GaussianMixtures.jl; 2026. https://github.com/davidavdav/GaussianMixtures.jl., Software version v0.3.14.
[30] Fjelde TE, Xu K, Widmann D, Tarek M, Pfiffer C, Trapp M, Axen SD, Sun X, Hauru M, Yong P, Tebbutt W, Ghahramani Z, Ge H. Turing.jl: a general-purpose probabilistic programming language. ACM Trans Probab Mach Learn 2025;1. https://doi.org/10.1145/3711897
[31] Bagaev D, Podusenko A, de Vries B. RxInfer: a Julia package for reactive real-time Bayesian inference. J Open Source Softw 2023;8. https://doi.org/10.21105/joss.05161
[32] Cusumano-Towner MF, Saad FA, Lew AK, Mansinghka VK. Gen: a general-purpose probabilistic programming system with programmable inference. In: Proceedings of the 40th ACM SIGPLAN conference on programming language design and implementation, PLDI 2019. ACM; 2019. p. 221–36. https://doi.org/10.1145/3314221.3314642
[33] Schulz O, Beaujean F, Caldwell A, Grunwald C, Hafych V, Kröninger K, Cagnina SL, Röhrig L, Shtembari L. BAT.jl: a Julia-based tool for Bayesian inference. SN Comput Sci 2021;2. https://doi.org/10.1007/s42979-021-00626-4
[34] Miller JW. BayesianMixtures.jl; 2020. https://github.com/jwmi/BayesianMixtures.jl., Software version v0.1.1.
[35] Trapp M, Saba E, contributors. BayesianNonparametrics.jl; 2016. https://github.com/OFAI/BayesianNonparametrics.jl., Software version v0.1.0.
[36] Bartunov S, contributors. DirichletProcessMixtures.jl; 2015. https://github.com/sbos/DirichletProcessMixtures.jl.
[37] Ullah I, Mengersen K. Bayesian mixture models and their big data implementations with application to invasive species presence-only data. J Big Data 2019;6. https://doi.org/10.1186/s40537-019-0188-1
[38] Escobar MD, West M. Bayesian density estimation and inference using mixtures. J Am Stat Assoc 1995;90:577–88. https://doi.org/10.1080/01621459.1995.10476550
[39] Grün B, Malsiner-Walli G, Frühwirth-Schnatter S. How many data clusters are in the Galaxy data set? Bayesian cluster analysis in action. Adv Data Anal Classif 2022;16:325–49. https://doi.org/10.1007/s11634-021-00461-8
[40] Riihimäki J, Vehtari A. Laplace approximation for logistic Gaussian process density estimation and regression. Bayesian Anal 2014;9. https://doi.org/10.1214/14-BA872
[41] Lavine M. Some aspects of Pólya tree distributions for statistical modelling. Ann Stat 1992;20:1222–35. https://doi.org/10.1214/aos/1176348767
[42] Griffin JE, Kolossiatis M, Steel MFJ. Comparing distributions by using dependent normalized random-measure mixtures. J R Stat Soc B Stat Methodol 2013;75:499–529. https://doi.org/10.1111/rssb.12002
[43] Müller-Widmann D, Pfiffer C, Fjelde TE, contributors. AbstractMCMC.jl; 2026. https://github.com/TuringLang/AbstractMCMC.jl., Software version v5.15.1.
[44] Neal RM. Markov chain sampling methods for Dirichlet process mixture models. J Comput Graph Stat 2000;9:249–65. https://doi.org/10.1080/10618600.2000.10474879
[45] Frühwirth-Schnatter S, Malsiner-Walli G, Grün B. Generalized mixtures of finite mixtures and telescoping sampling. Bayesian Anal 2021;16:1279–307. https://doi.org/10.1214/21-BA1294
[46] Wand MP. densEstBayes: density estimation via Bayesian inference engines, R package version 1.0-2.2, 2023. https://doi.org/10.32614/CRAN.package.densEstBayes
[47] Malsiner-Walli G, Grün B, Frühwirth-Schnatter S. telescope: Bayesian mixtures with an unknown number of components, R package version 0.2-1, 2025. https://doi.org/10.32614/CRAN.package.telescope
[48] Corradin R, Canale A, Nipoti B. BNPmix: an R package for Bayesian nonparametric modeling via Pitman–Yor mixtures. J Stat Softw 2021;100:1–33. https://doi.org/10.18637/jss.v100.i15
[49] R Core Team. R: a language and environment for statistical computing; 2025. https://www.R-project.org/.