GetDist: Kernel Density Estimation

Antony Lewis

University of Sussex

(Dated: June 4, 2015)

This note reviews Kernel Density Estimation (KDE) techniques for estimating 1D and 2D densities from samples as implemented in the Python GetDist program. We focus on methods that are applicable to correlated weighted samples, as produced by MCMC sampling and/or importance sampling, with possible boundary priors. The baseline method consists of applying a linear boundary kernel, and then using multiplicative bias correction; bandwidth is selected automatically based on a plug-in method using the expected scaling with an effective number of samples (defined to account for MCMC correlations and weights). Two-dimensional KDE uses an automatically determined plug-in elliptical Gaussian kernel with linear boundary and multiplicative bias correction.

I. WEIGHTED SAMPLES

We work with weighted samples, so each point in parameter space \( X_i \) is associated with a weight \( w_i \). Estimators for the mean of a function \( F(x) \) under the distribution \( f(x) \) are then given by weighted sums over \( n \) sample points:

\[
\hat{\bar{F}} = \frac{1}{N} \sum_{i=1}^{n} w_i F(X_i),
\]

where \( N = \sum_{i=1}^{n} w_i \). Define \( f_p(w_i, X_i) \) as the probability of getting sample point \( X_i \) with weight \( w_i \). For the weighted sum to give the expectation under \( f \) the weights must satisfy

\[
f(X_i) = \frac{1}{\langle w \rangle} \int dw_i w_i f_p(w_i, X_i) = \frac{\langle w(X_i) \rangle f_p(X_i)}{\langle w \rangle},
\]

since then

\[
\langle \hat{F} \rangle = \frac{1}{N} \sum_{i=1}^{n} \int dX_i w_i F(X_i) f_p(w_i, X_i) = \frac{\langle w \rangle}{N} \sum_{i=1}^{n} \int F(X_i) f(X_i) = \frac{n\langle w \rangle}{N} \hat{F} \approx \bar{F}.
\]

Here and below we neglect differences between \( \langle N \rangle = n\langle w \rangle \) and \( N \); convergence to the true mean is guaranteed as \( n \to \infty \). Eq. (2) is true for importance weights where the weights are non-stochastic and given by \( w(X_i) = \alpha f(X_i)/f_p(X_i) \) for arbitrary constant \( \alpha \). It also holds for MCMC chains where \( w_i \geq 1 \) integer weights count the number of steps due to rejected proposals at each point.

II. KERNEL DENSITY ESTIMATION (KDE)

Kernel Density Estimation (KDE) is the standard term for a wide class of non-parametric methods of estimating probability densities from samples, improving on simple histograms by making some weak assumptions about smoothness. There are many reviews of the basics (e.g. see Refs. [1–4]). For analysis of MCMC samples, we can continue to run the MCMC until we have ‘enough’ samples; for good convergence using standard criteria, typically \( O(1000) \) independent-equivalent samples (and even more KDE-equivalent samples, see Sect. [II D]). This is rather larger than many cases where when KDE is applied to small samples of data, and as such more accurate methods that can be unstable with smaller numbers of samples can be used successfully. I’ll start by reviewing some of the basics, then various complications with boundaries, sample correlations, and improved estimators using multiplicative bias correction.

1 https://github.com/cmbant/getdist/

* URL: http://cosmologist.info
The basic ingredient is an estimate \( \hat{f}(x) \) of the form

\[
\hat{f}(x) = \frac{1}{n} \sum_{i=1}^{n} K_h(x - X_i) \approx \frac{1}{n} \sum K_h(x_b)K_h(x - x_b)
\]

where \( \{X_i\} \) are the sampled points, with \( n \) samples in total. It is sometimes called the “Parzen–Rosenblatt” window estimator. The kernel \( K_h \) can be chosen in different ways; GetDist uses (slightly truncated) zero-centred Gaussians by default, with a width parameter \( h \) (or more generally a covariance). The width parameter \( h \) determines how broad the kernel is, and hence how smooth the estimated function is. In practice, assuming we are only interested in low-dimensional densities, to get good scaling with large number of samples, the samples \( \{X_i\} \) can be binned (finely compared to the scale of \( K_h \)), to give sample counts \( H_b(x_b) \) in a set of bins with centres \( x_b \) (with \( n = \sum H_b \)). Also evaluating \( \hat{f} \) as a (finely) binned density we then have a simple convolution that is fast to evaluate using FFTs:

\[
\hat{f} \approx \frac{1}{n} \hat{H} * K_h.
\]

In general we have weighted samples, with each sample having a weight \( w_i \) each, in which case

\[
\hat{f}(x) = \frac{1}{N} \sum_{i=1}^{n} w_i K_h(x - X_i) \approx \frac{1}{N} \hat{H} * K_h
\]

where \( N \equiv \sum_i w_i \), and \( \hat{H} \) is now the weighted sum of the samples in each bin. In the continuum limit the histogram function is \( \hat{H}(x) = \sum_i w_i \delta(x - X_i) \), and we have

\[
\frac{1}{N} \langle H(x) \rangle = \frac{1}{N} \sum_i \int dX_i dw_i f_p(w_i, X_i) \delta(x - X_i) = \frac{\langle w \rangle}{N} \sum_i \int dX_i f(X_i) \delta(x - X_i) \approx f(x).
\]

The KDE estimator of Eq. \( \text{(6)} \) therefore has expectation

\[
\langle \hat{f}(x) \rangle = \langle [K_h * f](x) \rangle,
\]

which converges to \( f(x) \) when \( K_h \) tends to a delta function as the kernel width goes to zero \( (h \to 0) \).

### A. KDE bias and linear boundary kernels

Where there is a boundary, for example a prior on some parameter that it is positive, smoothing over the boundary will give biased results, since there are no samples on one side. Let’s assume our function is of the form

\[
f(x) = B(x) \hat{f}(x)
\]

where \( B \) is zero in the disallowed region, and one in the allowed region\(^2\), and \( \hat{f} \) is a smooth function over the scale of the kernel (and equal to \( f \) where \( B = 1 \)). Series expanding \( \hat{f} \) around \( x \) using its assumed smoothness \( \hat{f}(x - \delta) = \hat{f}(x) - \hat{f}^{(1)}(x) \cdot \delta + \ldots \), from Eq. \( \text{(6)} \) we have

\[
\langle \hat{f}(x) \rangle = \frac{1}{N} \int \langle H(x - \delta) \rangle K_h(\delta) d\delta = \int B(x - \delta) \hat{f}(x - \delta) K_h(\delta) d\delta
\]

\[
= \int B(x - \delta) \left[ \hat{f}(x) - \delta \cdot \hat{f}^{(1)}(x) + \frac{1}{2} \delta^2 \hat{f}^{(2)}(x) + \ldots \right] K_h(\delta) d\delta
\]

\[
= \left[ (K_h * B) \hat{f}(x) - (K_h^2 * B) \hat{f}^{(1)}(x) + \frac{1}{2} (K_h^3 * B) \hat{f}^{(2)}(x) + \ldots \right],
\]

\(^1\) Or directly if the number of points is relatively small. FFTs could also be replaced by fast gauss transforms (see e.g. umiacs.umd.edu/~morariu/figtree/)

\(^2\) \( B \) can be more general. Specifically, for the binned densities it can account for the fraction of the bin allowed by the prior (e.g. \( B = 1/2 \) for points where the prior cuts a bin in half). It could also account for other known locations of sharp features or structure
where $K_{h}^{ijk...} \equiv K_h x^i x^j x^k \ldots$. Away from the boundary so that $B = 1$, we have $(K_h \ast B) = 1$ and $(K_h' \ast B) = 0$ (for symmetric kernels), so the estimator is unbiased to linear order. The second order bias scales with the covariance of the kernel $(K_h^{ij} \ast B \rightarrow \text{cov}(K_h))^{ij}$ and the local curvature of $f$, and describes the broadening of peaks by convolution (hence typically overestimation of the variance). In units of the width of $f$, the second order bias is $O(h^2)$, and hence is small as long as the kernel is narrow enough compared to $f$.

With a boundary, the estimator is biased even at zeroth order. Normalizing by $(K_h \ast B)$ removes the leading bias, but leaves a linear bias if there is a non-zero gradient at the boundary (the convolution makes the shape at the boundary too flat). A simple solution to this is to use a linear boundary kernel [6]: using a non-symmetric kernel near the boundary to remove the bias. Starting with a simple symmetric kernel $K_h$, we can construct a more general kernel

$$K_h'(x) = K_h(x) \left( A_0 + A_1^i x_i + \frac{1}{2} A_2^{ij} x_i x_j + \ldots \right),$$

(12)

and solve for coefficients $\{A\}$ to render the estimator unbiased. In one dimension this is straightforward to quadratic order$^3$, but it gets messy in more dimensions, and the multiplicative correction (described in Sect. [II C] seems to be generally better at removing higher order biases. So here we restrict to linear kernels and set $A_{\geq 2} = 0$. We then have

$$\langle \hat{f}(x) \rangle = \int B(x - \delta) \hat{f}(x - \delta) K_h(\delta) (A_0 + A_1^i \delta_i + \ldots) d\delta$$

(13)

$$= \int B(x - \delta) \left[ \hat{f}(x) - \delta_i f_i^{(1)}(x) + \ldots \right] K_h(\delta) \left( A_0 + A_1^i \delta_i + \ldots \right) d\delta$$

(14)

$$= \left[ (K_h \ast B) A_0 + A_1^i (K_h' \ast B) \right] f_i^{(1)}(x)$$

(15)

Solving for unit response to $\hat{f}$ and zero gradient bias then gives

$$A_0 = \frac{1}{W_0 - W_1^j W_2^j W_2^i} A_1^i = -[W_2^{-1}]^{ij} W_1^j A_0,$$

(16)

where $W^{ij} \equiv (K_h^{ij} \ast B), W_i^l \equiv (K_h \ast B), W_0 \equiv (K_h \ast B)$. The residual bias is then $O(h^2)$, even approaching the boundary. Note that the correction kernel is only different from the starting kernel within a kernel width of the boundary, since $W_0 = 1, W_1 = 0$ for symmetric kernels where $B = 1$. However for generality the terms can also be calculated by full convolutions.

One issue with the linear boundary kernel estimators is that they are not guaranteed to be positive. A simple fix is to impose positivity by using the positive estimate

$$\hat{f}_P \equiv f \exp \left( \hat{f} / \hat{f} - 1 \right),$$

(17)

where $\hat{f}$ is the simple de-biased kernel formed by normalizing by $(K_h \ast B)$ [3]. We also always renormalize so that the kernel density integrates to unity (or has peak normalized to one for convenient plotting). If $\hat{f}_P$ is only used as a pilot estimate for a later higher order estimator, accuracy of $\hat{f}_P$ near the boundary is in any case not critical.

### B. Statistical and total error

To quantify the error in the kernel estimator, people often use the mean integrated squared error

$$\text{MISE} \equiv \int dx \left\langle (f(x) - \hat{f}(x))^2 \right\rangle,$$

(18)

largely because it is convenient to calculate analytically in simple cases. There are contributions from bias and statistical noise. Assume for simplicity there are no boundary priors here, so Eq. (11) gives the leading bias

$$\langle \hat{f} - f \rangle = \frac{1}{2} \left[ \text{Cov}(K_h) \right]^{ij} f_i^{(2)} + \ldots .$$

(19)

---

3 Giving a fourth order kernel, see e.g. [7]
To see the dependence on the smoothing scale \( h \) of the \( d \)-dimensional kernel, we can define the kernel as \( K_h(x) = \frac{1}{Nh} K(x/h) \), and hence

\[
[Cov(K_h)]^{ij} = \frac{1}{h^d} \int d^d x x^i x^j K(x/h) = h^2 [Cov(K)]^{ij}.
\]

The bias is independent of whether the samples are weighted or correlated. The statistical term is more tricky however. For now, just take the sample locations to be independent. Then (taking \( N \) to be non-stochastic)

\[
\int dx \text{ var} \hat{f} = \frac{1}{N^2} \int dx \sum_i \left\{ \int dX_i w_i^2 K^2_h(x - X_i) f_p(X_i) - \left[ \int dx w_i f_p(X_i) K_h(x - X_i) \right]^2 \right\}
\]

\[
= \frac{n(w^2)}{N^2} \int dx' K^2_h(x') - \frac{1}{N} \int dx (\hat{f})^2 = \frac{n(w^2)}{N^2 h^d} R(K) - \frac{1}{N} R(f) + O(h^2/N),
\]

where \( R(K) \equiv \int dy K^2(y) \). We can define an effective sample number

\[
N_{\text{eff}}^{\text{indep}} \equiv \frac{N^2}{\sum_i w_i^2} = \frac{\sum_i w_i^2}{n(w^2)} \approx \frac{N^2}{n(w^2)},
\]

so that the leading statistical variance scales \( \propto 1/N_{\text{eff}}^{\text{indep}} \).

\[
\int dx \text{ var} [\hat{f}(x)] \approx \frac{1}{N_{\text{eff}}^{\text{indep}} h^d} R(K) - \frac{1}{N} R(f) + \ldots.
\]

For small \( h \), the first term dominates.

The total mean integrated error of Eq. (18) is the sum of the bias and statistical terms, and evaluating the leading terms to get the asymptotic mean integrated squared error (AMISE) gives

\[
\text{AMISE} = \int dx \text{ var} \hat{f} + \int dx (\hat{f} - f)^2 = \frac{1}{N_{\text{eff}}^{\text{indep}} h^d} R(K) - \frac{1}{N} R(f) + \frac{h^4}{4} \int dx \left( [Cov(K)]^{ij} f_{ij}^{(2)} \right)^2 + \ldots,
\]

Minimizing this with respect to \( h \) gives \( h \propto [N_{\text{eff}}^{\text{indep}}]^{-1/(4+d)} \), or explicitly an asymptotically-optimal kernel smoothing scale of

\[
h = \left( \frac{R(K) d}{N_{\text{eff}}^{\text{indep}} I} \right)^{1/(4+d)}
\]

where \( I \equiv \int dx \left( [Cov(K)]^{ij} f_{ij}^{(2)} \right)^2 \).

Apart from the scaling with the effective number of samples \( N_{\text{eff}}^{\text{indep}} \), \( h \) also scales with the curvature of \( f \) via the dependence on \( I \): the larger the average squared second derivative, the more structure the gets smoothed out, and hence the smaller \( h \) should be. But remember that this is specific to the the simple linear kernel estimator of Eq. (4), assuming independent sample points.

C. Multiplicative bias correction

Using boundary kernels renders estimates that are unbiased to \( O(h^2) \). However there is still a systematic broadening of peaks, which can lead to systematically overestimated errors unless there are sufficiently many samples that \( h \ll 1 \).

We can do better (or save computing time by generating fewer samples), by using a higher-order estimator.

Note that that the simple estimator is exactly unbiased if the density is flat (or linear). We can therefore try to flatten the density before performing the convolutions. Specifically, doing the multiplicative bias correction to form

\[
\hat{f} = g(K_h * [H/g]),
\]

where \( g \) is hopefully close to the shape of \( f \), so that \( H/g \) is nearly flat. Absent any prior information about the shape, the simplest thing to do is use \( g = \hat{f} \), where \( \hat{f} \) is a standard linear kernel density estimate; the \( \hat{f} \) estimator
then has bias $O(h^4)$ away from boundaries (assuming sufficient smoothness of $f$) \[9\]. To improve the flattening near boundaries, we can take $\hat{f}$ to be the linear boundary kernel estimate from the Sect. \[10\]. In principle the flattening can also be iterated, but for good choice of smoothing widths usually little is to be gained (and iterations will not converge due to random fluctuations being magnified). The simple multiplicative bias correction method compares well with other higher-order kernel methods for many distributions \[7\] and seems to work well in practice as long as the density is indeed sufficiently smooth. In principle different bandwidths can be used for the pilot estimator $g$ and the final estimate $\hat{f}$ (see e.g. \[10\] who recommend $g$ is over-smoothed compared to $\hat{f}$), but for simplicity we take them to be the same. Other approaches to bias reduction are possible, including the ‘data sharpening’ \[11\] \[12\] method, which is a special case of a more general diffusion approach \[13\].

**D. Correlated samples**

In reality, samples from MCMC are correlated. Expressed in weighted form (where weights count the rejections of the next proposal), there are non-trivial weights and correlations between chain positions. Correlations will increase the error. However, perhaps surprisingly, correlations don’t have a dramatic effect on the optimal kernel bandwidth: the main impact of correlation on the variance does not scale with $h$, since correlated errors between nearby $x$ can not be lowered by more smoothing; see Refs. \[14\] \[15\].

The result of Eq. (25), using the $N_{\text{eff}}^{\text{inde}}$ for independent weighted samples (equivalent to Ref. \[15\]) cannot the full the full story however for finite $h$ of practical interest. For example, proposals in orthogonal subdimensions could leave the parameter(s) of interest exactly unchanged between steps even though they appear as different points in the full-dimensional parameter space. This could be remedied by using a parameter-dependent $N_{\text{eff}}^{\text{inde}}$ in Eq.(25), where the weights now count all consecutive identical points in the parameter space of the kernel density. However it is also clear that very small changes in a parameter, for example due to accepted proposals along very nearly orthogonal eigendirections, should contribute nearly the same as exactly identical points. In other words, whether or not the correlation matters when determining the bandwidth depends on the shape of the correlation function; e.g., whether there is high probability for $|X_i - X_{i+k}| < \sigma h X$, or whether the distribution is broad compared to $\sigma X$.

In detail, we have

$$\int \, dx \, \hat{f}^2(x) = \frac{1}{N^2} \int \, dx \sum_{i,j} w_i w_j K_h(x - X_i) K_h(x - X_j) = \frac{1}{N^2} \sum_{i,j} w_i w_j \langle K_h * K_h \rangle (X_i - X_j).$$

Assuming stationarity leads to

$$\int \, dx \, \langle \hat{f}^2(x) \rangle = \frac{n \langle w^2 \rangle}{N^2} \int \, dx' K^2_h(x') + \frac{2}{N^2} \sum_{k=1}^{n-1} (n - k) \langle w_i w_{i+k} [K_h * K_h](X_i - X_{i+k}) \rangle$$

and transforming $K_h \to K$ then gives

$$\int \, dx \, \langle \hat{f}^2(x) \rangle = \frac{R(K)}{N_{\text{eff}}^{\text{inde}} \varrho h^d} + \frac{2}{N^{2} h^d} \sum_{k=1}^{n-1} (n - k) \langle w_i w_{i+k} [K * K](X_i - X_{i+k}) \rangle.$$  \hspace{1cm} (30)

This makes it clear that the result depends on the prevalence of sequence of points within distance $h$ of each other, as determined by the local $K * K$ filter. Note that the last term in Eq. (30) contains a large contribution $\int \langle \hat{f}(x) \rangle^2$ from points that have close to the same value (a fraction $O(h/n)$ of the terms in the sum), even in the absence of correlations. If points separated by $k \lesssim \Delta$ are strongly correlated with $P(X_{i+k}|X_i) \sim \delta(X_{i+k} - X_i)$, the second term also has a contribution $\sim R(K) \Delta / Nh^d$ that is the same order as the first; this limit is what is considered in Ref. \[15\], and accounts for rejection steps that leave the parameter value exactly unchanged. In general we define a parameter subspace dependent effective number of samples, estimated from the samples, as

$$N_{\text{eff}}^X \equiv \sum_i w_i^2 + 2R(K)^{-1} \sum_k \sum_i \langle w_i w_{i+k} [K * K](X_i - X_{i+k})/h - \mu_K \rangle,$$

where the sum over $k$ can be taken only up to order of the correlation length ($O(L^X_h)$ where the terms are significantly non-zero (and hence is reasonably fast to evaluate), and $\mu_K \equiv \langle w_i w_j [K * K](X_i - X_j)/h \rangle$ takes out the $\sim \langle \hat{f} \rangle^2$ contribution expected for uncorrelated samples (estimated here roughly by a sum over widely separated small subset of samples). In the $h \to 0$ limit this definition therefore isolates the term that contributes to the total variance as

$$\int \, dx \, \text{var} \hat{f}(x) \approx \frac{1}{N_{\text{eff}}^X h^d} R(K) + O(1/N),$$

(32)
and hence includes the effect of exactly duplicated samples from MCMC rejection. The definition of Eq. \[31\] obeys consistency under sample-splitting, so it does not matter how samples are grouped up in to weighted samples or split up. More generally, Eq. \[31\] very roughly includes other tight short-range correlation effects from MCMC sampling (but also some additional covariance that is actually mostly \(h\)-independent, which ideally should not affect the bandwidth choice). As defined \(N_{\text{eff}}^X\) does however itself depend on \(h\). We take a fiducial value \(h \approx 0.2\sigma\) for bandwidth selection. Typical values of \(N_{\text{eff}}^X\) typically lie between \(N_{\text{eff}}^{\text{indep}}\) and the \(N_{\text{eff}}\) of Eq. \[50\] that determines the sampling error on parameter means.

**Relation of weights to acceptance probability**

Note that if the (integer) weights are from chain rejections during MCMC (but we neglect correlations between accepted points), then \(P(w) = (1 - a)^{w-1}a\), where \(a\) is the acceptance probability. Evaluating expectations gives

\[
\langle w \rangle = \frac{1}{a}, \quad \langle w^2 \rangle = \frac{2 - a}{a^2}.
\]

So for raw chains, neglecting correlations between accepted points, we have

\[
N_{\text{eff}}^{\text{indep}} \approx \frac{n}{\langle w^2 \rangle} \approx \frac{N\langle w \rangle}{\langle w^2 \rangle} \approx N \frac{a}{2 - a}.
\]

This relates results in terms of weights to results in the literature terms of acceptance probability (e.g. Ref. \[13\]).

**E. Choice of kernel bandwidth**

A good choice of kernel width is import to get good results: too broad, and features are washed out; too narrow, and sampling noise shows up. Recall from Eq. \[25\] that the Parzen-Rosenblatt estimator has bias \(O(h^2)\), and the statistical variance goes as \(O([Nh]^{-1})\). Minimizing with respect to \(h\) gave \(h \propto N^{-1/5}\) (1D case of Eq. \[20\]), corresponding to an overall convergence rate \(\propto N^{-4/5}\). The constant in the optimal width depends on the distribution (and kernel); assuming one dimension and Gaussians gives the rule of thumb for parameter \(X\) (‘normal scale rule’):

\[
h = 1.06\hat{\sigma}_X (N_{\text{eff}}^X)^{-1/5},
\]

where \(h\) is the standard deviation of the Gaussian smoothing Kernel to use and \(\hat{\sigma}_X\) is an estimate of the standard deviation of parameter \(X\). In practice, for potentially non-Gaussian densities, \(\hat{\sigma}_X\) can be set from a variety of scale measures, for example a width based on central quantiles to avoid over-estimation due to broad tails or a more refined method based on order statistics \[16\]. However scale rules can be quite suboptimal for many non-Gaussian densities. We only use a scale rule for choosing a fiducial scale for evaluating Eq. \[31\], following a simplified version of Ref. \[16\] (taking \(\hat{\sigma}_X = \sigma_X \min[1, R_{0.4}/1.048]\), where \(R_x\) is a the smallest parameter range enclosing \(x\) of the probability (\(\min R_{0.4} = 1.048\) for a Normal) and searching over ranges starting at \(p = 0, 0.1, 0.2...0.6\).

An optimal bandwidth choice can be derived using Eq. \[20\]. The only problem here is that the optimal bandwidth depends on second derivatives (I) of the (unknown) density \(f\). Replacing the derivative term with an estimator gives so-called ‘plug-in’ methods, which can perform much better especially for multi-modal distributions. For reviews and variations of methods see e.g. \[13\] [17][19]. The main problem is that to estimate the second derivative you need to use a bandwidth, which gives you a recursive unknown bandwidth problem. Ref. \[13\] present a neat solution, where the optimal bandwidth is obtained as an equation fixed point that can be found numerically, called the “Improved Sheather-Jones” (ISJ) estimate. Using a Discrete Cosine Transform (DCT), this can also efficiently handle leading-order boundary effects, so that boundaries are not mistaken for large second derivatives \[13\]. The method only required one DCT of the binned data and some binned array dot products, and hence is fast; we adopt it as our auto-bandwidth selector\(^4\). The DCT imposes even symmetry about boundaries, so we only use it for the bandwidth choice, not the actual KDE (the linear boundary kernel gives better accuracy by allowing general gradients at the boundaries).

\(^4\) e.g. Note none of the standard python libraries handle boundaries or optimization well, see https://jakevdp.github.io/blog/2013/12/01/kernel-density-estimation/
FIG. 1: Left: a set of test Gaussian-mixture distributions, comparing the true distribution (red) with the density estimate using 10000 independent samples (black) using multiplicative bias correction and a linear boundary kernel. The Gaussian panels at the bottoms are truncated Gaussian distributions. Right: scaling of the average integrated squared error $\langle (\Delta P)^2 \rangle / \int P^2$ of the density estimate, where 1000 simulations are run of 10000 samples for each estimation. The $x$-axis is a scaling away from the automatically chosen kernel width (e.g. by Eq. (36)), so that one corresponds to the performance with default settings. Lines compare different kernel estimates: solid lines a multiplicative bias correction (MBC) and linear boundary kernel (black: default, blue: next-order multiplicative bias correction). Dotted red is the basic Parzen kernel (for which the plug-in kernel-width estimator is optimizing), magenta is with linear boundary correction, and dashed cyan is a second-order kernel. The MBC kernel width is sub-optimally chosen for Gaussian, where the leading bias term happens to be zero, but about right in many other cases. [see getdist_tests/test_distributions.py]

With multiplicative bias correction the bias is higher order, with bias $O(h^4)$, so the total error scales as $Ah^8 + B/(Nh)$. Optimization now gives $h \propto N^{-1/9}$ and overall convergence $\propto N^{-8/9}$. Again the proportionality constant will depend on the distributions, various examples are given in Ref. [20]. As a first guess we can take the rule of thumb

$$h = h_{ISJ}(N_{eff}^X)^{1/5-1/9},$$

(In general we can replace 1/9 with 1/(4p + 1) for a higher order estimator where the leading bias goes as $h^{2p}$). These smoothing widths are larger than for the basic Parzen-Rosenblatt estimator, and have lower statistical noise (the basic estimator is forced to have smaller widths to avoid significant bias). For $N_{eff}^X \sim O(1000)$, the smoothing width is about twice as broad as the basic estimator. A more refined estimate could be made analogously to the ISJ
multivariate bandwidth matrix

We could also define the Kernel in terms of an isotropic Gaussian kernel $K(x) = K(|x|)$ and a kernel matrix $M$, with $K_M(x) = |M|^{-1/2} K(M^{-1/2} x)$, so that

$$\int dx \text{var} \hat{f}(x) = \frac{1}{n} |M|^{-1/2} \int dy K^2(y) + \cdots \equiv \frac{1}{n} |M|^{-1/2} R(K) + \ldots. \quad (37)$$
See e.g. Ref. [21]. If $K(x)$ has identity covariance, $\text{cov}(K) = I$, then $M$ is just the covariance of $K_M(x)$; hence

$$\text{AMISE} = \frac{1}{N_{\text{ef}}} |M|^{-1/2} R(K) + \frac{1}{4} \int \text{d}x \left( M^{ij} f_i(x)^2 \right)^2 + \ldots$$  \hspace{1cm} (38)

If we parameterize the Gaussian kernel covariance as $M = \left( \begin{array}{cc} h_x^2 & ch_x h_y \\
ch_x h_y & h_y^2 \end{array} \right)$, Eq. (38) becomes

$$\text{AMISE} = \frac{1}{4N_{\text{ef}} \pi h_x h_y \sqrt{1-c^2}} + \frac{1}{4} \left[ h_x^4 \psi_{4,0} + h_y^4 \psi_{0,4} + 2 h_x^2 h_y^2 (2 c^2 + 1) \psi_{2,2} + 4 c h_x h_y (h_x^2 \psi_{3,1} + h_y^2 \psi_{1,3}) \right],$$  \hspace{1cm} (39)

where we defined $\psi_{m_1,m_2}$ as

$$\psi_{m_1,m_2} = (-1)^{i+j} \int \text{d}x \left( \frac{\partial^{i+j}}{\partial x_1^i \partial x_2^j} f(x) \right) \left( \frac{\partial^{p+q}}{\partial x_1^p \partial x_2^q} f(x) \right) = \int \text{d}x f(x) \left( \frac{\partial^{p+q+i+j}}{\partial x_1^p \partial x_2^q \partial x_1^i \partial x_2^j} f(x) \right)$$  \hspace{1cm} (40)

assuming no boundary terms, where $m_1 = p + i$ and $m_2 = q + j$ and $m_1 + m_2$ is even. For $m_1$ and $m_2$ both even, $\psi_{m_1,m_2}$ can be evaluated following the fixed point method of Ref. [13], where we assume an isotropic Gaussian kernel for evaluation $\psi_{m_1,m_2}$. For the odd elements, the analogous argument to Ref. [13] (Appendix E) using Eq. 3.2 from [22] gives an equation for the bandwidth for estimating $\psi_{m_1,m_2}$ as

$$h_{m_1,m_2} = \left( \frac{3N^2}{2m_1 + 2m_2 + 2} \right) \left( \frac{\psi_{0,0} R(K(m_1,m_2))}{\psi_{m_1,m_2 + 2} \psi_{m_1 + 2,m_2}} \right)^{1/(2m_1 + 2m_2 + 6)}$$  \hspace{1cm} (41)

where

$$R(K(m_1,m_2)) = \frac{(2m_1 - 1)!!(2m_2 - 1)!!}{2m_1 + 2m_2 + 2 \pi}$$  \hspace{1cm} (42)

and $\psi_{0,0}$ can be estimated using the method for even elements.

In the case that the correlation $c$ is zero, Eq. (39) can be optimized analytically to give [22]

$$h_x = \left[ \frac{\psi_{0,4}^{3/4} R(K)}{\psi_{4,0}^{3/4} \left( \psi_{4,0}^{1/2} / \psi_{0,4}^{1/2} \right)} \right]^{1/6} \quad h_y = \left( \psi_{0,4}^{1/4} / \psi_{4,0} \right)^{1/4} h_x.$$  \hspace{1cm} (43)

In the general correlated case the minimum must found numerically. In the specific case that the target distribution is Gaussian, the optimal Gaussian bandwidth matrix covariance is [21]

$$M = C(N_{\text{ef}})^{-1/3}$$  \hspace{1cm} (44)

where $C$ is the sample covariance. This can be used to define a rule of thumb for Gaussian-like distributions, but in general (especially in the multi-modal case) can be very bad.

There are several other issues here

- The bias term in Eq. (39) is not guaranteed to be positive if $c \neq 0$, so numerical minimization can fail. (see Ref. [23] for a possible alternative solution)

- With boundaries, the even $\psi_{m_1,m_2}$ derivative terms can be approximated by imposing reflection boundary conditions (i.e. evaluating using DCT), but with $m_1$ or $m_2$ odd, $\psi_{m_1,m_2}$ cannot be evaluated from the DCT transform (which assumes symmetry by construction). They can be evaluated by FFT if there are no sharp boundaries, but there is no easy way to approximately account for boundaries in this case.

- Since the $\psi_{m_1,m_2}$ are evaluated using isotropic Gaussian kernels, they may be rather inaccurate if the optimal kernel is strongly elliptical.

We therefore adopt the following strategy:

- Assuming there are no boundaries, or a boundary in only one of the $x$ or $y$ directions (but not both), use the sample covariance to perform a Cholesky parameter rotation to define uncorrelated transformed variables. The Cholesky rotation is chosen so that if $x$ or $y$ has a boundary it remains unchanged, so the boundary in the transformed parameters remains parallel to the edge of the DCT box. The transformed samples are scaled (so roughly isotropic) and binned, so that evaluation of $\psi_{m_1,m_2}$ using an isotropic kernel is not too suboptimal.
• If there is a boundary the even derivatives are evaluated following Ref. [13] by DCT, and the optimal diagonal bandwidth matrix evaluated from Eq. (43). This is then rotated back to the original coordinates.

• If there are no boundaries, the even and odd $\psi_{m_1,m_2}$ derivatives are estimated, and (39) is minimized numerically. If this fails, Eq. (43) is used as a fall back. The bandwidth matrix is then rotated back to the original coordinates.

• If there are boundaries in both the $x$ and $y$ directions, variable transformation cannot preserve both boundaries, so the samples are not transformed. The diagonal form of Eq. (43) is evaluated on the untransformed samples, unless the sample correlation is very high, in which case a Gaussian rule of thumb bandwidth is assumed using the sample covariance.

The expected asymptotic scaling of the optimal bandwidths are $h \propto N^{-1/6}$ and $h \propto N^{-1/10}$ respectively for methods with quadratic and quartic bias. With multiplicative bias correction we therefore scale the elements of the bandwidth matrix determined above to give

$$h_{x,y} = 1.1 h_{x,y}^{BS1}(N_{\text{eff}})^{1/6-1/10},$$

(45)

where the 1.1 factor is empirically chosen. (In general we can replace $1/10$ with $1/(2p+2)$ for a higher order estimator where the leading bias goes as $h^{2p}$). See Fig. 1 for performance on typical distributions, showing that Eq. (45) slightly underestimates the bandwidth for a Gaussian distribution (and tail-truncated Gaussians), but is a reasonable compromise for most other cases and gives significant performance gains compared to the basic Parzen estimator.

### III. CORRELATION LENGTHS AND SAMPLING ERROR ON PARAMETER MEANS

From MCMC, possibly with post importance-sampling, the samples generally have non-trivial weights and non-trivial correlations. Consider a sample estimate for the mean $\bar{X}$ of a parameter $X$, given by

$$\bar{X} = \frac{1}{N} \sum_{i=1}^{n} w_i X_i.$$  

(46)

From independent unit-weight samples, the variance of the mean estimator is $\sigma_{\bar{X}}^2/N$; we can use this to define an effective $N_{\text{eff}}$ for the correlated weighted samples. The variance of $\bar{X}$ is given by

$$\langle (\bar{X} - \bar{X})^2 \rangle = \frac{1}{N^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \langle w_i (X_i - \bar{X}) w_j (X_j - \bar{X}) \rangle.$$  

(47)

Defining $d_i \equiv w_i (X_i - \bar{X})$, for chains in equilibrium we should have $\langle d_i d_j \rangle = C_d(|i-j|)$, where $C_d(k)$ is the autocorrelation function at lag $k$. Using this

$$\langle (\bar{X} - \bar{X})^2 \rangle = \frac{1}{N^2} \left[ nC_d(0) + 2 \sum_{k=1}^{n-1} (n-k)C_d(k) \right]$$  

(48)

If we assume that the correlation length is much shorter than the chain length\(^5\), so $k \ll n$ for terms which matter, this is

$$\langle (\bar{X} - \bar{X})^2 \rangle \approx \frac{n}{N^2} \left[ C_d(0) + 2 \sum_{k=1}^{\infty} C_d(k) \right].$$  

(49)

We define this to be equal to $\sigma_{\bar{X}}^2 / N_{\text{eff}}$ so that

$$N_{\text{eff}} \approx \frac{n^2 \sigma_{\bar{X}}^2}{n[C_d(0) + 2 \sum_{k=1}^{\infty} C_d(k)]}$$  

(50)

\(^5\) Actually we don’t need to do this, the finite estimator for the autocorrelation from the samples follows the original expression.
We can also define a correlation length by

\[ L_d^w \equiv \frac{n}{N\sigma_X} \left[ C_d(0) + 2 \sum_{k=1}^{\infty} C_d(k) \right], \]  

so that \( N_{\text{eff}} = N/L_d^w \). For unweighted samples \( L_d^w \) corresponds to the standard definition of the correlation length. For importance sampled chains, it is the length in ‘weight units’ (it scales with the arbitrary normalization of the importance weights). We can also define a correlation length \( L_d^x \) in ‘sample units’, so that \( N_{\text{eff}} = n/L_d^x \), which gives an idea of how independent the different points are.

In general we use (50), which can be estimated quickly using weighted sample convolutions and allows for both correlations and importance weights. Of course in general there are correlations between parameters, so this is just an estimate for a single parameter, and will in general be optimistic.


