ggdensity: Improved Bivariate Density Visualization in R

by James Otto and David Kahle

Abstract The ggdensity R package extends the functionality of ggplot2 by providing more interpretable visualizations of bivariate density estimates using highest density regions (HDRs). The visualizations are created via drop-in replacements for the standard ggplot2 functions used for this purpose: geom_hdr() for geom_density_2d_filled() and geom_hdr_lines() for geom_density_2d(). These new geoms improve on those of ggplot2 by communicating the probabilities associated with the displayed regions. Various statistically rigorous estimators are available, as well as convenience functions geom_hdr_fun() and geom_hdr_fun_lines() for plotting HDRs of user-specified probability density functions. Associated geoms for rug plots and pointdensity scatterplots are also presented.

1 Introduction

Density estimation is foundational to modern statistics. Not only does it provide a theoretical basis for maximum likelihood estimation (Scott, 1992), it is also an important tool in exploratory data analysis. This is especially true for univariate data: histograms, frequency polygons, and kernel density estimates (KDEs) all visualize an estimated density.

With bivariate data, the situation is more complicated as the estimated density is a 3D surface, and there is a tendency to avoid 3D visualization in static graphics due to visual perception biases. A more common strategy is to represent the surface using other geometric objects or aesthetics in a 2D plot, most commonly via contours of the density's level sets. Typically the density is estimated with a KDE and the contours correspond to the level sets of an equally spaced mesh over $(0, M]$, where M is the maximum of the estimated density's height (usually rounded to the closest "pretty" value). We will refer to these contours as ordinate mesh density contours (OMDCs), and the corresponding graphics as traditional density contour plots or OMDC plots. By ordinate, we mean the variable $z = f(x, y)$, or in general the last element of the graph of a function $f(\mathbf{x})$, which in this context represents density.

For example, the **MASS** package documentation suggests using `MASS::kde2d()` with `graphics::contour()`, which selects its level sets by calling `base::pretty()` on 10 such breaks over the $(0, M]$ ordinate range, and **ggplot2**'s `geom_density_2d()` and `geom_density_2d_filled()` do the same (Venables and Ripley, 2002; Wickham, 2009; Wilkinson, 2005). Unfortunately, the resulting regions—those bounded by the OMDCs—cannot be immediately identified with corresponding probabilities, and are challenging to interpret in the best of cases.

Following Hyndman (1996), we propose the use of highest density regions (HDRs) as replacements for density visualization based on OMDCs. In a sense made rigorous in the next section, an HDR is the smallest region containing a certain percentage of the estimated distribution, e.g. 90%. An HDR contour is the boundary of this region. HDRs are constructed by determining "good" cutoff values for the density, whereby cutoff values of the density we mean the ordinate values corresponding to the HDR contours (i.e. the HDR contours are the level sets of these "good" cutoff values). Unlike OMDCs, HDR contours' ordinate values are almost never equally spaced in the ordinate range, and computing them presents a number of practical and technical challenges.

In this article we introduce **ggdensity**, a new R package intended to address these challenges in facilitating the visualization of bivariate HDRs and related topics in the **ggplot2** framework. **ggdensity** extends **ggplot2** with a tight integration: instead of wrapping **ggplot2** calls to return `ggplot` objects that are hard to modify, **ggdensity** uses **ggplot2**'s API to provide new extensible (geom, stat) pairs that behave in the way **ggplot2** users have come to expect. These new stats provide a range of density estimation options, as we describe in the next sections.

2 Motivating example

We begin with a motivating example to show how traditional density contour plots can be misleading when exploring bivariate distributions. The top left plot in Figure 1 is a scatterplot of simulated bivariate standard normal data whose distribution we want to visualize. On the bottom left, we present the traditional way of visualizing the data's 2d distribution: a contour plot of slices of its estimated density (OMDCs). The function that created this graphic, `geom_density2d()` (alternatively `geom_density_2d()`), has been in **ggplot2** since its inception. It was modeled after a similar graphic made with base graphics using `MASS::kde2d()` with `contour()`. In the top right is the filled contour

version of the same plot, made with `geom_density_2d_filled()`. This type of plot was introduced in 2020 with **ggplot2** version 3.3.2, which leveraged **ggplot2**'s new dependency on **isoband** that came in **ggplot2** version 3.3.0 (Wilke and Pedersen, 2021).

In the bottom right is our proposed alternative, `ggdensity::geom_hdr()`. Each of the three contour plots show contours from the same estimated density surface, but the contours plotted by `geom_hdr()` are HDR contours and are chosen to be inferentially relevant. By default these are the smallest regions containing 50%, 80%, 95%, and 99% of the estimated density.

Plotting the HDRs results in a significantly more interpretable graphic that conveys more information than equally spaced density contours. To make a more direct comparison, in Figure 2 we superimpose the HDR contours onto the filled traditional density contour plot in the top right of Figure 1. The result reveals that nearly 20% of the estimated distribution is outside the lowest OMDC. Consequently, we would expect almost 1 out of every 5 observations to fall outside the traditional density contour plot.

This is somewhat of a cautionary tale: while the contours seen in the bottom left and top right plots of Figure 1 do seem to communicate some information to the viewer, it's hard to say exactly what that information is. And worse: it seems surprisingly easy to draw wrong conclusions. Upon scrutiny, the overall OMDC strategy seems suspect as a general purpose tool for visualizing where the probability mass of a bivariate distribution resides as it focuses exclusively on the density and ignores the region over which that density extends. On the other hand, with HDRs one immediately understands where the majority of the observed data lie and roughly how much data lies in each region. It is not possible to achieve these insights with traditional density contour plots, since equivalent interpretations require double integrals of the estimated density.

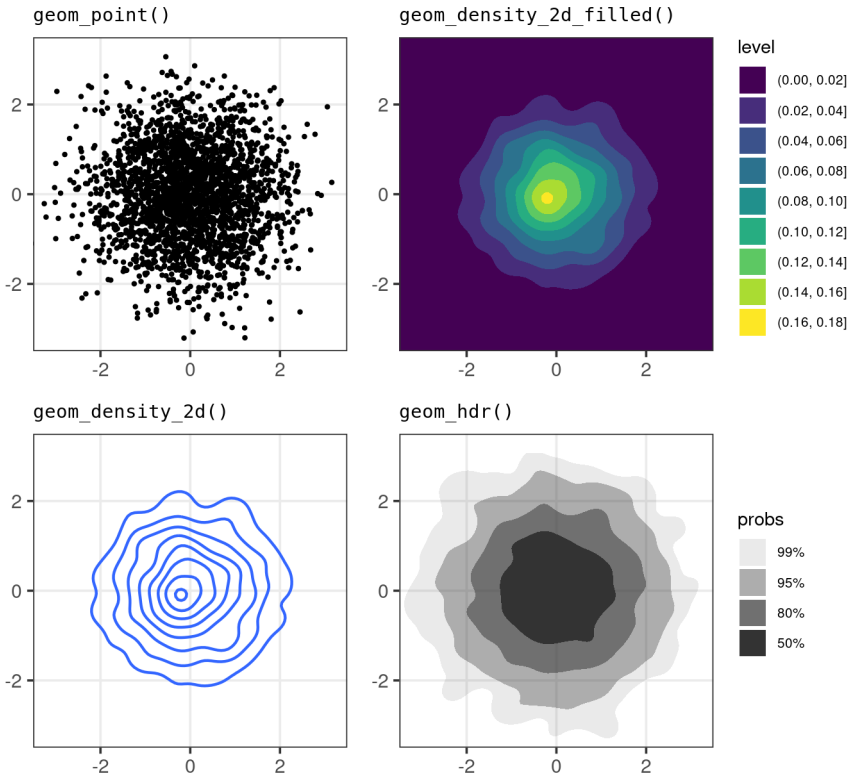

Figure 1: Comparing various geoms on a bivariate standard normal sample of size $n = 2500$.

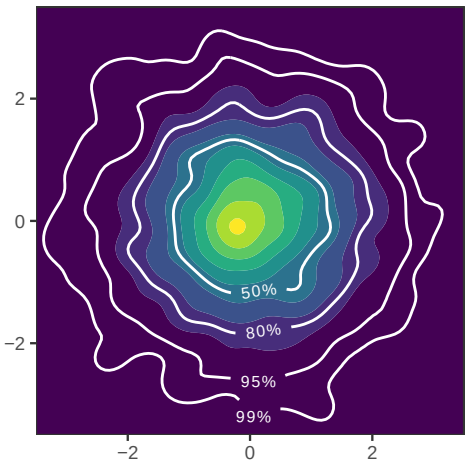

Figure 2: `geom_density_2d_filled()` and `geom_hdr()` (white) from Figure 1, showing that the traditional density contour plots can mislead: nearly 20% of the estimated distribution falls outside the lowest OMDC. Labels generated with **geomtextpath** (Cameron and van den Brand, 2022).

3 Highest density regions

More formally, following Hyndman (1996) we adopt the following definition of highest density regions (HDRs):

Definition 1 *Let $f(\boldsymbol{x})$ be the probability density function (PDF) of a random vector $\boldsymbol{X} \in \mathbb{R}^p$ and $\alpha \in (0,1)$. For any constant $c \in \mathbb{R}$ define $\mathcal{R}_f(c) = \{\boldsymbol{x} \in \mathbb{R}^p : f(\boldsymbol{x}) \geq c\}$ to be the subset of the sample space of $\boldsymbol{X}$ with density at least c. The $100(1-\alpha)\%$ HDR of $\boldsymbol{X}$ is the subset $\mathcal{R}_f(f_\alpha)$, where f_α is the largest constant c such that $p_f(c) := P_f\left[\boldsymbol{X} \in \mathcal{R}_f(c)\right] \geq 1-\alpha$. When $f(x)$ is clear, we will simplify this to $\mathcal{R}_\alpha = \mathcal{R}_f(f_\alpha)$.*

Note that $p_f(c)$ is non-increasing in c. Intuitively, we see that as c gets bigger the region where $f(\mathbf{x}) \geq c$ shrinks (or possibly remains unchanged), and so the probability of the region gets correspondingly smaller (or at least can't get bigger). f_α is the largest point at which this probability is at least $1-\alpha$. We revisit this in the next section.

Of course, in practice we are given data $\mathbf{x}_1, \ldots, \mathbf{x}_n$, ideally a random sample from $f(\mathbf{x})$, and we need to estimate the population HDRs. The problem of estimating HDRs, and more generally density contour estimation, has been widely studied and several estimators have been proposed. Estimators generally fall in one of three classes: plug-in estimators (Rigollet and Vert, 2009; Cadre, 2006), excess mass estimators (Muller and Sawitzki, 1991; Polonik, 1995), and convex contour estimators (Hartigan, 1987). In this work we focus exclusively on plug-in estimators of HDRs due to their straightforward interpretation and implementation. These estimators are of the form $\widehat{\mathcal{R}}_\alpha = \widehat{\mathcal{R}}_f(f_\alpha) := \mathcal{R}_{\hat{f}}(\hat{f}_\alpha)$ for some PDF estimate $\hat{f}(\mathbf{x})$ of $f(\mathbf{x})$, where $\hat{f}_\alpha$ is the largest value c such that $\hat{p}_f(c) := p_{\hat{f}}(c) = P_{\hat{f}}\left[\mathbf{X} \in \mathcal{R}_{\hat{f}}(c)\right] \geq 1-\alpha$.

Thus, plug-in HDR estimators estimate population HDRs with the HDRs of estimated densities. As there are many ways to estimate a density, both parametric and nonparametric, one can arrive at many different HDR estimates with the same data and probability mass $1-\alpha$, and different choices confer advantages and disadvantages. We revisit this notion in HDRs using different density estimators using **ggdensity** after explaining how such estimates can be computed.

Before addressing that topic, it is worth reflecting on another aspect that makes HDRs so special: their size. While HDRs are primarily of interest because their corresponding probabilities are immediately interpretable, they are also important because they are the smallest such sets that contain their probabilities.

For any continuous distribution, there are an infinite number of different regions in its support that contain probability at least $1-\alpha$. The standard normal distribution provides a simple univariate example. If $\Phi^{-1}(x)$ is the quantile function of the standard normal distribution and $0 \leq l < u \leq 1$ with $u - l = 1 - \alpha$, any interval of the form $[\Phi^{-1}(l), \Phi^{-1}(u)]$ contains probability exactly $1-\alpha$, and indeed these are all such intervals that do. This is because $P\left[\Phi^{-1}(l) \leq Z \leq \Phi^{-1}(u)\right] = P\left[Z \leq \Phi^{-1}(u)\right] - P\left[\Phi^{-1}(l) \leq Z\right] = \Phi(\Phi^{-1}(u)) - \Phi(\Phi^{-1}(l)) = u - l = 1 - \alpha$. Setting $1-\alpha = .95$, we can choose $l = 0$, $u = .95$ yielding the interval $(-\infty, 1.645]$. We can find similar intervals with other choices of l, u:

$[-2.326, 1.751]$; $[-2.054, 1.881]$; $[-1.881, 2.054]$; $[-1.751, 2.326]$; and $[-1.645, \infty)$, all of which contain 95% probability.

However, all intervals are typically not equally preferable: if we are interested in summarizing the normal distribution with a set of probability $1-\alpha$, we typically want to provide the smallest such set. The intervals above have lengths ∞, 4.08, 3.93, 3.93, 4.08, and ∞. As is well-known, if we want the smallest interval, we use $[\Phi^{-1}(\alpha/2), \Phi^{-1}(1-\alpha/2)] = [-1.96, 1.96]$ as the interval bounds, with the interval length of 3.92. This interval corresponds to the region where the density exceeds $f_\alpha = \phi(\Phi^{-1}(\alpha/2))$; and as such meets the definition of an HDR.

This is a general feature of HDRs: whether in one or many dimensions, they constitute the smallest regions containing their corresponding probabilities, a fact seen from the following measure-theoretic argument. For some $f(\mathbf{x})$, $\alpha \in [0,1]$, and an associated HDR $\mathcal{R}_\alpha$, let $\eta = P[\mathbf{X} \in \mathcal{R}_\alpha]$. While η may be equal to $1-\alpha$, in some cases it may be more, for instance in the case of the uniform distribution. Suppose $\mathcal{A}$ is a subset of the sample space of $\mathbf{X}$ with probability $\xi \geq \eta$ such that $\mathcal{A} \setminus \mathcal{R}_\alpha$ is non-null, i.e. $\mathcal{A}$ and $\mathcal{R}_\alpha$ are not the same set. Then

$$\eta = P[\mathbf{X} \in \mathcal{R}_\alpha] = \int_{\mathcal{R}_\alpha} f(\mathbf{x})d\mathbf{x} = \int_{\mathcal{R}_\alpha \cap \mathcal{A}} f(\mathbf{x})d\mathbf{x} + \int_{\mathcal{R}_\alpha \setminus \mathcal{A}} f(\mathbf{x})d\mathbf{x}.$$

Similarly,

$$\xi = P[\mathbf{X} \in \mathcal{A}] = \int_{\mathcal{R}_\alpha \cap \mathcal{A}} f(\mathbf{x})d\mathbf{x} + \int_{\mathcal{A} \setminus \mathcal{R}_\alpha} f(\mathbf{x})d\mathbf{x}.$$

Since $\xi \geq \eta$,

$$\int_{\mathcal{A} \setminus \mathcal{R}_\alpha} f(\mathbf{x})d\mathbf{x} \geq \int_{\mathcal{R}_\alpha \setminus \mathcal{A}} f(\mathbf{x})d\mathbf{x}$$

By the definition of $\mathcal{R}_\alpha$, $f_\alpha > f(\mathbf{x})$ over $\mathcal{A} \setminus \mathcal{R}_\alpha$ and $f(\mathbf{x}) \geq f_\alpha$ over $\mathcal{R}_\alpha$. Thus,

$$f_\alpha m(\mathcal{A} \setminus \mathcal{R}_\alpha) = \int_{\mathcal{A} \setminus \mathcal{R}_\alpha} f_\alpha d\mathbf{x} > \int_{\mathcal{A} \setminus \mathcal{R}_\alpha} f(\mathbf{x})d\mathbf{x} \geq \int_{\mathcal{R}_\alpha \setminus \mathcal{A}} f(\mathbf{x})d\mathbf{x} \geq \int_{\mathcal{R}_\alpha \setminus \mathcal{A}} f_\alpha d\mathbf{x} = f_\alpha m(\mathcal{R}_\alpha \setminus \mathcal{A}),$$

where m denotes the Lebesgue measure of the given set, its size. Thus $m(\mathcal{A} \setminus \mathcal{R}_\alpha) > m(\mathcal{R}_\alpha \setminus \mathcal{A})$ and consequently $m(\mathcal{A}) > m(\mathcal{R}_\alpha)$.

The minimality of HDRs comes with a few trade-offs. First, there may be other regions of the sample space with probability between $1-\alpha$ and η, the nominal and actual probabilities of the HDRs respectively, that are smaller than the HDR. This can happen in cases where the PDF is constant on some set of positive measure so that $p_f(c)$ jumps discontinuously over $1-\alpha$. The $100(1-\alpha)\%$ quantile interval of the Unif$(0,1)$ distribution, $[\alpha/2, 1-\alpha/2]$, illustrates this, for instance; it contains probability exactly $1-\alpha$ and is of length $1-\alpha$. By contrast, the $100(1-\alpha)\%$ HDR for the uniform distribution is $\mathcal{R}_\alpha = [0,1]$ for any α, contains 100% of the distribution, and is of length 1. While this is a rare circumstance when using density estimators, it does occur when constructing HDRs from histogram density estimates – they always contain more than nominal probability. The second trade-off is that HDRs are only connected sets for all α if the distribution is unimodal. This is rarely the case for density estimators, so it is common for the lowest probability HDRs, the smallest sets, to be disconnected: two or more intervals in 1D and unions of blobs in 2D. A third challenge is that HDRs are non-trivial to compute. It is to this challenge that we now turn.

4 Computing highest density regions

ggdensity enables the computation and visualization of 2D HDRs based on various plug-in estimators, extending the functionality of **ggplot2**. In order to understand how HDRs are computed in two dimensions, we find it helpful to first illustrate the process in one dimension.

4.1 Computing HDRs in one dimension

Consider the challenge of computing the 95% HDR of the standard normal distribution with PDF $f(x) = \phi(x)$. One way to do this might be to fix a c, determine the interval $[l, u]$ over which $\phi(x) \geq c$, integrate $\phi(x)$ over $[l, u]$, and move c up or down depending on whether the interval contains too much or too little probability. l and u, the boundaries of the interval where $\phi(x) \geq c$, can be determined numerically or algebraically from $\phi(x) = c$, and the integral can be done via numerical integration. f_α, the special c that provides $1-\alpha$ probability, can also be determined numerically in many ways.

Unfortunately, this method doesn't scale well. In general, the set $\mathcal{R}_f(c)$ will not be a simple interval as in the normal case. It will instead be a union of an unknown number of intervals whose boundaries are unknown and hard to determine. Once known, numerical integration can be relied

upon to determine the integral, but only easily in one dimension. In two or more dimensions, the situation is significantly more complex. The region $\mathcal{R}_f(c)$ is implicitly described, and therefore some form of grid-based approximation would be required before numerical integration could be applied. Notice that there are essentially two hard problems here: computing $\mathcal{R}_f(c)$, in the sense of determining a useful description of it, and computing $p_f(c)$.

The basic idea used by **ggdensity** is simple: discretize and compute. In the univariate case, the fundamental algorithm is this:

1. Evaluate $f(x)$ on a regular mesh $\{x_i : i = 1, \dots N\}$ to obtain $f_i = f(x_i)$ over some interval nominally larger than the support of the data to create a table with rows (x_i, f_i) for $i = 1, \dots, N$.
2. Normalize the points into a discrete distribution $p_i = f_i / \sum_i f_i$ to create the table (x_i, f_i, p_i).
3. Sort the N rows of the table (x_i, f_i, p_i) by p_i in decreasing order to obtain $(x_{(i)}, f_{(i)}, p_{(i)})$.[1]
4. Compute the cumulative sum $a_{(k)} = \sum_{i=1}^{k} p_{(i)}$ to create the table $(x_{(i)}, f_{(i)}, p_{(i)}, a_{(i)})$.
5. Estimate f_α with the first $f_{(i)}$ such that $a_{(i)} \geq 1 - \alpha$.

The point masses p_i of the discrete approximation are in fact the areas of the rectangles of a Riemann (i.e. piecewise constant) approximation to $f(x)$ collapsed to individual points: they approximate the probability over a range such as $[x_i - \delta/2, x_i + \delta/2]$, where δ is the resolution of the mesh. Just as Riemann approximations converge to the true value of the integral under very minor conditions on $f(x)$, arbitrarily accurate approximations to f_α can be obtained by setting N suitably large. Because of this, we will refer to approximating f_α as "computing" f_α instead of estimating it, and our notation will not reflect the fact that it is an approximated quantity.

With f_α in hand, the HDR can be approximated in any of a number of ways that are practically equivalent for sufficiently large N. The easiest is to simply union of intervals $[x_i - \delta/2, x_i + \delta/2]$ such that $f_i \geq f_\alpha$. Figure 3 provides an illustration of the process using a very coarse mesh to emphasize the process.

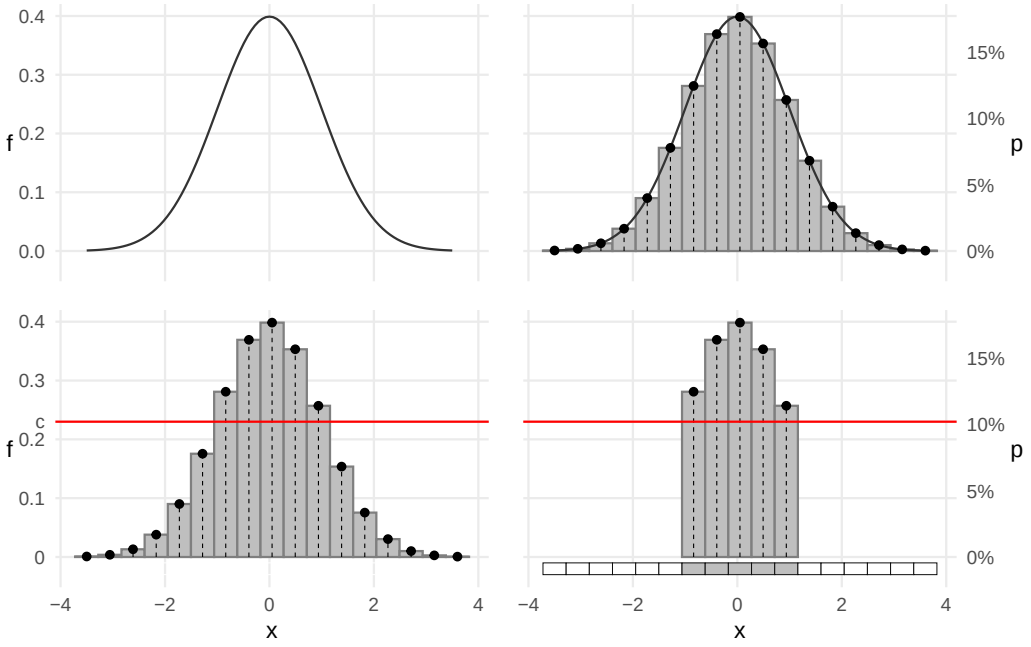

Figure 3: $p_f(c)$ and $\mathcal{R}_f(c)$ are computed by discretizing the density $f(x)$, determining the probabilities with densities above c, and constructing HDRs as unions of intervals. In this illustration, $c = .23$, yielding $p_f(c) = .763$.

[1] This is actually the opposite of order statistics notation: here $p_{(1)}$ is the *largest* probability of the discretized distribution, and $x_{(1)}$ is the corresponding x_i value.

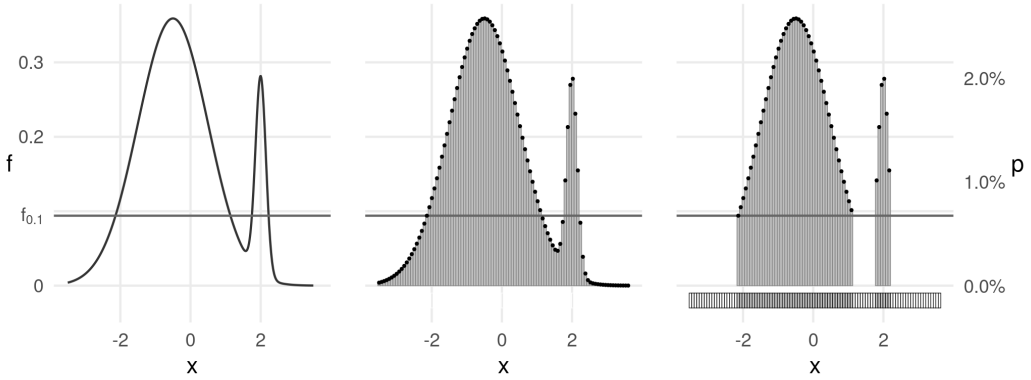

Figure 4: As the mesh size N grows, the HDR approximation improves for any $f(x)$. Here $N = 100$ and the 90% HDR is illustrated.

N governs the accuracy of the approximation of $p_f(c)$ and the accuracy and resolution of $\mathcal{R}_f(c)$ in the resulting plot. Consequently, a reasonably large number is desired: **ggdensity** defaults this parameter to $N = 512$ in `geom_hdr_rug()`. Figure 4 illustrates a more complicated example with a bimodal f where $N = 100$ is used so that it is still possible to see the approximation.

Two adjustments are needed to fully operationalize 1D HDRs for data: using an approximation $\hat{f}(x)$ in place of $f(x)$ and plotting several HDRs. Fortunately, both of these are easy. In practice density estimator implementations usually return their estimates as values of the function $\hat{f}_i = \hat{f}(x_i)$ on a mesh, not as some analytic expression. Such is the case for `stats::density()`, for example, which accepts a univariate vector and returns its estimated density at $N = 512$ points on a regular mesh a little larger than the data. Similarly, computing several HDRs at once requires virtually no added computational expense: HDRs are nested regions, so to find several HDRs simply continue looking down the list of accumulated probabilities $a_{(i)}$ until the desired probabilities are reached. This is illustrated in Figure 5.

Figure 5: Computing several HDRs can be done with no added computational complexity, since the regions are nested. Here the 50%, 80%, and 90% HDRs $\widehat{\mathcal{R}}_\alpha$ are illustrated using an estimated density based on $n = 1000$ draws.

4.2 Finding HDRs in two dimensions

The procedure for finding $\mathcal{R}_\alpha$ for bivariate data is very similar to the univariate case. The basic idea behind the computation of HDRs in **ggdensity** is again to discretize $f(\mathbf{x}) = f(x, y)$ and compute. The fundamental algorithm is this:

1. Evaluate $f(x, y)$ on a fine mesh $\{(x_i, y_j) : i = 1, \dots N_x, j = 1, \dots N_y\}$ to obtain $f_{ij} = f(x_i, y_j)$ over some rectangular region nominally larger than the support of the data to create a table (x_i, y_j, f_{ij}) for $i = 1, \dots, N_x$ and $j = 1, \dots, N_y$. In practice, we usually use $N = N_x = N_y$.
2. Normalize the points into a discrete distribution $p_{ij} = f_{ij} / \sum_{i,j} f_{ij}$ to obtain the table $(x_i, y_j, f_{ij}, p_{ij})$.
3. Sort the rows of the table by p_{ij} in decreasing order to obtain $(x_{(k)}, y_{(k)}, f_{(k)}, p_{(k)})$, where $k = 1, \dots, N_x \times N_y$ and the parenthetical notation is used for consistency with the univariate case. The original order is immaterial to the algorithm.
4. Compute and append the cumulative sum $a_{(k)} = \sum_{i=1}^{k} p_{(i)}$ to the table to obtain the table $(x_{(k)}, y_{(k)}, f_{(k)}, p_{(k)}, a_{(k)})$.
5. Estimate f_α with the first $f_{(k)}$ such that $a_{(k)} \geq 1 - \alpha$.

Similar to the univariate case, the point masses p_{ij} of the discrete approximation can be thought of as the volumes of the rectangular prisms representing a Riemann approximation to $f(x, y)$ collapsed to individual points. And again, just as Riemann approximations converge to the true value of the integral, arbitrarily accurate approximations to f_α can be obtained by setting N_x and N_y suitably large for any reasonable $f(x, y)$. We illustrate this in Figure 6. In practice, $N = N_x = N_y$ governs the resolution of the HDRs in the resulting plot and is set to a suitably large number, **ggdensity** defaults this parameter to 100. Note, too, that this same approach will work in three dimensions and more, however, the computational complexity does not scale into higher dimensions well, and **ggdensity** does not support 3D graphics, so we do not pursue this further here.

In one dimension, the HDR corresponding to the points for which $f(x) \geq f_\alpha$ is naturally described by union of the corresponding rectangular regions of the Riemann approximation. The same could be done in two dimensions, too, as illustrated in Figure 6, however, any contour generating algorithm would work. **ggdensity** uses an implementation of marching squares provided by the **isoband** package (Wilke and Pedersen, 2021). Given a rectangular array of zeros and ones, the basic function implementing the algorithm results in parametrically-described polygonal regions whose interior contains only the points corresponding to the 1's. In the present setting, such an array is provided by the table with $f_{ij} \geq f_\alpha$. This is illustrated in Figure 7.

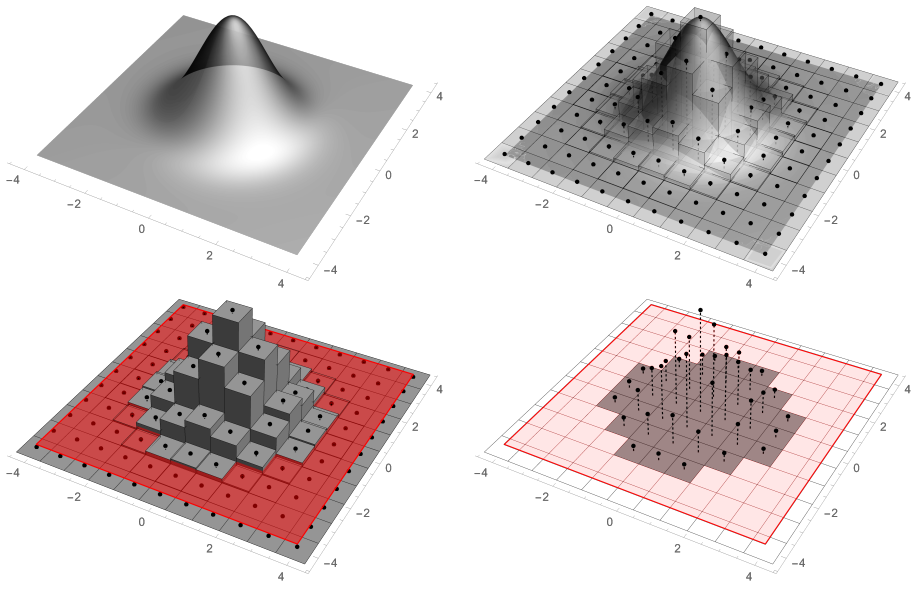

Figure 6: In two dimensions and by analogy with Figure 3, f_α is computed by discretizing the density $f(x, y)$, and HDRs are constructed from points (x_i, y_j) where $f(x_i, y_j) \geq f_\alpha$. Here $\alpha = .05$ was used.

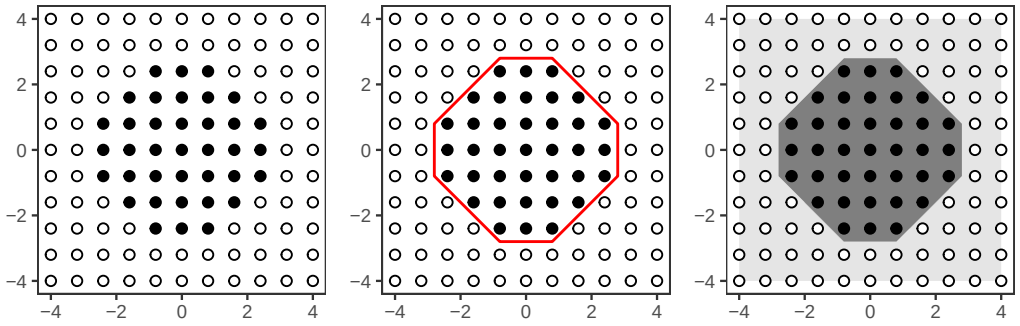

Figure 7: Figure 6 from the plane perspective. **ggdensity** constructs HDRS by applying the marching squares contouring algorithm to binary grid where $f_{ij} \geq f_\alpha$ provided by `isolines()` and `isobands()`.

An alternative approach is worth mentioning at this point. The method described above is essentially what Hyndman (1996) refers to as the "numerical integration approach". However, alternative approaches exist. Hyndman (1996) suggests using the simple, consistent quantile estimate $\hat{f}_\alpha = \hat{f}_{(j)}$, where $\hat{f}_{(j)}$ is the (j/n) sample quantile of $\{\hat{f}(x_i, y_i)\}$ and $j = \lfloor \alpha n \rfloor$. Presented with data $(x_1, y_1), \ldots, (x_n, y_n)$, if $f(x, y)$ is known, any estimate of the $1 - \alpha$ quantile of $f(X, Y)$ is an estimate of f_α; this is referred to as the "density quantile approach". Notice that this requires contours intersect at least one data point and forces a certain proportion of observed values outside of the HDRs, regardless of sample size. Unavailable in **ggdensity**, this method[2] is implemented in both **hdrcde** (Hyndman et al., 2021) and **gghdr** (O'Hara-Wild et al., 2022).

[2]There are many valid choices of $\hat{f}_\alpha$, **hdrcde** and **gghdr** make use of `stats::quantile()` with `type = 7` which estimates a continuous sample quantile function with linear interpolation, a slight modification of the strategy outlined in Hyndman (1996).

4.3 HDRs using different density estimators

As we noted previously, sample HDRs can be computed using many density estimation methods. Figure 8 illustrates how 95% HDRs are calculated for histogram estimators and KDEs on a sample of size $n = 1,000$ from the standard bivariate normal distribution. These use the exact same method as previously described.

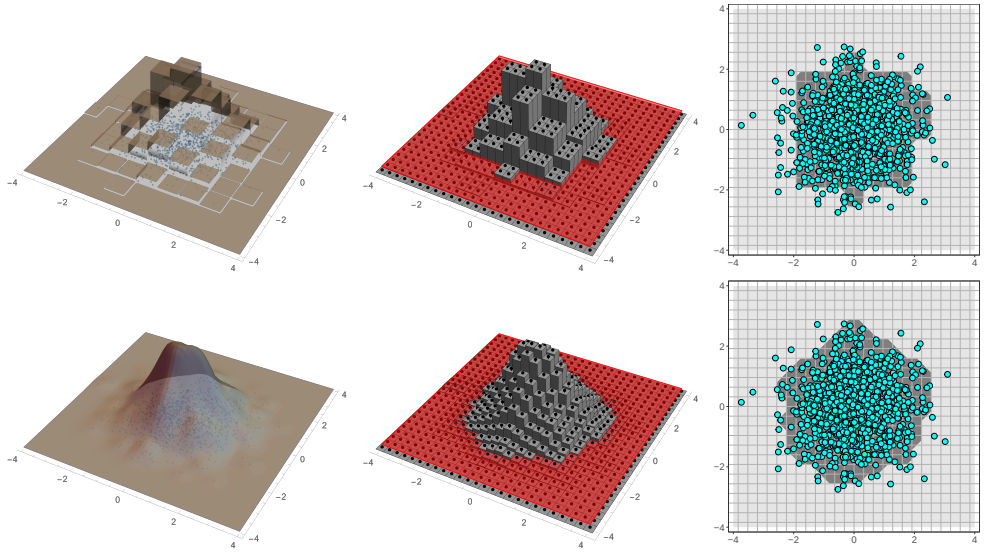

Figure 8: ggdensity facilitates using different density estimators to determine HDRs, including histograms (top, 11 bins in each dimension) and kernel density estimators (bottom), the default. Here 95% HDRs are illustrated using $N = 25$ and $n = 1,000$ draws from the standard bivariate normal distribution. The illustration refects the method of construction, not output of `geom_hdr()`.

Figure 9 displays the output of `geom_hdr()` using the full range of methods available for three different simulated data sets with different features. By default, `geom_hdr()` and `geom_hdr_lines()` plot the 50%, 80%, 90% and 95% HDRs. The methods are available in both `geom_hdr()` and `geom_hdr_lines()` through the `method` argument, which allows for the specification of various nonparametric and parametric estimators, each offering advantages in certain contexts. For example, histogram estimators result in HDRs that obey constrained supports. Normal estimators, i.e. the best-fit bivariate normal estimator, can be helpful in providing simplified visuals that give the viewer a sense of where the distributions are, potentially at the expense of over-simplifying and removing important features of how the variables co-vary.

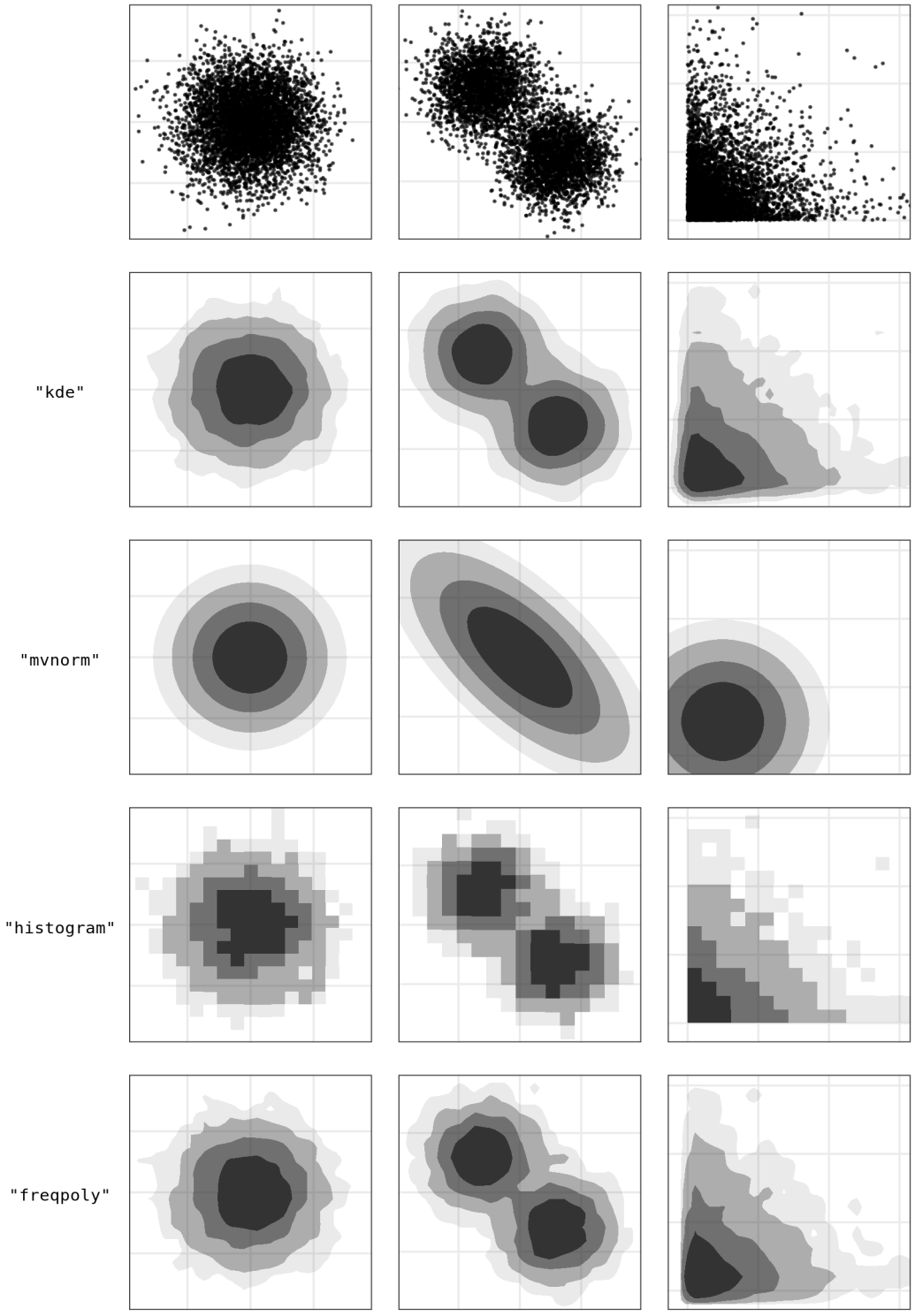

Figure 9: Comparing HDRs obtained with different `method` arguments to `geom_hdr()`.

4.4 HDRs from known density functions

The same method can be used to determine the HDRs of a given density function. This is implemented for bivariate densities in **ggdensity** as `geom_hdr_fun()` and `geom_hdr_lines_fun()`, both accepting a PDF via the `fun` argument. Figure 10 illustrates this by plotting the HDRs of bivariate random vector $\mathbf{X} = (X_1,\ X_2)$ with $X_1 \perp X_2$, $X_1 \sim \mathcal{N}(0,1)$ and $X_2 \sim \text{Gamma}(5,3)$ (left), and $\mathbf{Y} \sim f_{\mathbf{Y}}(y_1, y_2) \propto \exp\left\{-\frac{1}{2(.20)^2}(y_1^2 + y_2^2 - 1)^2\right\}$ (right), which concentrates its probability along the unit circle $\mathcal{S}^1$. In this case we make use of the fact that `geom_hdr_fun()` can find the HDRs of unnormalized PDFs. It does so by leveraging the fact that over a given window, the discretization is not affected by whether or not the density is normalized.

```
f_X <- function(x1, x2) dnorm(x1) * dgamma(x2, 5, 3)
ggplot() + geom_hdr_fun(fun = f_X, xlim = c(-4, 4), ylim = c(0, 5))


f_Y <- function(y1, y2) exp(-1/(2 * .20^2) * (y1^2 + y2^2 - 1)^2)
ggplot() +
  geom_hdr_fun(fun = f_Y, normalized = FALSE, xlim = c(-4, 4), ylim = c(-4, 4)) +
  coord_equal()
```

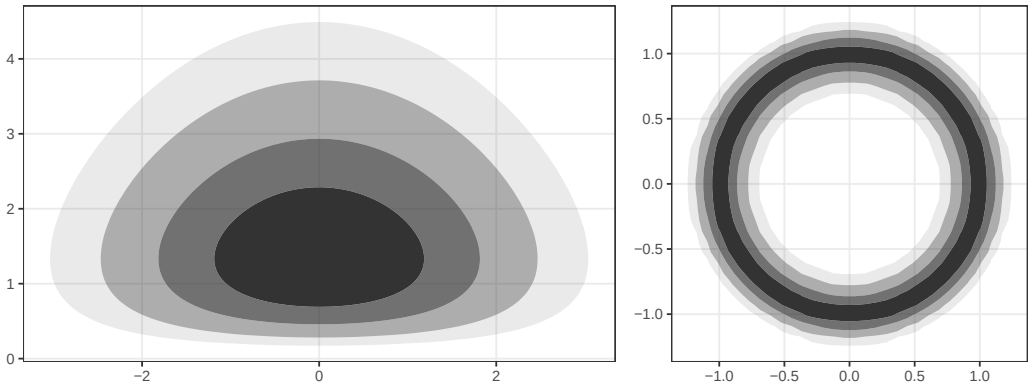

Figure 10: `geom_hdr_fun()` can be used to plot HDRs of normalized and un-normalized known PDFs.

In both of these examples, determining the exact contours is a nontrivial proposition. However, we have not had to derive any results regarding the distributions of $\mathbf{X}$ or $\mathbf{Y}$ to plot them or find the values of f_α – this is all been numerically approximated by **ggdensity** via the previously discussed "numerical integration" method. This represents a simple, powerful tool for visualizing and understanding the probabilistic behavior of arbitrary densities, so long as their support is roughly known.

Beyond the utility of visualizing HDRs of theoretical densities, `geom_hdr_fun()` and `geom_hdr_lines_fun()` can be used to plot HDRs for arbitrary parametric estimates of f. We discuss this at the end of the following section.

5 Further examples

We conclude with a series of more advanced examples that illustrate the flexibility and power of **ggdensity** through more complicated use-cases.

5.1 Comparing populations

Since `geom_hdr()` and `geom_hdr_lines()` use transparency (the `alpha` aesthetic) to communicate probability, color remains available to communicate group membership in the context of more than one population via either the `fill` or `color` aesthetics. This allows for easy comparison of multiple bivariate populations via their HDRs. In Figure 11, we use this strategy to compare the relationship between flipper length and bill length for different species of penguins using the popular Palmer penguins dataset (Horst et al., 2020). In this case `geom_hdr_lines()` is used to reduce overplotting.

As discussed previously, **ggdensity** provides several nonparametric and parametric estimators to compute the HDRs. Figure 11 assumes a bivariate normal distribution, expressed by setting `method = "mvnorm"` in `geom_hdr_lines()`. This implies that each group's HDRs are elliptical and the resulting visualization is a useful approximation of the true distributions. With it we can easily see the general location of each of the groups and that all have similar covariance structures. These details can be obscured when more flexible non-parametric HDR estimators are used, especially when sample sizes are small.

```
ggplot(penguins, aes(flipper_length_mm, bill_length_mm, fill = species)) +
  geom_hdr_lines(aes(color = species), method = "mvnorm") +
  geom_jitter(shape = 21)
```

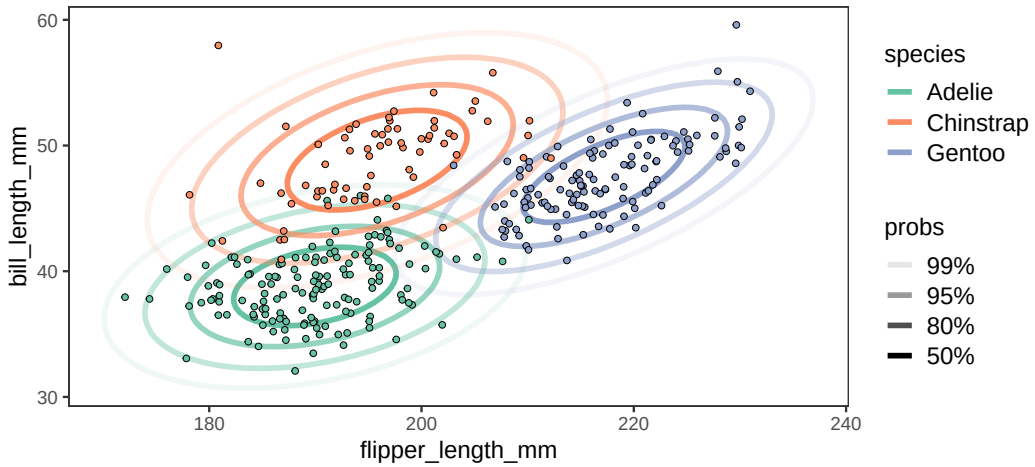

Figure 11: Using `geom_hdr_lines()` with a color aesthetic can be used to reduce overplotting when visualizing the HDRs for different subgroups of data.

5.2 HDRs and goodness of fit

It can be useful to combine `geom_hdr()` and `geom_hdr_lines()` to compare different estimators of f. A powerful example is plotting elliptical contour lines corresponding to an estimated normal model on top of filled contours of the KDE, facilitating a visual exploration of goodness of fit. We have included two examples of this strategy in Figure 12. The left graphic illustrates bill length versus flipper length for the Chinstrap penguins from Figure 11. Notice that the filled contours generally match the contour lines, providing visual evidence towards the validity of an assumption of normality. By contrast, the right graphic explores the relationship between two measurements from a dataset comparing 178 wines from Forina et al. (1986) exported as `wines` in **sn** (Azzalini, 2022). The filled contours do not coincide with the contour lines – the nonparametric estimate of the density is visibly more skewed – indicating that a normal approximation might not be appropriate for this data.

```
penguins |>
  filter(species == "Chinstrap") |>
  ggplot(aes(flipper_length_mm, bill_length_mm)) +
  geom_hdr() +
  geom_hdr_lines(color = "red", method = "mvnorm") +
  geom_jitter(color = "red")


ggplot(wines, aes(uronic, malic)) +
  geom_hdr() +
  geom_hdr_lines(method = "mvnorm", color = "red") +
  geom_jitter(color = "red")
```

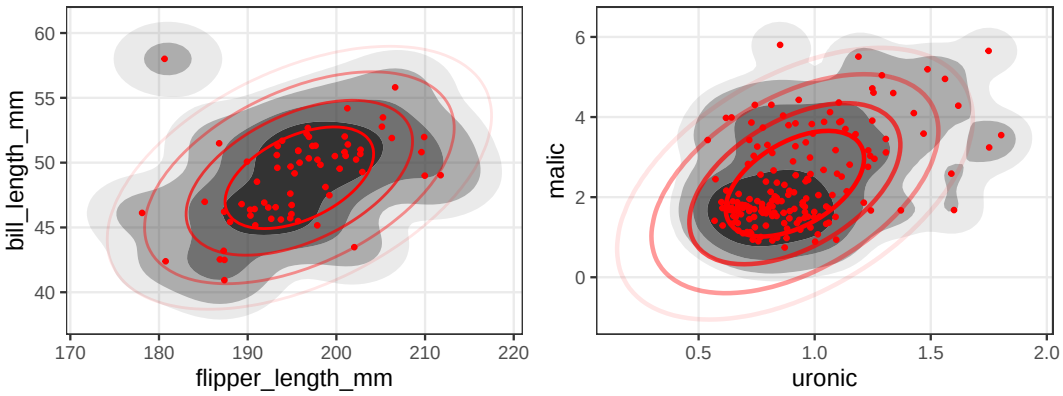

Figure 12: Normality can be visually assessed by layering the HDRs of a KDE (black) with that of a parametrically estimated bivariate normal (red), here illustrated with Palmer penguin data (left) and wines data (right). The points in each plot have been jittered due to rounding in the data, notice that this leads to small inconsistencies between the plotted data and HDRs.

This strategy can be extended to evaluating goodness of fit for arbitrary parametric models via combining `geom_hdr()` and `geom_hdr_lines_fun()`, following the strategy outlined in HDRs for arbitrary parametric models.

5.3 Other related geoms

ggdensity also includes functions `geom_hdr_points()` and `geom_hdr_rug()` for alternative methods of visualizing HDRs[3]. These are illustrated in Figure 13, in which we visualize the old faithful dataset (Azzalini and Bowman, 1990). The left image displays the standard visualization of HDRs from `geom_hdr()`. The graphic in the middle, created by `geom_hdr_points()`, displays the data itself with points colored by their HDR membership–this can be useful in situations where overplotting is a concern. The plot on the right presents the original data with the estimated marginal HDRs via `geom_hdr_rug()`. Note that the scale is the same across all of the plots, with the 50%, 80%, 95%, and 99% HDRs being visualized by default.

```
p <- ggplot(faithful, aes(eruptions, waiting))

p + geom_hdr()
p + geom_hdr_points()
p + geom_hdr_rug()
```

[3]The previously mentioned **gghdr** includes similar tools, we discuss this further in Discussion and future directions.

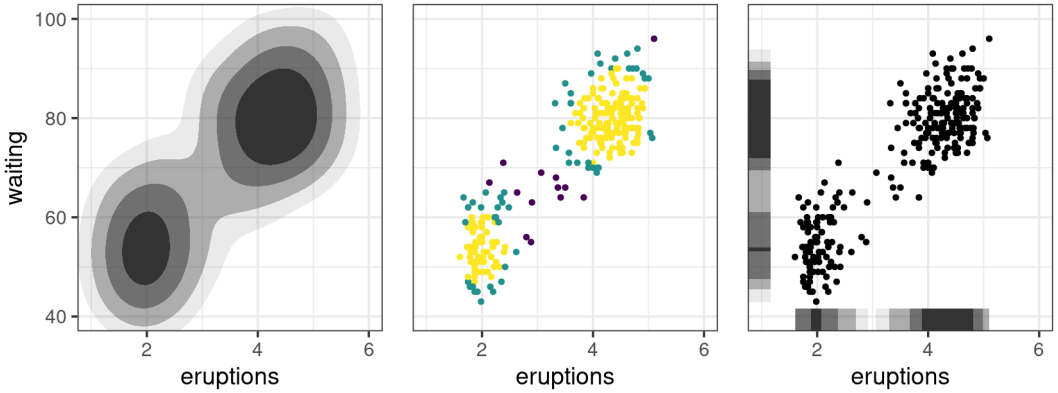

Figure 13: `geom_hdr_points()` and `geom_hdr_rug()` can provide more insight into bivariate scatterplots, as seen here with the faithful dataset.

It is important to note that `geom_hdr_rug()` can also be used when only an `x` or `y` aesthetic is provided. This is illustrated in Figure 14 where the KDE of eruption duration is visualized alongside its estimated HDRs. In this graphic we have chosen to communicate the HDRs via colors – in some cases we have found this to be preferable when using `geom_hdr_rug()`.

```
ggplot(faithful, aes(eruptions)) +
  geom_density() +
  geom_hdr_rug(aes(fill = after_stat(probs)), length = unit(.05, "npc"), alpha = 1) +
  scale_fill_viridis_d(option = "magma", begin = .8, end = 0)
```

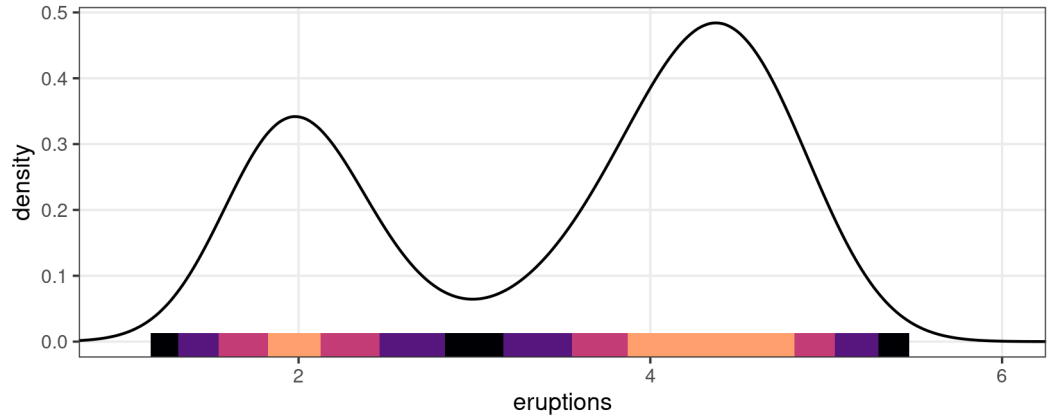

Figure 14: `geom_hdr_rug()` can also improve the visualization of univariate densities.

5.4 HDRs for arbitrary parametric models

Historically, there has been much focus on contour estimation based on non-parametric estimates of f, typically KDEs. To our knowledge, there has been relatively little focus on parametric estimation of HDRs. If a probability model is specified, estimated HDRs are simple to derive from $\hat{f}_{\text{MLE}}$, the density's maximum likelihood estimator (MLE). This allows for the visualization of a much larger class of HDR estimators than those built into `geom_hdr()`; users can specify and estimate arbitrary parametric models and provide the resulting density estimate to `geom_hdr_fun()`.

We include an example of HDRs corresponding to a custom estimated parametric density in Figure 15. Here we generated $n = 100$ draws from a bivariate exponential distribution $(X, Y) \sim f(x, y|\boldsymbol{\theta}) = \text{Exp}(\boldsymbol{\theta})$, estimated $\boldsymbol{\theta}$ with its MLE, and passed the resulting estimate $\hat{f}(x, y) = f(x, y|\hat{\boldsymbol{\theta}})$ to `geom_hdr_fun()` via the `fun` argument.

```
set.seed(1)

df <- data.frame(x = rexp(100, 1), y = rexp(100, 1))

# pdf for parametric density estimate
f <- function(x, y, lambda) dexp(x, lambda[1]) * dexp(y, lambda[2])

# estimate parameters governing joint pdf
lambda_hat <- apply(df, 2, mean)

ggplot(df, aes(x, y)) +
  geom_hdr_fun(fun = f, args = list(lambda = lambda_hat)) +
  geom_point(fill = "lightgreen", shape = 21) +
  coord_fixed() +
  scale_x_continuous(limits = c(0, 7)) +
  scale_y_continuous(limits = c(0, 7))
```

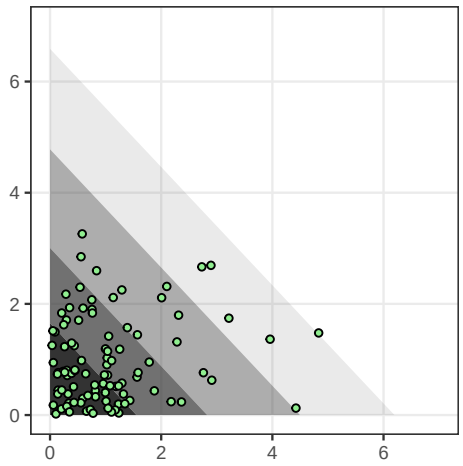

Figure 15: Plotting HDRs of specified distributions can be achieved with `geom_hdr_fun()`.

6 Discussion and future directions

As described in Finding HDRs in two dimensions, **ggdensity** approximates sample HDRs via the numerical integration approach, unlike other software such as **hdrcde** and **gghdr** that both rely on the quantile approach (Hyndman, 1996). In practice the two approaches perform similarly, although there are several technical ways in which they differ. One example is that HDRs estimated via sample quantiles are not guaranteed to contain a certain proportion of estimated density, even if $p_{\hat{f}}(c)$ is strictly decreasing. In other words, there is no guarantee that $\int_{\hat{\mathcal{R}}_\alpha} \hat{f}(\mathbf{x})\, d\mathbf{x} = 1 - \alpha$ when $\hat{\mathcal{R}}_\alpha$ is determined using $\hat{f}_\alpha$ computed via the quantile method. Additionally, as the quantile method requires a sample from f it is not possible to calculate HDRs from arbitrary densities as in `geom_hdr_fun()`. Historically, the density quantile approach has been favored due to computational limitations associated with numerical integration (Hyndman, 1996). However, with modern computing power, this is not a concern anymore – we have found **ggdensity** to be very performant.

With both **ggdensity** and **gghdr** being extensions to **ggplot2** for visualizing HDRs there is overlap in their capabilities. There are analogs to `geom_hdr_rug()` and `geom_hdr_points()` implemented as `gghdr::geom_hdr_rug()` and the helper function `gghdr::hdr_bin()`, respectively. From the user's perspective these implementations are similar, with **ggdensity** offering HDRs estimated via different methods. A more serious distinction between the two is that **gghdr** does not provide a way to plot *bivariate* HDRs in a way similar to `geom_hdr()` or `geom_hdr_lines()`. At present, **ggdensity** is the only package that facilitates the visualization of bivariate HDR contours with **ggplot2**.

Another important difference is that both **gghdr** and **hdrcde** implement visualizations of conditional HDRs, something we plan on implementing in **ggdensity** in the future. These allow users to make visuals similar to regression-style modeling bands. We also plan on extending **ggdensity**'s

capabilities to plot univariate HDRs, implementing something similar to `ggdensity::geom_hdr_rug()` for the main plotting window; this will result in a tool similar to `gghdr::geom_hdr_boxplot()`. This future feature also bears resemblance to the `stat_slabinterval()` family from **ggdist**, another **ggplot2** extension for visualizing densities and their estimates (Kay, 2023). Finally, we also look to implement more density estimators available via the `method` argument, for example skew-normal and mixture models.

References

A. Azzalini. sn: The Skew-Normal and Related Distributions Such as the Skew-t and the SUN, Mar. 2022. URL https://CRAN.R-project.org/package=sn. [p231]

A. Azzalini and A. W. Bowman. A Look at Some Data on the Old Faithful Geyser. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 39(3):357–365, 1990. ISSN 0035-9254. doi: 10.2307/2347385. URL https://www.jstor.org/stable/2347385. Publisher: [Wiley, Royal Statistical Society]. [p232]

B. Cadre. Kernel estimation of density level sets. *Journal of Multivariate Analysis*, 97(4):999–1023, Apr. 2006. ISSN 0047-259X. doi: 10.1016/j.jmva.2005.05.004. URL https://www.sciencedirect.com/science/article/pii/S0047259X05000825. [p222]

A. Cameron and T. van den Brand. *geomtextpath: Curved Text in 'ggplot2'*, 2022. URL https://CRAN.R-project.org/package=geomtextpath. R package version 0.1.0. [p222]

M. Forina, C. Armanino, M. Castino, and M. Ubigli. Multivariate data analysis as a discriminating method of the origin of wines. *VITIS - Journal of Grapevine Research*, 25(3):189–189, 1986. ISSN 2367-4156. doi: 10.5073/vitis.1986.25.189-201. URL https://ojs.openagrar.de/index.php/VITIS/article/view/5950. Number: 3. [p231]

J. A. Hartigan. Estimation of a Convex Density Contour in Two Dimensions. *Journal of the American Statistical Association*, 82(397):267–270, Mar. 1987. ISSN 0162-1459. doi: 10.1080/01621459.1987.10478428. URL https://www.tandfonline.com/doi/abs/10.1080/01621459.1987.10478428. Publisher: Taylor & Francis _eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1987.10478428. [p222]

A. Horst, A. Hill, and K. Gorman. palmerpenguins: Palmer Archipelago (Antarctica) Penguin Data, July 2020. URL https://CRAN.R-project.org/package=palmerpenguins. [p231]

R. Hyndman, J. Einbeck, M. Wand, S. Carrignon, and F. Cheng. hdrcde: Highest Density Regions and Conditional Density Estimation, Jan. 2021. URL https://CRAN.R-project.org/package=hdrcde. [p227]

R. J. Hyndman. Computing and Graphing Highest Density Regions. *The American Statistician*, 50(2):120–126, 1996. ISSN 0003-1305. doi: 10.2307/2684423. URL https://www.jstor.org/stable/2684423. Publisher: [American Statistical Association, Taylor & Francis, Ltd.]. [p220, 222, 227, 234]

M. Kay. *ggdist: Visualizations of Distributions and Uncertainty*, 2023. URL https://mjskay.github.io/ggdist/. R package version 3.2.1. [p235]

D. W. Muller and G. Sawitzki. Excess Mass Estimates and Tests for Multimodality. *Journal of the American Statistical Association*, 86(415):738–746, 1991. ISSN 0162-1459. doi: 10.2307/2290406. URL https://www.jstor.org/stable/2290406. Publisher: [American Statistical Association, Taylor & Francis, Ltd.]. [p222]

M. O'Hara-Wild, S. Pearce, R. Nakagawara, S. Gupta, D. Vanichkina, E. Tanaka, T. Fung, and R. Hyndman. gghdr: Visualisation of Highest Density Regions in 'ggplot2', Feb. 2022. URL https://CRAN.R-project.org/package=gghdr. [p227]

W. Polonik. Measuring Mass Concentrations and Estimating Density Contour Clusters-An Excess Mass Approach. *The Annals of Statistics*, 23(3):855–881, June 1995. ISSN 0090-5364, 2168-8966. doi: 10.1214/aos/1176324626. URL https://projecteuclid.org/journals/annals-of-statistics/volume-23/issue-3/Measuring-Mass-Concentrations-and-Estimating-Density-Contour-Clusters-An-Excess/10.1214/aos/1176324626.full. Publisher: Institute of Mathematical Statistics. [p222]

P. Rigollet and R. Vert. Optimal rates for plug-in estimators of density level sets. *Bernoulli*, 15(4):1154–1178, Nov. 2009. ISSN 1350-7265. doi: 10.3150/09-BEJ184. URL https://projecteuclid.org/journals/bernoulli/volume-15/issue-4/Optimal-rates-for-plug-in-estimators-of-density-level-sets/10.3150/09-BEJ184.full. Publisher: Bernoulli Society for Mathematical Statistics and Probability. [p222]

D. W. Scott. *Multivariate density estimation: theory, practice, and visualization / David W. Scott*. Wiley series in probability and mathematical statistics. Wiley, New York, 1992. ISBN 978-0-471-54770-9. [p220]

W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. URL https://www.stats.ox.ac.uk/pub/MASS4/. ISBN 0-387-95457-0. [p220]

H. Wickham. *ggplot2*. Springer, New York, NY, 2009. ISBN 978-0-387-98140-6 978-0-387-98141-3. doi: 10.1007/978-0-387-98141-3. URL http://link.springer.com/10.1007/978-0-387-98141-3. [p220]

C. O. Wilke and T. L. Pedersen. *isoband: Generate Isolines and Isobands from Regularly Spaced Elevation Grids*, 2021. URL https://CRAN.R-project.org/package=isoband. R package version 0.2.5. [p221, 226]

L. Wilkinson. *The Grammar of Graphics*. Statistics and Computing. Springer-Verlag, New York, 2005. ISBN 978-0-387-24544-7. doi: 10.1007/0-387-28695-0. URL http://link.springer.com/10.1007/0-387-28695-0. [p220]

James Otto
Baylor University
One Bear Place #97140
Waco, TX 77005
https://orcid.org/0000-0002-0665-2515
jamesotto852@gmail.com

David Kahle
Baylor University
One Bear Place #97140
Waco, TX 77005
https://orcid.org/0000-0002-9999-1558
david_kahle@baylor.edu