Shape-Adaptive Kernel Density Estimation

L.E.N. Baakman

July 12, 2017

Abstract

Kernel density estimation has gained popularity in the past few years. Generally the methods use symmetric kernels, even though the data of which the density is estimated are not necessarily spread equally in all dimensions. To account for this asymmetric distribution of data we propose the use of shape adaptive kernels: kernels whose shape changes to fit the spread of the data in the local neighborhood of the point whose density is estimated. We compare the performance of the shape adaptive kernels on both artificial and real world datasets with the performance of symmetric kernels on the unprocessed datasets and on the whitened datasets.

Results

Conclusion

textwidth in cm: 14.92038cm textheight in cm: 22.70108cm columnwidth in cm: 7.28448cm

1 Introduction

Estimating densities with kernels has been fairly popular of late; in the medical field it has been used to predict dose-volume histograms, which are instrumental in the determination of radiation doses [5]. Ecologists have applied it to explore the habitats of seabirds [4]. Ferdosi et al. [3] have described it as “a critical first step in making progress in many areas of astronomy." Within this discipline density estimation is, among other things, used to estimate the density of the cosmic density field, which is required for the reconstruction of the large-scale structure of the universe.

Formally the aim of density estimation is to find the probability density $f(\mathbf{x})$ in the d-dimensional Euclidean space underlying N points $\mathbf{x}_1, \ldots, \mathbf{x}_N$, that have been selected independently from $f(\mathbf{x})$.

Kernel density estimation methods approximate $f(\mathbf{x})$ by placing bumps, referred to as kernels, on the different observations and summing these bumps to arrive at a final density estimate. This paper is concerned with a method to make the shape of these bumps adaptive to the local neighborhood of $\mathbf{x}$. Before introducing the process used to determine the shape of the kernel we first review the different symmetric kernel density estimation methods that our approach is based on.

The Parzen approach [6] is one of the most simple kernel density estimation methods. It estimates the density of $\mathbf{x}$ according to:

$$\hat{f}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} h^{-d} K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right). \qquad (1)$$

The shape of the bumps is determined by the kernel function $K(\bullet)$, its width by the bandwidth h. The Parzen approach requires the kernel to be a probability density function, i.e. $K(\mathbf{x}) \geq 0$ and $\int K(\mathbf{x}) = 1$ [7]. The bandwidth directly influences the result of the density estimation process; a too small bandwidth results in a density estimate with spurious fine structures, whereas kernels that are too wide can oversmooth the density estimate. Kernel estimators, such as the Parzen approach, that use kernels of the same width for all $\mathbf{x}_j$, are called fixed-width estimators.

One downside of fixed-width methods is that they cannot respond appropriately to variations in the magnitude of the density function, i.e. the peakedness of the kernel is not data-responsive. Consequently in low density regions the density estimate will have peaks at the few sample points and be too low elsewhere. In areas with high density, the sample points are more densely packed together, which causes the Parzen estimate to spread out [1]. Adaptive-width methods address this disadvantage of the fixed-width methods by allowing the width of the kernel to vary per data point. For example the estimator introduced by Breiman, Meisel, and Purcell uses the distance between $\mathbf{x}_i$ and the k-nearest

neighbor of $\mathbf{x}_i$, denoted by $D_{i,k}$, to adapt the width of the kernel:

$$\hat{f}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N}(\alpha \cdot D_{i,k})^{-d} K_{\mathcal{G}}\left(\frac{\mathbf{x}-\mathbf{x}_i}{\alpha \cdot D_{i,k}}\right). \quad (2)$$

In Equation (2) $K_{\mathcal{G}}$ is used to represent a Gaussian kernel, and α is a multiplicative constant. The values of both α and k can be determined with a minimization algorithm on a goodness of fit statistic. Comparing Equation (1) with (2) one finds that the bandwidth h of the Breiman estimator is defined as $\alpha D_{i,k}$. The second factor of this bandwidth depends on the local neighborhood of $\mathbf{x}_i$. In low density regions $D_{i,k}$ is large, and the kernel spreads out due to its high bandwidth. In areas with relatively many data points the converse occurs.

Silverman [7] shows that the minimization procedure used by Breiman, Meisel, and Purcell implicitly uses a k-NN pilot estimate. If pilot estimates are used explicitly the density estimation process becomes:

(i) Compute pilot densities with some estimator that ensures that $\forall i\ \tilde{f}(\mathbf{x}_i) > 0$.

(ii) Define local bandwidths λ_i as

$$\lambda_i = \left(\frac{\tilde{f}(\mathbf{x}_i)}{\mathrm{GM}\left(\tilde{f}(\mathbf{x}_0),\ldots,\tilde{f}(\mathbf{x}_N)\right)}\right)^{-\beta}, \quad (3)$$

where GM denotes the geometric mean and the sensitivity parameter β must lie in the range $[0,1]$.

(iii) Compute the adaptive kernel estimate as

$$\hat{f}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N}(h\cdot\lambda_i)^{-d} K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h\cdot\lambda_i}\right) \quad (4)$$

with K integrating to unity.

Since the pilot densities computed in step (i) do not need to be sensitive to the fine details of the pilot estimate a convenient method, e.g. the Parzen approach, can be used to estimate them [7]. The local bandwidths computed in (ii) depend on the exponent β. The higher this value is the more sensitive the local bandwidths are to variations in the pilot densities. Choosing $\beta = 0$ reduces Equation (4) to a fixed-width method. In the literature two values of β are prevalent. Breiman, Meisel, and Purcell argue that choosing $\beta = {}^1/_d$ ensures that the number of observations covered by the kernel will be approximately the same in all areas of the data. Whereas Silverman favors $\beta = {}^1/_2$ independent of the dimension of the data, as this value results in a bias that can be shown to be of a smaller order than that of the fixed-width kernel estimate.

One disadvantage of the Breiman estimator is its computational complexity. This is partially due to the use of a Gaussian kernel. Because of the infinite base of this kernel an exponential function has to be evaluated N times to estimate the density of one data point. The Modified Breiman Estimator (MBE), introduced by Wilkinson and Meijer [8], reduces this computational complexity in two ways. Firstly they replace the infinite base Gaussian kernel with a spherical Epanechnikov kernel in both the computation of the pilot densities and the final density estimate. They define this kernel as:

$$K_{\mathcal{E}}(\mathbf{x}) = \begin{cases} \frac{d+2}{2c_d}(1-\mathbf{x}\cdot\mathbf{x}) & \text{if } \mathbf{x}\cdot\mathbf{x} < 1 \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

where c_d denotes the volume of the d-dimensional unit sphere. It should be noted that the kernel defined in Equation (5) does not have unit variance. this can be corrected by multiplying the bandwidth, h, with the square root of the variance of $K_{\mathcal{E}}$, i.e. $\sqrt{{}^{16}/_{21}}$. There are two advantages to using this kernel, firstly it is computationally much simpler than the Gaussian kernel, in part due to its finite base and secondly it is optimal in the sense of the Mean Integrated Square Error (MISE) [2]. A disadvantage of this kernel is that it is not continuously differentiable. This is irrelevant when computing the pilot densities, as they are only used to determine the local bandwidths. In the computation of the final densities it is a trade off between a continuously differentiable $\hat{f}$ and a low computational complexity.

The second change Wilkinson and Meijer introduce is the indirect computation of the pilot densities. They first compute the pilot densities for the vertices of a grid that covers all data points, before determining the actual pilot densities by multi-linear interpolation. The bandwidth of the kernel used in the computation of the pilot densities is defined as

$$h = \sigma \cdot N^{-1/(d+4)}\left(\frac{8(d+4)\cdot\left(2\sqrt{\pi}\right)^d}{c_d}\right)^{\frac{1}{d+4}}, \quad (6)$$

where σ represents the square root of the average of the variances of the different dimensions. Wilkinson and Meijer estimate the final densities Equation (4) using the general and local bandwidths estimated with Equation (6) and (3), respectively.

Ferdosi et al. [3] consider the application of density estimation on large datasets, i.e. sets with more than 50 000 points with the dimension of the data points ranging from ten to hundreds of elements.

They use the MBE, but introduce a computationally less complex method to estimate the bandwidth. First they determine an intermediate bandwidth for each dimension l of the data:

$$h_l = \frac{P_{80}(l) - P_{20}(l)}{\log N}, l = 1, \ldots, d, \qquad (7)$$

where $P_{20}(l)$ and $P_{80}(l)$ are the twentieth and eightieth percentile of the data in dimension l, respectively. To avoid oversmoothing the pilot window width is then defined as:

$$h = \min_l h_l.$$

Although the widths of the kernels of the discussed adaptive-width methods are sensitive to the data, the shapes of the kernels depend only on the kernel itself. To further increase the responsiveness of the estimator to the data we propose the use of shape-adaptive kernels in density estimation. Not only the width but also the shape of these kernels is steered by the local neighborhood of the data.

A possible disadvantage of these shape-adaptive kernels is that in regions where the density of sample points is low, the number of data points is insufficient to reliably compute the shape of the kernel. Therefore we let the amount of influence exerted by the local data on the shape of the kernel depend on the number of data points in the local neighborhood.

This paper is organized as follows. Section 2 introduces the proposed shape-adaptive kernels. The experiments used to investigate the performance of these kernels are discussed in Section 3, their results are presented in Section 4. The discussion of these results can be found in Section 5. The paper is concluded in Section 6.

2 Method

We use shape adaptive kernels in combination with the Modified Breiman Estimator introduced by Wilkinson and Meijer [8] with the general bandwidths computed according to the method introduced by Ferdosi et al. [3] for its lower computational complexity. The pilot densities are computed on a $20 \times \cdots \times 20$ grid that covers all data points. We have empirically determined that using $\beta =$ works best in our case. The final density estimated is estimated with an Epanechnikov kernel according to:

hoe hebben we dat vastgesteld

Een of andere waarde

$$\hat{f}(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{\det(\lambda_i \cdot \mathbf{H})} K\left((\lambda_i \cdot \mathbf{H})^{-1} (\mathbf{x} - \mathbf{x}_i)\right). \qquad (8)$$

The shape of the kernel $K(\bullet)$ is determined by the bandwidth matrix $\mathbf{H}$. If a fixed width method is used $\lambda_i = 1$. Equation (8) reduces to the fixed-width Parzen estimator defined in Equation (1), if $\mathbf{H} = h \cdot \mathbb{I}_{d \times d}$.

For each data point $\mathbf{x}$ of which the density is estimated the bandwidth matrix is determined according to these steps:

(i) Find $C_\mathbf{x}$, the k-nearest neighbors of $\mathbf{x}$.

(ii) Compute Σ, the unbiased covariance matrix of the local neighborhood $C_\mathbf{x}$.

(iii) Determine $\mathbf{H}$ by scaling Σ with

$$s = h \left(\prod_{l=1}^{d} \lambda_l \right)^{-\frac{1}{d}} \qquad (9)$$

where $\lambda_1, \cdots, \lambda_d$ are the eigenvalues of Σ.

Step (i) determines the local neighborhood of $\mathbf{x}$ with the k nearest neighbors algorithm (k-NN) with Euclidean distance. We use this approach rather than a fixed-radius neighborhood to ensure that, independent of the sparsity of the data, the kernel shape is always based on a reasonable number of data points. We follow Silverman's [7] recommendation of choosing $k = \sqrt{N}$. To ensure that Σ is nonsingular we also need $k > d$, therefore

$$k = \max\left(\left\lfloor \sqrt{N} \right\rceil, d\right) + 1.$$

Step (ii) determines the basic shape of the bandwidth covariance matrix.

Step (iii) ensures that the kernels used in the density estimation of different patterns have the same domain, modulo the local bandwidths. Equation (9) scales the bandwidth matrix in such a way that the volume of the ellipsoid defined by the eigenvectors and values of $\mathbf{H}$ is equal to that of the eigenellipsoid of the bandwidth matrix that is implicitly used in Equation (1).

3 Experiment

We compare the performance of the shape-adaptive method with that of the Modified Breiman Estimator. Section 3.1 introduces the datasets used in our experiments. The metrics used to compare the performance of the different estimators are introduced in Section 3.2.

3.1 Datasets

The performance of the estimators is investigated with two groups of datasets: a number of simulated datasets with known density fields and a real world dataset with an unknown density field. The simulated datasets allow us to test how well a method can recover simple density distributions. Whereas the real world dataset makes it possible to compare the performance of the estimators in the context in which they would be applied.

Simulated Datasets

The simulated datasets are a superset of the sets used by Ferdosi et al. [3]. Figure 1 shows scatter plots of these sets, their definition is given in Table 1.

Dataset one through three, shown in Figures 1a to 1c, are taken from Ferdosi et al. [3]. They consist of a number of spherical Gaussian distributions with random noise added. The means of the Gaussian distribution are chosen in such a way that it is unlikely that the distributions overlap.

Figures 1f to 1h present dataset four through six. These datasets are created from one through three in such a way that the volumes of the eigenellipsoids defined by the covariance matrices of the components of these datasets are equal to volumes of the eigenspheres of the covariance matrix of the component in dataset one through three that inspired them. Furthermore if a is the eigenvalue of the original covariance matrix, the eigenvalues of the covariance matrix in dataset four through six are a^2, $\sqrt{a}$ and $\sqrt{a}$. Consequently the minor axes of these datasets have the same length.

In dataset seven and eight, illustrated in Figures 1h to 1d the semi axes of the ellipsoids all have a different length. The largest minor axis of the Trivariate Gaussian in dataset seven is a factor two larger than the smallest minor axis in that dataset. Whereas in dataset eight the largest minor axis is exponentially larger than the smallest minor axis.

Figure 1i illustrates dataset nine. This set consists of a horizontal wall-like structure and a vertical filament-like structure.

The tenth dataset, shown in Figure 1j, contains three intersecting walls. For each point in these walls its position in two of the three dimensions is drawn from a uniform distribution, the third coordinate is sampled from a Gaussian distribution.

We expect comparable performance from all estimators on dataset one through three, as other than the randomly sampled noise these sets only contain data sampled from a Gaussian distribution with a diagonal covariance matrix. Which results in an equal spread of the data in all dimensions for the non-noise data. Given the elongated shape of the non-noise components in four through eight we expect the shape-adaptive estimator to outperform the MBE. Dataset nine and ten are clearly spread more in one dimension than in other dimensions, thus we expect the shape adaptive estimator to outperform the MBE estimator.

The increasing complexity of these datasets allows us to investigate the performance of the classifier on simple situations, one cluster of data with some noise, to complex density fields that approximate real world data. The advantage of using simulated data is that the true densities of the data points are known, which allows us to test how well the different methods estimate the densities.

Real World Datasets

Iets over real world data

3.2 Error Measures

To quantify the performance of the estimators on the simulated datasets we use the Mean Squared Error (MSE):

$$\mathrm{MSE}\left(\hat{f}(\bullet)\right) = \frac{1}{N}\sum_{j=1}^{N}\left(\hat{f}(\mathbf{x}_j) - f(\mathbf{x}_j)\right)^2.$$

Iets over het quantificeren van de resultaten op real world data.

4 Results

Introduction

4.1 Simulated Datasets

Introduction

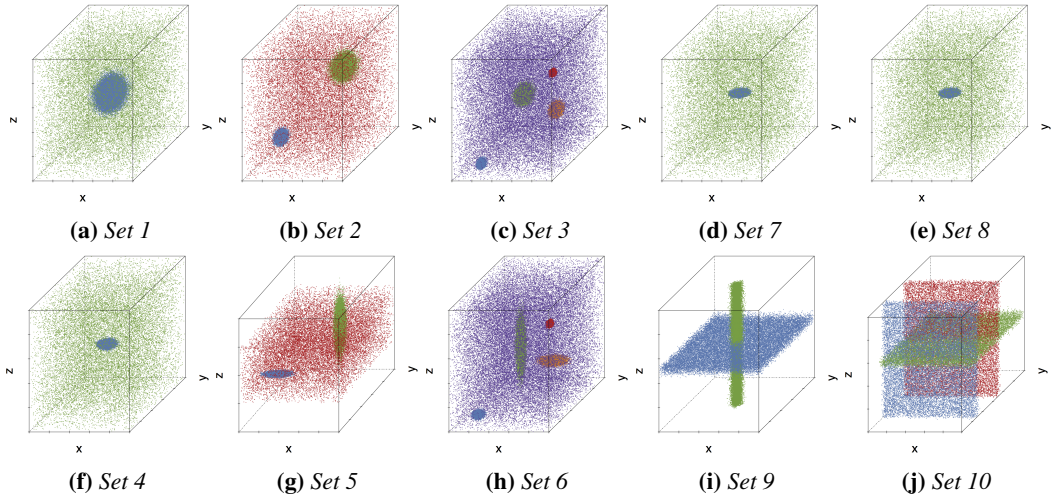
4.2 Real World Datasets

Something

Set		Component	Number	Distribution
1	●	Trivariate Gaussian	4.0×10^4	$(x,y,z) \sim \mathcal{N}([50,50,50], \mathrm{diag}(30))$
	●	Uniform random background	2.0×10^4	$(x,y,z) \sim \mathcal{U}([0,0,0],[100,100,100])$
2	●	Trivariate Gaussian 1	2.0×10^4	$(x,y,z) \sim \mathcal{N}([25,25,25], \mathrm{diag}(5))$
	●	Trivariate Gaussian 2	2.0×10^4	$(x,y,z) \sim \mathcal{N}([65,65,65], \mathrm{diag}(20))$
	●	Uniform random background	2.0×10^4	$(x,y,z) \sim \mathcal{U}([0,0,0],[100,100,100])$
3	●	Trivariate Gaussian 1	2.0×10^4	$(x,y,z) \sim \mathcal{N}([24,10,10], \mathrm{diag}(2))$
	●	Trivariate Gaussian 2	2.0×10^4	$(x,y,z) \sim \mathcal{N}([33,70,40], \mathrm{diag}(10))$
	●	Trivariate Gaussian 3	2.0×10^4	$(x,y,z) \sim \mathcal{N}([90,20,80], \mathrm{diag}(1))$
	●	Trivariate Gaussian 4	2.0×10^4	$(x,y,z) \sim \mathcal{N}([60,80,23], \mathrm{diag}(5))$
	●	Uniform random background	4.0×10^4	$(x,y,z) \sim \mathcal{U}([0,0,0],[100,100,100])$
4	●	Trivariate Gaussian	4.0×10^4	$(x,y,z) \sim \mathcal{N}([50,50,50], \mathrm{diag}([9,\sqrt{3},\sqrt{3}]))$
	●	Uniform random background	2.0×10^4	$(x,y,z) \sim \mathcal{U}([0,0,0],[100,100,100])$
5	●	Trivariate Gaussian 1	2.0×10^4	$(x,y,z) \sim \mathcal{N}([25,25,25], \mathrm{diag}([25,\sqrt{5},\sqrt{5}]))$
	●	Trivariate Gaussian 2	2.0×10^4	$(x,y,z) \sim \mathcal{N}([65,65,65], \mathrm{diag}([\sqrt{20},\sqrt{20},400]))$
	●	Uniform random background	2.0×10^4	$(x,y,z) \sim \mathcal{U}([0,0,0],[100,100,100])$
6	●	Trivariate Gaussian 1	2.0×10^4	$(x,y,z) \sim \mathcal{N}([24,10,10], \mathrm{diag}([4,\sqrt{2},\sqrt{2}]))$
	●	Trivariate Gaussian 2	2.0×10^4	$(x,y,z) \sim \mathcal{N}([33,70,40], \mathrm{diag}([\sqrt{10},\sqrt{10},100]))$
	●	Trivariate Gaussian 3	2.0×10^4	$(x,y,z) \sim \mathcal{N}([90,20,80], \mathrm{diag}(1))$
	●	Trivariate Gaussian 4	2.0×10^4	$(x,y,z) \sim \mathcal{N}([60,80,23], \mathrm{diag}([25,\sqrt{5},\sqrt{5}]))$
	●	Uniform random background	4.0×10^4	$(x,y,z) \sim \mathcal{U}([0,0,0],[100,100,100])$
7	●	Trivariate Gaussian	4.0×10^4	$(x,y,z) \sim \mathcal{N}([50,50,50], \mathrm{diag}([9,2*\sqrt{3},{}^1\!/_2\sqrt{3}]))$
	●	Uniform random background	2.0×10^4	$(x,y,z) \sim \mathcal{U}([0,0,0],[100,100,100])$
8	●	Trivariate Gaussian	4.0×10^4	$(x,y,z) \sim \mathcal{N}([50,50,50], \mathrm{diag}([9,3,1]))$
	●	Uniform random background	2.0×10^4	$(x,y,z) \sim \mathcal{U}([0,0,0],[100,100,100])$
9	●	Wall-like structure	3.0×10^4	$(x,y) \sim \mathcal{U}([0,0],[100,100]),\ (z) \sim \mathcal{N}(50,5)$
	●	Filament-like structure	3.0×10^4	$(x,y) \sim \mathcal{N}([50,50], \mathrm{diag}(5)),\ (z) \sim \mathcal{U}(0,100)$
10	●	Wall-like structure 1	2.0×10^4	$(x,z) \sim \mathcal{U}([0,0],[100,100]),\ (y) \sim \mathcal{N}(10,5)$
	●	Wall-like structure 2	2.0×10^4	$(x,y) \sim \mathcal{U}([0,0],[100,100]),\ (z) \sim \mathcal{N}(50,5)$
	●	Wall-like structure 3	2.0×10^4	$(x,z) \sim \mathcal{U}([0,0],[100,100]),\ (y) \sim \mathcal{N}(50,5)$

Table 1: *The simulated datasets used to test the estimators. The column 'Number' indicates for each component of the dataset how many data points are sampled from that component.* $\mathcal{N}(\mu,\Sigma)$ *denotes a Gaussian distribution with mean* μ *and covariance matrix* Σ*. A diagonal matrix with the values* $x_1, \cdots, x_d$ *on the diagonal is represented as* $\mathrm{diag}([x_1, \cdots, x_d])$*, a scalar matrix with* x *on the diagonal is shown as* $\mathrm{diag}(x)$*.* $\mathcal{U}(a,b)$ *denotes a uniform distribution with its minimum and maximum set to* a *and* b*, respectively. The colors shown in the second column correspond with the colors used for these components of the data set throughout the paper.*

Before Final Version: Remove ticks and labels.



(a) *Set 1* **(b)** *Set 2* **(c)** *Set 3* **(d)** *Set 7* **(e)** *Set 8*

(f) *Set 4* **(g)** *Set 5* **(h)** *Set 6* **(i)** *Set 9* **(j)** *Set 10*

Figure 1: *Scatter plot representation of the simulated datasets defined in Table 1.*

5 Discussion

6 Conclusion

References

[1] L. Breiman, W. Meisel, and E. Purcell. "Variable Kernel Estimates of Multivariate Densities". In: *Technometrics* 19.2 (1977), pp. 135–144.

[2] V.A. Epanechnikov. "Non-Parametric Estimation of a Multivariate Probability Density". In: *Theory of Probability & Its Applications* 14.1 (1969), pp. 153–158.

[3] B.J. Ferdosi et al. "Comparison of Density Estimation Methods for Astronomical Datasets". In: *Astronomy & Astrophysics* 531 (2011).

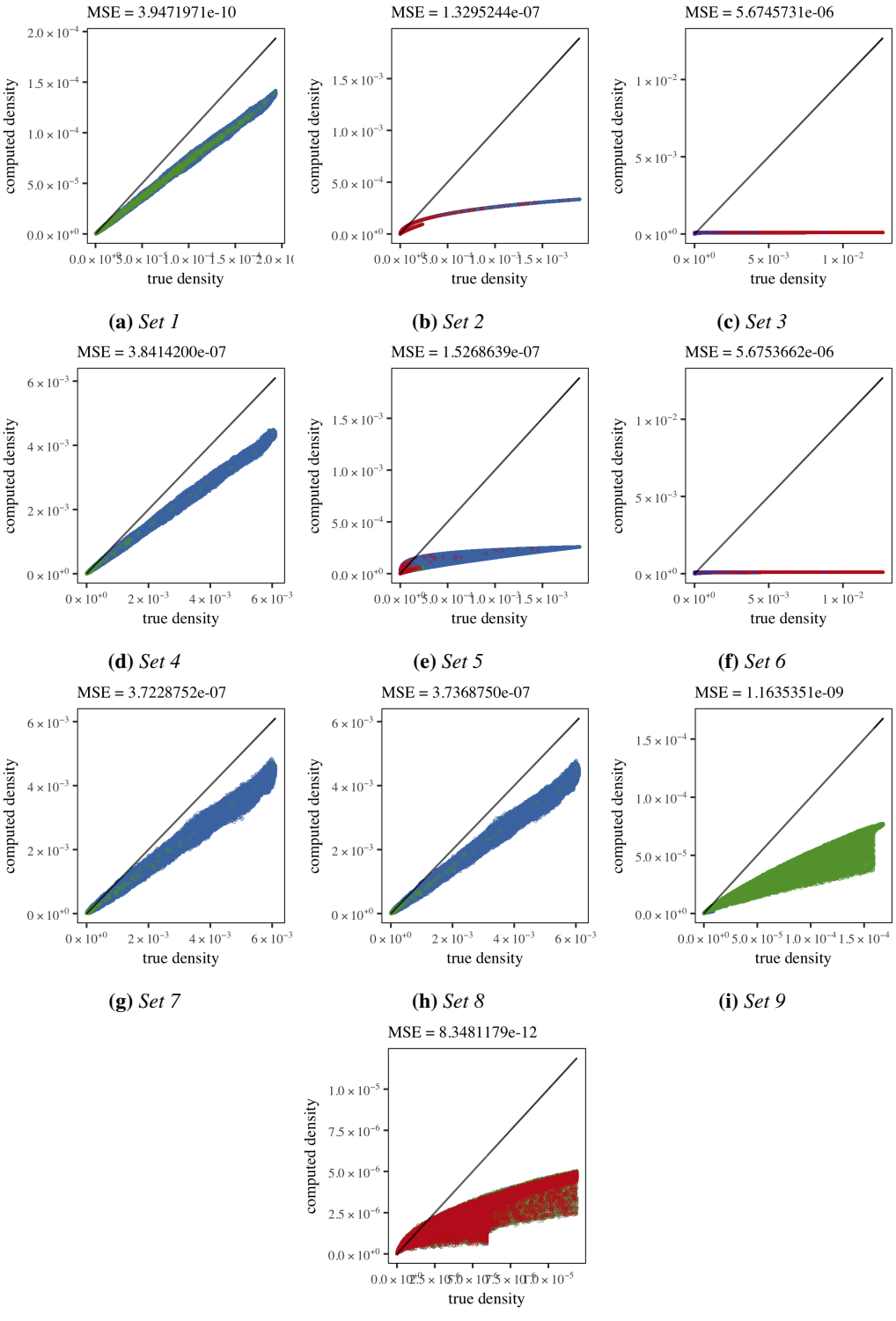
[4] Kirsty J Lees, Andrew J Guerin, and Elizabeth A Masden. "Using kernel density estimation to explore habitat use by seabirds at a marine renewable wave energy test facility". In: *Marine Policy* 63 (2016), pp. 35–44.

[5] Johanna Skarpman Munter and Jens Sjőlund. "Dose-volume histogram prediction using density estimation". In: *Physics in Medicine and Biology* 60.17 (2015), p. 6923.

[6] E. Parzen. "On Estimation of a Probability Density Function and Mode". In: *The Annals of Mathematical Statistics* 33.3 (1962), pp. 1065–1076.

[7] B.W. Silverman. *Density Estimation for Statistics and Data Analysis*. Monographs on Statistics and Applied Probablity. Springer-Science+Business Media, B.V., 1986.

[8] M.H.F. Wilkinson and B.C. Meijer. "DATAPLOT: A Graphical Display Package for Bacterial Morphometry and Fluorimetry Data". In: *Computer Methods and Programs in Biomedicine* 47.1 (1995), pp. 35–49.



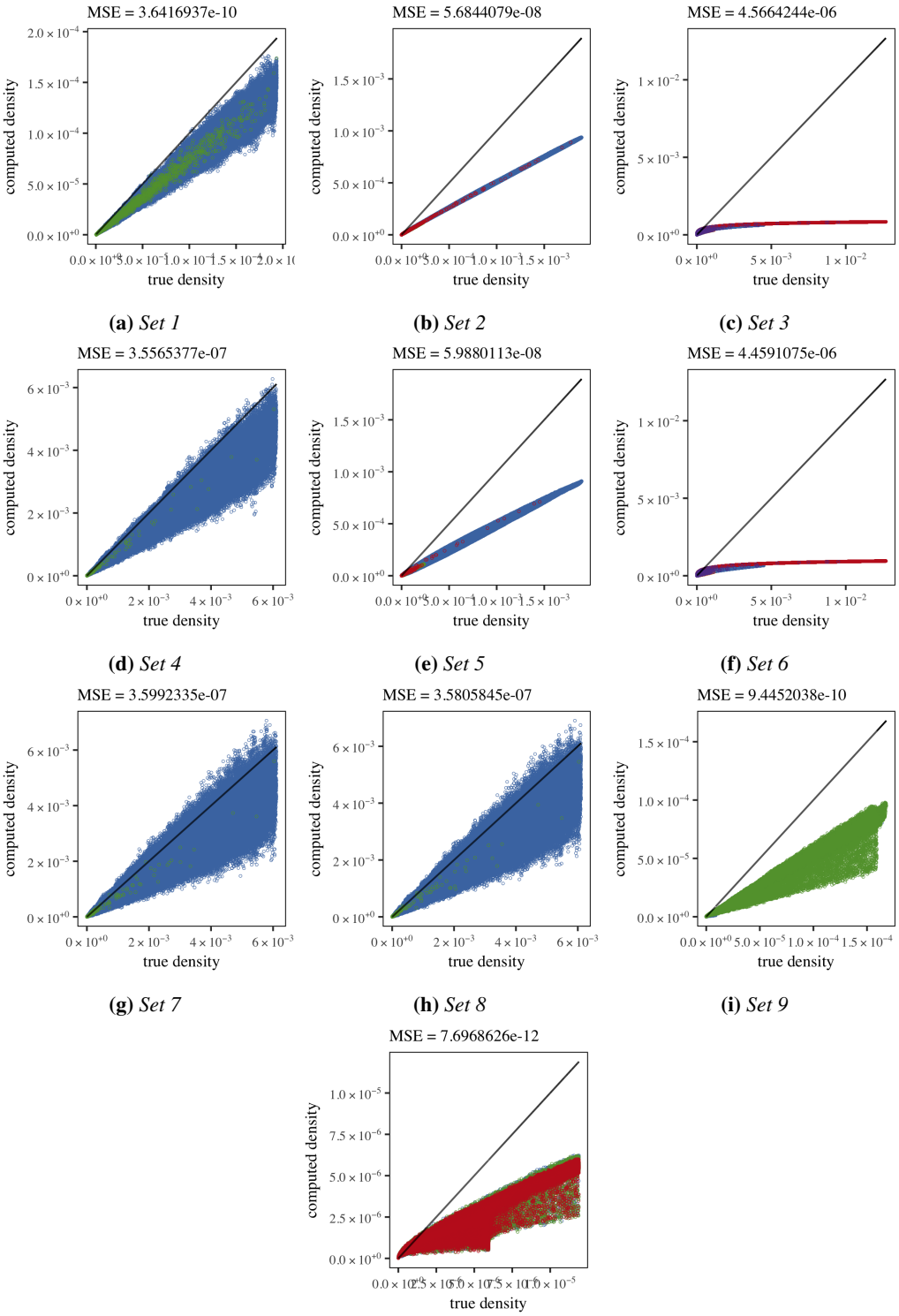
(a) *Set 1* **(b)** *Set 2* **(c)** *Set 3*

(d) *Set 4* **(e)** *Set 5* **(f)** *Set 6*

(g) *Set 7* **(h)** *Set 8* **(i)** *Set 9*

(j) *Set 10*

Figure 2: *Parzen results*



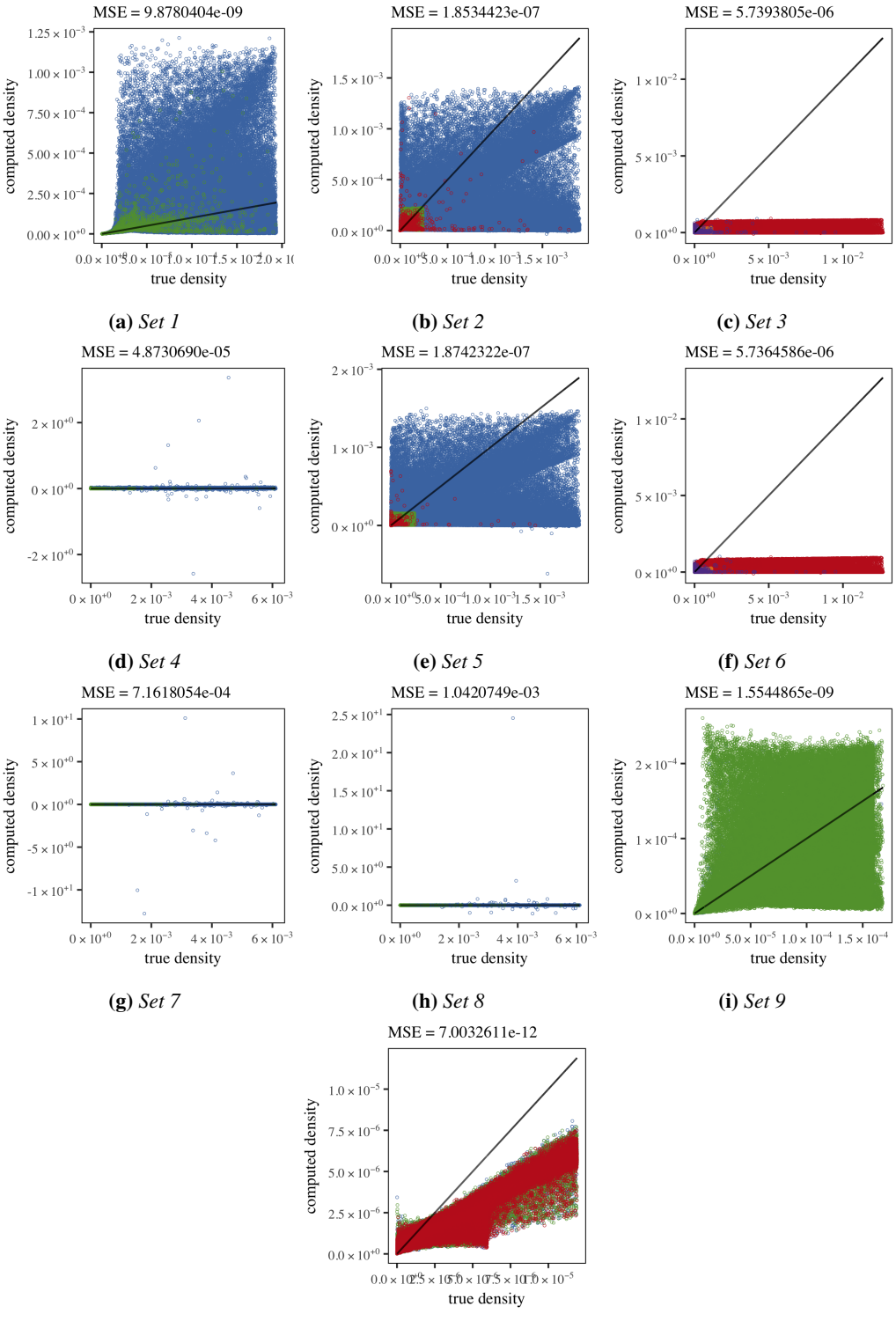
(a) *Set 1* **(b)** *Set 2* **(c)** *Set 3*

(d) *Set 4* **(e)** *Set 5* **(f)** *Set 6*

(g) *Set 7* **(h)** *Set 8* **(i)** *Set 9*

(j) *Set 10*

Figure 3: *MBE results*



(a) *Set 1* **(b)** *Set 2* **(c)** *Set 3*

(d) *Set 4* **(e)** *Set 5* **(f)** *Set 6*

(g) *Set 7* **(h)** *Set 8* **(i)** *Set 9*

(j) *Set 10*

Figure 4: *SAMBE results*