Computational Statistics
https://doi.org/10.1007/s00180-024-01543-9

SOFTWARE ARTICLE

Statistical visualisation of tidy and geospatial data in R via kernel smoothing methods in the eks package

Tarn Duong[1]

Received: 25 June 2024 / Accepted: 21 August 2024

Abstract
Kernel smoothers are essential tools for data analysis due to their ability to convey complex statistical information with concise graphical visualisations. Their inclusion in the base distribution and in the many user-contributed add-on packages of the R statistical analysis environment caters well to many practitioners. Though there remain some important gaps for specialised data, most notably for tidy and geospatial data. The proposed `eks` package fills in these gaps. In addition to kernel density estimation, this package also caters for more complex data analysis situations, such as density derivative estimation, density-based classification (supervised learning) and mean shift clustering (unsupervised learning). We illustrate with experimental data how to obtain and to interpret the statistical visualisations for these kernel smoothing methods.

Keywords Classification · Clustering · ggplot2 · GIS · Kernel density estimation · sf · Tidyverse

1 Introduction

Kernel smoothers form an essential suite of statistical techniques for data analysis in the 21st century due to their ability to convey complex statistical information in a concise and intuitive visual format. This ability arises from their shared characteristic of transforming data samples into smoothed estimates. Kernel smoothers have provided insight in data analysis problems in many situations. A small recent selection of these includes: the identification of important biomedical functions, such as characterising different sub-cellular structures in single cells (Schauer et al. 2010) or characterising a single cell population in mixed cell samples (Chacón et al. 2011); the evaluation of predicted extreme temperatures to calibrate climate models (Béranger et al. 2019); the estimation of the home range of animal movements

✉ Tarn Duong
tarn.duong@gmail.com

[1] Paris, France

Published online: 14 September 2024

(Baíllo and Chacón 2021); or the detection of traffic anomalies from traffic flows (Kalair and Connaughton 2021).

A major access point to kernel smoothers in the R statistical programming environment is the `ks` ('kernel smoothing') add-on package (Duong 2007), which implements density estimation, density derivative estimation, classification (unsupervised learning), clustering (unsupervised learning), and inferential methods. This package utilises the base R graphics engine to generate its statistical graphics. Whilst it remains the most comprehensive graphics engine in R, the `ggplot2` graphics engine (Wickham 2016) has gained popularity, as part of the 'tidyverse', especially with data analysis practitioners. Despite the dramatic rise in the number of analysis methods available in the tidyverse, nonetheless it comprises a limited range of natively implemented kernel smoothers. The first goal of the `eks` ('extended kernel smoothing') package (Duong 2023) is to provide access to a comprehensive suite of kernel smoothers in the tidyverse.

There is an analogous lack of kernel smoothers for geospatial data analysis. Since the term 'geospatial' data analysis refers to many different yet overlapping concepts, we employ it in this paper to refer to data analysis which is compatible with 'Geographical Information Systems' (GIS). Within R, the `sf` package (Pebesma 2018) provides geospatial/GIS functionality via its robust implementation of the 'simple features' GIS standard data format (OGC 2010). The `eks` package relies on this simple features implementation, in order to facilitate visualisations in both `ggplot2` and base R graphical engines, and input/output to external GIS software (such as ArcGIS and QGIS). The second goal of the `eks` package is to provide access to a comprehensive suite of kernel smoothers for geospatial analysis.

Thus a wide range of kernel smoothers is now available for tidy and geospatial data, and for `ggplot2` and base R graphical visualisations. The user is able to select and combine these components, with their differing strengths and applicabilities, in order to construct suitable data analysis workflows. To illustrate kernel smoothers, we employ the tidy data set `air` from the `ks` package, and the geospatial data set `grevilleasf` from the `eks` package, as shown in Fig. 1.

A tidy data set is a data matrix where (i) each variable forms a column and (ii) each observation forms a row, and it is also known as a 'long' data set (Wickham 2014). The first three records of the `air` data set are

```
R> data(air, package="ks")
R> air
        date  time no no2 pm10 co2 temp humi
1 2013-01-01 01:00  6  31  182 776 21.5 46.2
2 2013-01-01 02:00  6  30  166 800 21.6 47.3
3 2013-01-01 03:00  4  27  124 799 21.8 47.0
...
```

These are the hourly mean air quality measurements from 01 January 2013 to 31 December 2016 in the Châtelet underground train station, which is a major hub in the Paris transport network (RATP 2016). We focus on the concentrations of carbon dioxide CO_2 (g/m^3) (`co2`) and of particulate matter less than 10 μm in diameter PM_{10}

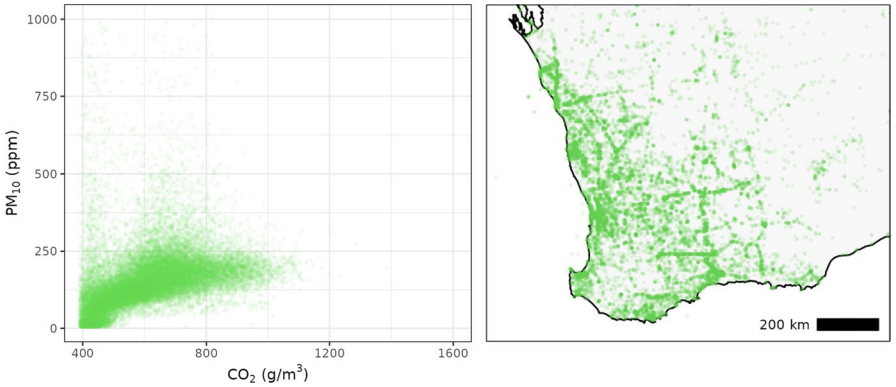

Fig. 1 Scatter plots of data sets. (Left) Tidy air quality measurements `air` (n = 30,239), in the Châtelet underground train station in Paris, France. (Right) Geospatial *Grevillea* locations `grevilleasf` (n = 22,203), in the biodiversity hotspot of south-western Western Australia

(parts per million) (`pm10`), and the hourly interval (`time`). The concentrations of CO_2 indicate the renewal rate of fresh air, and of PM_{10} the potential to affect adversely respiratory health. There are n = 30,239 complete (`co2`, `pm10`) measurements.

The geospatial data `grevilleasf` data set consists of 22,203 plants from 238 different *Grevillea* species collected in Western Australia. The south-west corner of Western Australia is one of the 25 'biodiversity hotspots' which are 'areas featuring exceptional concentrations of endemic species and experiencing exceptional loss of habitat' identified in Myers et al. (2000) to assist in formulating priorities in biodiversity conservation policies. The geodetic coordinates (degrees) of the *Grevillea* locations are transformed into planar coordinates (metres) using the GDA2020/MGA zone 50 (EPSG:7850) projection. They are encoded as a simple feature in the `geometry` column, which is a special data structure that cannot be treated like the usual floating point variables in data frames or tibbles, and require specialised methods implemented by the `sf` package. So we refer to `grevilleasf` solely as a geospatial data set, and omit any mention of its tidy status, to emphasise its distinct geospatial characteristics.

```
R> data(grevilleasf, package="eks")
R> grevilleasf
Simple feature collection with 22303 features and 2 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 73519.97 ymin: 6120859 xmax: 1795868 ymax: 8451928
Projected CRS: GDA2020 / MGA zone 50
                  name   species                 geometry
1    Grevillea robusta   robusta POINT (390106.5 6462671)
2   Grevillea speciosa  speciosa POINT (382689.2 6457387)
3    Grevillea robusta   robusta POINT (390089.8 6462603)
...
```

This paper focuses on the software implementation of the kernel smoothers, and is complementary to Chacón and Duong (2018) which focuses on the underlying statistical framework. The `eks` package computes kernel smoothers for 1- and 2-dimensional tidy data, and 2-dimensional geospatial data. In Sect. 2, we explore kernel density estimation, in Sect. 3 classification (supervised learning), in Sect. 4 density gradient estimation, and in Sect. 5 clustering (unsupervised learning). We illustrate each case first for tidy data with `ggplot2` graphics, and then for geospatial data with `ggplot2` and base R graphics. In Sect. 6, we briefly mention kernel smoothers in other data analysis settings, which are implemented in the `eks` package but have been omitted for brevity, and we end with some concluding remarks.

2 Density estimation

Density estimation is a fundamental statistical analysis tool, since it supplies much information about the data set at hand. Our data $\mathbf{X}_1, \ldots, \mathbf{X}_n$ is a random sample drawn from the common density function f. The goal of density estimation, as its name suggests, is to estimate this unknown density. Kernel density estimates are a popular choice among the many available smoothed density estimation methods, since they possess an intuitive construction. It is the most widely used kernel smoother, and can be considered to be a smoothed version of the histogram. For an arbitrary estimation point $\boldsymbol{x}$, the kernel density estimate is

$$\hat{f}_{\mathbf{H}}(\boldsymbol{x}) = n^{-1} \sum_{i=1}^{n} K_{\mathbf{H}}(\boldsymbol{x} - \mathbf{X}_i). \tag{1}$$

Throughout the `eks` package, the kernel function is the Gaussian density function $K_{\mathbf{H}}(\boldsymbol{x}) = (2\pi)^{-1}|\mathbf{H}|^{-1/2} \exp\left(-\frac{1}{2}\boldsymbol{x}^{\top}\mathbf{H}^{-1}\boldsymbol{x}\right)$. Equation (1) tells us that to compute a kernel density estimate, we place a Gaussian function, with variance $\mathbf{H}$, at each data point $\mathbf{X}_i$, and then we sum these kernel functions. This way, the data sample $\mathbf{X}_1, \ldots, \mathbf{X}_n$ are transformed into a smooth surface $\hat{f}_{\mathbf{H}}$. Chacón and Duong (2018, Chap. 2) contains a more detailed overview of kernel density estimates.

The bandwidth matrix $\mathbf{H}$ in Eq. (1) is the crucial tuning parameter. A bandwidth matrix which is too small leads to an undersmoothed density estimate since it does not offer sufficient reduction in the complexity of the observed data. On the other hand, a bandwidth matrix which is too large leads to an oversmoothed density estimate that obscures important details in the observed data. Thus it is critical to find an optimal trade-off between this under- and oversmoothing. Many possible solutions for optimal smoothing are implemented in the `ks` package, and are thus available in the `eks` package, including the plug-in, unbiased cross validation and smoothed cross validation bandwidths. These selectors are implemented solely for the Gaussian kernel since it "allows for important mathematical and computational simplifications and avoids any possible problems with the non-existence of higher order derivatives of the kernel function when computing data-based bandwidth selectors" (Chacón and Duong 2018, p. 82).

2.1 Tidy density estimation

To illustrate density estimation for tidy data, we focus on a single hourly interval of the air quality measurements. Figure 2 compares the density estimates, for the $n = 1285$ measurements from 11:00 to 12:00, with an optimal bandwidth and a sub-optimal one. The optimal bandwidth is computed from the `eks` package and the sub-optimal one from the `ggalt` package (Rudis et al. 2017). The former, known as the bivariate plug-in bandwidth matrix (Duong and Hazelton 2003), is the default optimal bandwidth in the `eks` package, and it is obtained from a call to `ks::Hpi`. This optimality is the result from theoretical and numerical comparisons in Chacón and Duong (2018, Sect. 2.3) and the references therein. For the air quality measurements, the optimal `ks::Hpi` matrix is [342.1, 97.2; 97.2 365.2]. The presence of non-zero off-diagonal entries in the optimal matrix appropriately orients the kernel functions, and the resulting density estimate is unimodal, as shown in the centre panel of Fig. 2. The default bandwidth in `ggalt`, which is widely used in the tidyverse, is obtained from the element-wise application of the univariate plug-in bandwidth `KernSmooth::dpik`. For the air quality measurements, this bandwidth is [218.6, 0; 0, 124.9]. Since this sub-optimal matrix only applies smoothing in the coordinate axis directions, it yields an undersmoothed density estimate with spurious bimodal structure on the right panel.

In Fig. 2, the heights of the contour regions are calculated according to the probability contours method (Bowman and Foster 1993; Hyndman 1996). The pink region is the smallest region that contains 25% of the probability mass, the orange region plus the enclosed pink region is the smallest region that contains 50% of the probability mass, and the yellow region plus the enclosed orange and pink regions is the smallest region that contains 75% of the probability mass. Since these are relative heights, they facilitate the choice of the contour levels, since it involves selecting values from 0 to 100%, rather than from the range of the density values. These probability contours can also be considered as a multivariate extension of the univariate percentiles, e.g., the 50% contour region is a bivariate equivalent to the median. Due to their intuitive properties, these probability contours are employed throughout in

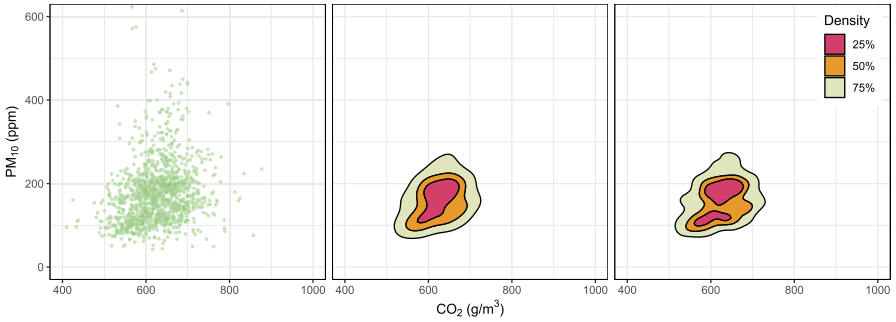

Fig. 2 Filled contour plots of density estimates for air quality measurements 11:00–12:00 ($n = 1285$) with quartile probability contour levels. (Left) Scatter plot. (Centre) Optimally smoothed. (Right) Undersmoothed

`eks`, with the quartile contour levels (25, 50, 75%) being the default values. In addition to their intuitive interpretation, these probability contours are straightforward to compute: the kernel density estimate is evaluated at the n observed data values $\hat{f}_{\mathbf{H}}(\mathbf{X}_1), \ldots, \hat{f}_{\mathbf{H}}(\mathbf{X}_n)$, then we compute τ_α as the α-quantile of these evaluated values, and the α probability contour region is the level set of the density estimate at τ_α, i.e. $\{\boldsymbol{x} : \hat{f}_{\mathbf{H}}(\boldsymbol{x}) > \tau_\alpha\}$ (Hyndman 1996). These probability contours are also implemented in the `ggdensity` package (Otto and Kahle 2023), though with a similar sub-optimal bandwidths as in `ggalt`.

The R code snippets included here are intended to give an overall idea of the syntax of the `eks` package, rather than a complete code to reproduce the figures. The latter is provided in the companion R script. The code snippet to compute the density estimate with the optimal bandwidth, in the centre panel in Fig. 2, is

```
R> ## tidy density estimate
R> air2 <- ungroup(filter(air, time=="11:00"))
R> air2 <- select(air2, co2, pm10)
R> t1 <- tidy_kde(air2, H=H1)
R> ggplot(t1, aes(x=co2, y=pm10)) + geom_contour_filled_ks(colour=1)
```

The function `tidy_kde` is a wrapper function for `ks::kde`, which computes the tidy density estimate explicitly. This differs from existing layer functions, e.g., `ggplot2::geom_density_2d` and `ggalt::geom_bkde2d`, which compute the density estimate internally and do not return a user-level R object. The tidy density estimate from `tidy_kde` is:

```
R> t1
# A tibble: 22,801 x 6
     co2  pm10  estimate ks         tks   label
   <dbl> <dbl>     <dbl> <list>     <chr> <chr>
 1  342. -28.8 -5.65e-24 <kde>      kde   Density
 2  346. -28.8 -7.42e-22 <int [1]> kde   Density
 3  350. -28.8  2.38e-22 <int [1]> kde   Density
...
```

This output is a tidy tibble with an added `tidy_ks` class, which allows for a `ggplot.tidy_ks` method to be defined for this object class. Otherwise, it can be treated as a tibble. The first two columns `co2, pm10` (same names as the input data) are the coordinates of the vertices in the estimation grid, the third column `estimate` is the density estimate value at `co2, pm10`. The fourth column `ks` holds the output from `ks::kde`. This is required for the computation of probability contours in the new layer function `geom_contour_filled_ks` to draw the filled contour plots for `tidy_ks` objects. The remaining columns indicate that the output is a density estimate computed from `ks::kde`, and they are employed in `ggplot.tidy_ks` to create default aesthetic mapping and legend labels. This default aesthetic mapping is `ggplot2::aes(x=co2, y=pm10, z=estimate, weight=ks)`. Whilst the

`x`, `y`, `z` aesthetics are as expected for a bivariate contour plot, the `weight` aesthetic is unorthodox, since it is not a weighting variable: it is a workaround in `ggplot2` graphics to mimic the dynamic display of probability contours in base R graphics.

For the air quality measurements from 11:00 to 12:00, the quartile contour levels for the optimally smoothed density estimate in the centre panel in Fig. 2 are 3.31e−5, 2.42e−5, 1.22e−5, and for the undersmoothed density estimate in the right panel are 3.51e−5, 1.22e−5, 2.53e−5. These probability contour heights are different for each different density estimate, even if the target contour probabilities remain the same at 25, 50, 75%. On the other hand, it is sometimes useful to have a set of fixed contour heights for all density estimates for a direct comparison. A heuristic method consists of computing the probability contour heights for each density estimate, for a fixed set of probabilities, which are then aggregated. We compute the corresponding probabilities for each density estimate for this aggregated set of contour heights, and we remove any contour levels whose estimated probability which are too close to each other. This procedure is implemented in the `contour_breaks` function, though some trial and error is still likely required to produce visually appealing contour plots for all density estimates (Chacón and Duong 2018, Sect. 2.2).

We revisit the density estimates for the air quality measurements from 11:00 to 12:00, this time with the fixed contour heights (3.90e−6, 1.46e−5, 2.48e−5, 3.25e−5, 4.15e−5) in Fig. 3. With these fixed contour heights, a direct comparison of different density estimates is possible. The optimally smoothed estimate on the right exceeds the two highest contour heights (4.15e−5 dark pink, 3.25e−5 dark orange) with a unimodal bump, whereas the density estimate on right exceeds them with a bimodal structure. Since the latter adds some spurious modal information, it is considered to be undersmoothed in comparison.

The code to produce two density estimates with a single set of contour heights in Fig. 3 is

```
R> ## fixed contour levels
R> H2 <- diag(sapply(air2, KernSmooth::dpik)^2)
R> t2 <- tidy_kde(air2, H=H2)
R> t3 <- c(t1, t2)
R> b <- contour_breaks(t3, cont=c(10,30,50,70,90))
R> ggplot(t3, aes(x=co2, y=pm10)) + geom_contour_filled_ks(colour=1, breaks=b)
+  facet_wrap(~group)
```

2.2 Geospatial density estimation

To illustrate density estimation for geospatial data, we utilise single species subsets of the *Grevillea* locations. Figure 4 compares the density estimates for the $n = 93$ locations of the *G. yorkrakinensis* species which result from an optimal plug-in bandwidth [8.84e8, −8.33e8; −8.33e8, 1.36e9] and a sub-optimal bandwidth [5.43e8, 0; 0, 9.10e8]. The optimally smoothed density estimate in the centre panel in Fig. 4 displays a trimodal structure with obliquely oriented contours. The oversmoothed density estimate in the right panel, whilst it also has trimodal structure, it

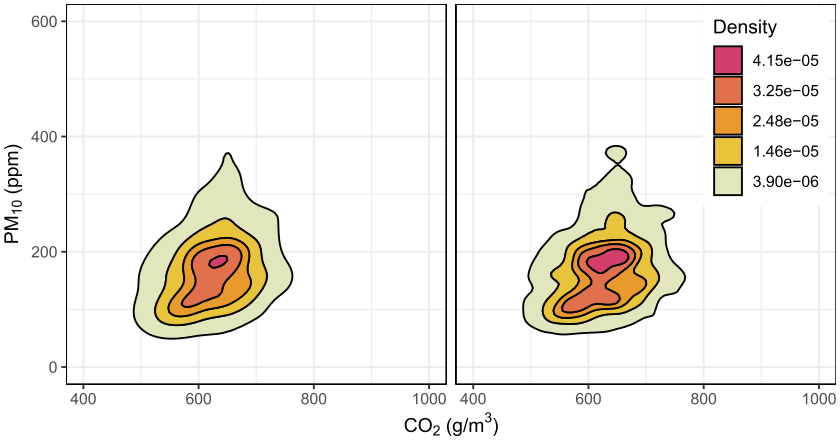

Fig. 3 Filled contour plots of density estimates for air quality measurements from 11:00 to 12:00 ($n = 1285$) with fixed contour levels. (Left) Optimally smoothed. (Right) Undersmoothed

has circular contours which do not follow closely the orientation of the observed data points, and so is considered to be oversmoothed.

To produce the centre panel in Fig. 4 for the *G. yorkrakinensis* locations, the commands are

```
R> ## geospatial density estimate
R> yorkr <- filter(grevilleasf, species=="yorkrakinensis")
R> s1 <- st_kde(yorkr)
```

The function `st_kde` is the geospatial equivalent of `tidy_kde`, and produces an object of class `sf_ks`, which is a list of 3 fields: `tidy_ks`, `grid`, and `sf`. The first field is a summary of the tidy density estimate from `tidy_kde`, the second are the rectangular polygons of the estimation grid, and the third are the 1% to 99% probability contour regions of the density estimate. We focus on the contour regions.

```
R> s1$sf
Simple feature collection with 99 features and 2 fields
Geometry type: GEOMETRY
Dimension:     XY
Bounding box:  xmin: 429181.6 ymin: 6333015 xmax: 757134 ymax: 6793115
Projected CRS: GDA2020 / MGA zone 50
   contlabel     estimate                          geometry
1         99 2.491961e-12 POLYGON ((430965 6755235, 4...
2         98 3.699348e-12 POLYGON ((437905.1 6746871,...
3         97 6.714617e-12 POLYGON ((448315.2 6740434,...
...
```

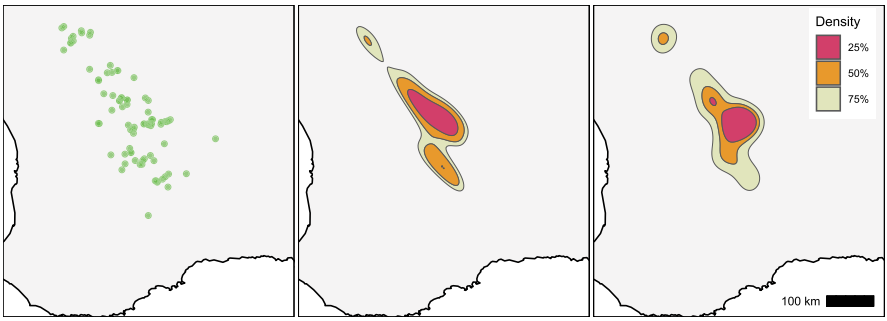

Fig. 4 Filled contour plots of density estimates for *G. yorkrakinensis* ($n = 93$) with quartile probability contour levels. (Left) Scatter plot. (Centre) Optimally smoothed. (Right) Oversmoothed

This has 2 attributes: `contlabel` (label of probability contour region) and `estimate` (height of probability contour region). Unlike for `tidy_kde` where the probability contour regions are computed dynamically in the layer function `geom_contour_filled_ks`, these 1–99% regions are converted to (multi)polygons prior to plotting since the dynamic conversion during plotting could be computationally heavy. Since we are unable replicate the automatic selection of the quartile contours 25, 50, 75% by default, like in `geom_contour_filled_ks`, for the `ggplot2::geom_sf` layer function, we first apply `st_get_contour` to the input of `ggplot2::geom_sf`. The `sf_ks` class also has a `ggplot.sf_ks` method which computes the default map legend.

```
R> ## geospatial density estimate geom_sf plot
R> ggplot(s1) + geom_sf(data=st_get_contour(s1), aes(fill=contlabel))
```

The following command produces the equivalent in base R graphics to the `ggplot2` plot in the centre panel in Fig. 3.

```
R> ## geospatial density estimate base R plot
R> plot(s1)
```

This `plot.sf_ks` method for `sf_ks` objects method internally calls `st_get_contour` to extract the required contour polygons for plotting, so it is more concise than `ggplot2::geom_sf` that requires an explicit user-level call to `st_get_contour`. The base R and `ggplot2` plots are essentially identical since they comply with the geospatial standard specifications for simple features.

2.3 Optimal bandwidth matrices

Since the bandwidth matrix is the crucial tuning parameter for kernel density estimates, we explore further their statistical properties. These properties are the subject of a vast body of research literature, which we do not attempt to review here, and instead provide a simplified outline of how the optimal bandwidth matrix in `eks` is obtained.

We begin with a squared error discrepancy between a density estimate $\hat{f}_{\mathbf{H}}$ and the target density f, i.e., $M(\mathbf{H}) = \int \mathbb{E}\,[\hat{f}_{\mathbf{H}}(\boldsymbol{x}) - f(\boldsymbol{x})]^2\,\mathrm{d}\boldsymbol{x}$. Since this expression involves the unknown target density f, it must be estimated for it to be of practical use. The plug-in bandwidth matrix in `ks::Hpi` computes the estimate $\hat{M}(\mathbf{H}) = (4\pi)^{-d/2} n^{-1} |\mathbf{H}|^{-1/2} + \frac{1}{4}\hat{\boldsymbol{m}}_4^{\top}(\mathrm{vec}\mathbf{H} \otimes \mathrm{vec}\mathbf{H})$. We omit to describe this estimate rigorously since it would require lengthy technical definitions: the interested reader is encouraged to consult Chacón and Duong (2018, Chap. 3) for details. We are content to state that the first term in $\hat{M}$ is related to the variance of the density estimate, and the second term to the square of the bias of the density estimate. An optimal bandwidth matrix $\hat{\mathbf{H}}$ is defined as

$$\hat{\mathbf{H}} = \operatorname*{argmin}_{\mathbf{H}} \hat{M}(\mathbf{H}) \tag{2}$$

where the minimisation is carried out over the space of all symmetric positive definite matrices. When this minimisation is achieved, then there is an optimal trade-off between the variance and the squared bias, or equivalently between over- and under-smoothing. When an optimal bandwidth matrix $\hat{\mathbf{H}}$ is substituted into Eq. (1), the resulting kernel density estimate is the closest to the target density f as measured by the discrepancy $\hat{M}$. Different bandwidth matrices arise from the different ways of computing $\hat{M}$ and/or from different ways of carrying out the minimisation. For example, the default bandwidth in `ggalt` treats the joint bivariate optimisation in Eq. (2) as two separate univariate optimisation problems. The density estimate functions `tidy_kde` and `st_kde` compute $\hat{\mathbf{H}}$ in Eq. (2) by calling the `ks::Hpi` function, and then substitute this $\hat{\mathbf{H}}$ into Eq. (1), to compute an optimal tidy/geospatial density estimate, as shown in the centre panels in Figs. 2 and 3.

Additional bandwidth matrices in the `ks` package include the normal scale `ks::Hns`, unbiased cross validation `ks::Hucv` and smoothed cross validation `ks::Hscv`. The commands are:

```
R> ## smoothed cross validation selector
R> H3 <- ks::Hscv(air2)
R> t3 <- tidy_kde(air, H=H3)
```

For most data samples, the plug-in bandwidth `ks::Hpi` yields fast and robust kernel estimates, though there remain some cases where other bandwidths are more suitable. For a review of the performance of these bandwidths, see Chacón and

Duong (2018, Chap. 3) and the references therein. For brevity, we illustrate kernel estimates only with the plug-in optimal bandwidths in the sequel.

3 Density-based classification (supervised learning)

The goal of classification is to assign future data to one of the known classes in the current data. So this is a supervised learning problem. The data are $(\mathbf{X}_1, Y_1), \dots, (\mathbf{X}_n, Y_n)$, where the $\mathbf{X}_i$ are the observed attributes, and the Y_i is the known class label from m classes. These data are a random sample from the mixture density $\pi_1 f_1 + \cdots + \pi_m f_m$, where π_j is the prior probability and f_j is the marginal density function for class j, for $j = 1, \dots, m$ (Chacón and Duong 2018, Sect. 7.2).

The Bayes classifier assigns a candidate point $\boldsymbol{x}$ to the class c with the highest density value at $\boldsymbol{x}$, i.e., $c(\boldsymbol{x}) = \operatorname{argmax}_{j=1,\dots,m} \pi_j f_j(\boldsymbol{x})$. This Bayes classifier has few assumptions on the form of the target densities f_j and achieves the smallest misclassification rate (Bayes error) among all classifiers given the attributes (Devroye et al. 1996, p. 2). The misclassification error is the probability that we do not classify a candidate point in class j given that it is drawn from class j, $\mathbb{P}\{c(\mathbf{X}) \neq j | \mathbf{X} \sim f_j\}$. The density-based classifier replaces the prior probability π_j with the observed sample class proportion $\hat{\pi}_j$, and the marginal density f_j with the marginal density estimate $\hat{f}_j$. Each marginal density estimate is computed with its own optimal bandwidth matrix. The estimated class label for $\boldsymbol{x}$ from the kernel density-based classifier is thus

$$\hat{c}(\boldsymbol{x}) = \operatorname*{argmax}_{j=1,\dots,m} \hat{\pi}_j \hat{f}_j(\boldsymbol{x}).$$

This kernel classifier is more adaptable than the usual linear and quadratic classifiers. The linear classifier uses Gaussian density fits with a common variance matrix for all classes, and the quadratic classifier Gaussian density fits with a different variance matrix for each class.

3.1 Tidy classification

Our data sample comprises the air quality measurements for three hourly intervals at six hours apart throughout the day, i.e. 07:00–08:00 (pink circles, $n_1 = 1282$), 13:00–14:00 (green triangles, $n_2 = 1280$), and 19:00–20:00 (blue squares, $n_3 = 1292$), as shown in the scatter plot in the left panel of Fig. 5. In the centre panel are the quartile probability contour plots of marginal density estimates $\hat{\pi}_1 \hat{f}_1$ (pink solid lines), $\hat{\pi}_2 \hat{f}_2$ (green dotted lines), and $\hat{\pi}_3 \hat{f}_3$ (blue dashed lines), where $\hat{f}_1$ is the density estimate for 07:00–08:00, $\hat{f}_2$ for 13:00–14:00, and $\hat{f}_3$ for 19:00–20:00. As the marginal density contours have considerable overlap in the central regions, it is difficult to decide visually which marginal density value is higher. This is resolved in the plot of estimated class labels from the density-based classifier on the right of Fig. 5. The regions where the 07:00–08:00 measurements are more likely are coloured in pink, the 13:00–14:00 measurements are more likely are in green, and the 19:00–20:00 measurements are more likely are in blue. The general trend is, as the

day progresses, both levels of CO_2 and PM_{10} increase, with the increase of PM_{10} being more sustained. The boundaries of these class label regions are complex and would not be well-estimated by the linear and quadratic classifiers. It appears that the upper right corner of the kernel classifier gives noisy decision boundaries but since this region has no observed data, it is has little effect on its accuracy. The misclassification rate of the kernel classifier is 0.38, in comparison to 0.44 for a linear classifier (`MASS::lda`) and 0.43 for a quadratic classifier (`MASS::qda`).

The command to compute a tidy kernel classifier is `tidy_kda`. It requires a grouped tidy tibble as its input (`air_gr`), grouped by the class factor variable (`time`). To produce the marginal densities plot for the density-based classifier in the centre panel in Fig. 5.

```
R> ## tidy density-based classifier contours
R> air <- group_by(air, time)
R> air_gr <- filter(air, time %in% c("07:00", "13:00", "19:00"))
R> air_gr <- mutate(air_gr, time=droplevels(time))
R> t4 <- tidy_kda(air_gr)
R> ggplot(t4, aes(x=co2, y=pm10)) + geom_contour_ks(aes(colour=time))
```

The layer function `geom_contour_ks` draws the contour lines for `tidy_ks` objects. In addition to the columns already present in the density estimate, the extra columns in the output of a density-based classifier relate to the classes: `prior_prob` (class sample proportion), `label` (estimated class label), `time` (same as input class label). The structure of a density-based classifier is similar to that for a density estimate grouped by a class variable.

3.2 Geospatial classification

Our geospatial data sample comprises the combined *G. hakeoides* (pink circles, $n_1 = 207$) and *G. paradoxa* (green triangles, $n_2 = 358$) locations, as shown in the

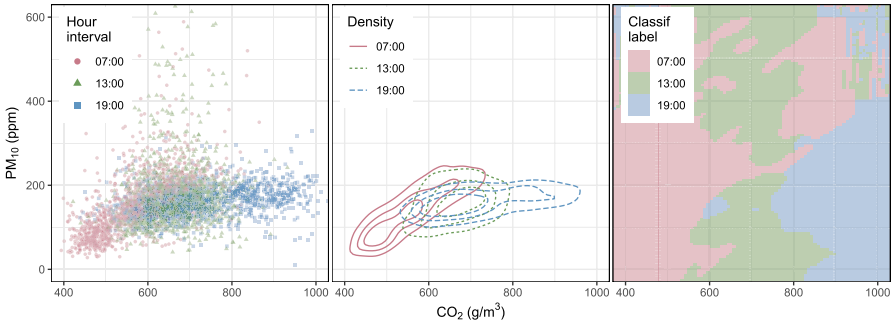

Fig. 5 Density-based classifier for air quality measurements at different hourly intervals. (Left) Scatter plots for 07:00–08:00 ($n_1 = 1282$), 13:00–14:00 ($n_2 = 1280$), 19:00–20:00 ($n_3 = 1292$). (Centre) Quartile probability contours of marginal density estimates. (Right) Class label estimates

scatter plot in the left panel of Fig. 6. In the centre panel are the quartile probability contour plots of marginal density estimates $\hat{\pi}_1\hat{f}_1$ (pink solid lines) and $\hat{\pi}_2\hat{f}_2$ (green dotted lines), where $\hat{f}_1$ is the density estimate for *G. hakeoides*, and $\hat{f}_2$ for *G. paradoxa*. The regions where *G. hakeoides* is more likely are coloured in pink, and where *G. paradoxa* is more likely are in green. For display purposes, the class labels have been truncated to the convex hull of the marginal density estimates so that they remain over the land area (in grey). We observe again that boundaries of these class label regions are complex and would not be well-estimated by the linear and quadratic classifiers.

A geospatial density-based classifier requires the input (`grevilleasf_gr`) to be grouped by the class factor variable (`species`):

```
R> ## geospatial density-based classifier
R> grevilleasf_gr <- filter(grevilleasf, species %in% c("paradoxa","hakeoides"))
R> grevilleasf_gr <- mutate(grevilleasf_gr, species=factor(species))
R> grevilleasf_gr <- group_by(grevilleasf_gr, species)
R> s4 <- st_kda(grevilleasf_gr)
```

The estimated class labels are stored in the `sf_ks` object in the `grid` field as a collection of rectangular polygons. To plot these class labels, as in the right panel in Fig. 6, we call `ggplot2::geom_sf` on the `grid` field for a `ggplot2` plot, and `plot(x, which_geometry=="grid")` for a base R plot.

```
R> ## geospatial density-based classifier geom_sf plot
R> ggplot(s4) + geom_sf(data=s4$grid, aes(fill=label), alpha=0.2, colour=NA)
R> ## base R plot
R> plot(s4, which_geometry="grid", border=NA)
```

The question of optimal bandwidths for a density-based classifier is more complicated than that for a density estimate. We opt for a simple and robust implementation in the `eks` package, where `tidy_kda` and `st_kda` call `ks::Hpi` for each class

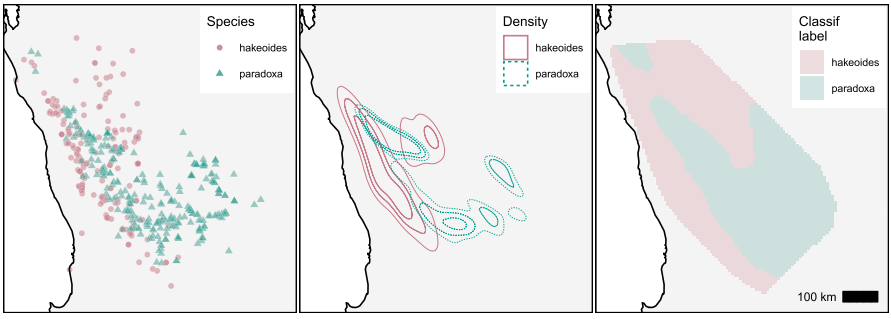

Fig. 6 Density-based classifier for *Grevillea* locations. (Left) Scatter plots for *G. hakeoides* ($n_1 = 207$), *G. paradoxa* ($n_2 = 358$). (Centre) Quartile probability contours of marginal density estimates. (Right) Class label estimates

data sub-sample. These class-wise optimal bandwidths are known to asymptotically minimise the misclassification error. Whilst there is an intuitive appeal in selecting bandwidths to exactly minimise the misclassification error, it is not clear how much is gained in practise with this more complicated approach over the simpler bandwidths. Moreover, there are currently no efficient algorithms to compute these more complicated bandwidths. See Chacón and Duong (2018, Sect. 7.2) for a discussion.

4 Density derivative estimation

Crucial information about the structure of a data set is not always revealed by examining solely the density values, and can only be discerned via the density derivatives. For example, the local minima/maxima of the data density are characterised as the locations where the first derivative is identically zero. A recent example of the utility of density derivative estimates in data analysis is the segmentation of digital images, which utilised the first density derivative of pixel colour-locations to guide the search for similar image segments more efficiently than using only the density of the pixel colour-locations (Beck et al. 2016).

With the same data as for the density estimation case, i.e., $\mathbf{X}_1, \ldots, \mathbf{X}_n$ is a random sample drawn from the common density function f, our goal is to estimate the first (gradient) derivative of the unknown density f. For 2-dimensional data, the gradient of a density function f is comprised of two partial derivatives $\mathsf{D}f = [\partial f/\partial x_1, \partial f/\partial x_2]$. The kernel estimate of the density gradient is given by

$$\mathsf{D}\hat{f}_{\mathbf{H}}(\boldsymbol{x}) = n^{-1} \sum_{i=1}^{n} \mathsf{D}K_{\mathbf{H}}(\boldsymbol{x} - \mathbf{X}_i) \tag{3}$$

where the gradient kernel function is $\mathsf{D}K_{\mathbf{H}}(\boldsymbol{x}) = -(2\pi)^{-1}|\mathbf{H}|^{-1/2}\mathbf{H}^{-1}\boldsymbol{x} \exp\left(-\frac{1}{2}\boldsymbol{x}^{\mathsf{T}}\mathbf{H}^{-1}\boldsymbol{x}\right)$.

4.1 Tidy density derivative estimation

Since there are two components of the density gradient, it can be visualised using two separate plots, one for each partial derivative. A more concise alternative is a quiver plot, in which arrows, whose length and direction are determined by the gradient, are drawn at each point in the estimation grid. The right panel of Fig. 7 is the quiver plot for the density gradient estimate for the air quality measurements from 13:00 to 14:00, superposed on the density estimate. The arrows for the density gradient point towards the peaks of the modal regions. These arrows are longer where the density gradient is steeper, and they are shorter in the density tails where the slope is flatter. These density gradients indicate the rate of change in the data density, which is not easy to ascertain from the density levels themselves in the underlying density contour plot.

The command for a tidy density gradient estimate is `tidy_kdde(x, deriv_order=1)`. The function `tidy_kquiver` converts the output from `tidy_kdde` into a format suitable for the quiver plot layer function `ggquiver::geom_quiver` (O'Hara-Wild 2019). The code to produce a quiver plot superposed on a density estimate is

```
R> ## tidy density gradient estimate
R> air3 <- ungroup(filter(air, time=="13:00"))
R> air3 <- select(air3, co2, pm10)
R> t5 <- tidy_kdde(air3, deriv_order=1)
R> t6 <- tidy_kquiver(t6, thin=9)
R> ggplot(t1, aes(x=co2,y=pm10)) + ggquiver::geom_quiver(data=t6, aes(u=u,v=v))
```

The output from `tidy_kdde` is a tidy tibble which is grouped by `deriv_group`. The columns present in a density estimate are also present in a density derivative estimate, along with some additional columns relating to the derivative: `deriv_order` (derivative order, 1 for the gradient), `deriv_ind` (partial derivative enumeration, from 1 to 2), `deriv_group` (partial derivative indices (1,0), (0,1) which correspond to $\partial/\partial x_1, \partial/\partial x_2$ respectively).

4.2 Geospatial density derivative estimation

For geospatial data, the left panel in Fig. 7 is the quiver plot for the density gradient estimate for *G. yorkrakinensis*, superposed on the density estimate. Whilst with `st_kquiver` we can compute a geospatial output, `ggplot2::geom_sf` is not able plot arrows, and it is not possible to overlay a `ggquiver::geom_quiver` layer

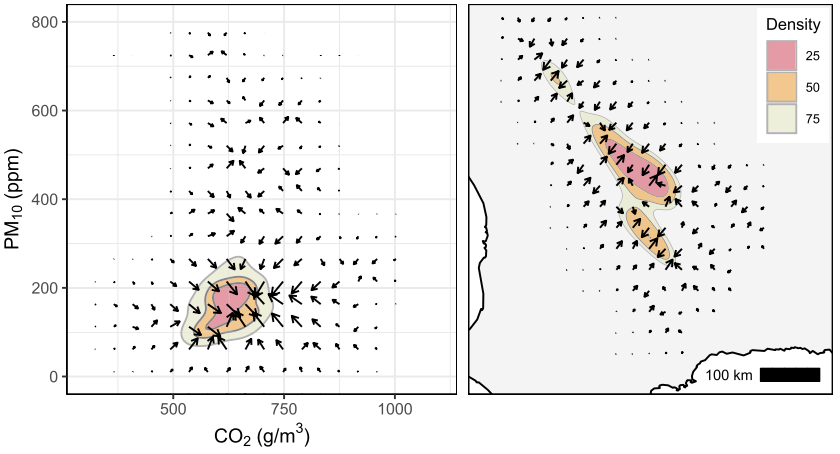

Fig. 7 Quiver plots of density gradient estimate, superposed over density estimates. (Left) Air quality measurements for 13:00–14:00 ($n = 1280$). (Right) *G. yorkrakinensis* locations ($n = 93$)

over a `geom_sf` layer. The current work-around is to overlay a `ggplot2::geom_segment` layer over a `geom_sf` layer, with some trial and error required in `grid::arrow` to produce suitable arrows.

```
R> ## geospatial density gradient estimate geom_sf plot
R> s5 <- st_kdde(yorkr, deriv_order=1)
R> s6 <- st_kquiver(s5, thin=9)
R> ggplot(s1) + geom_segment(data=s6$sf, aes(x=lon, xend=lon_end, y=lat,
+  yend=lat_end), arrow=grid::arrow(length=0.1*s6$sf$len))
```

On the other hand, for a base R plot, the display of geospatial and tidy data are freely interchangeable, so we can overlay the quiver plot `plot(x, display="quiver")` for a kernel density gradient estimate from the `ks` package.

```
R> ## geospatial density gradient estimate base R plot
R> plot(s6$tidy_ks$ks[[1]], display="quiver")
```

For optimal bandwidth selection for kernel density gradient estimates, it is crucial to note that the optimal bandwidth matrix for $\mathsf{D}\hat{f}_{\mathbf{H}}$ is not the same as that for $\hat{f}_{\mathbf{H}}$. For a density estimate the optimality criterion is $M(\mathbf{H}) = \int \mathbb{E}\,[\hat{f}_{\mathbf{H}}(\boldsymbol{x}) - f(\boldsymbol{x})]^2\,\mathrm{d}\boldsymbol{x}$, whereas the criterion for a density gradient estimate is $M_1(\mathbf{H}) = \int \mathbb{E}\,\|\mathsf{D}\hat{f}_{\mathbf{H}}(\boldsymbol{x}) - \mathsf{D}f(\boldsymbol{x})\|^2\,\mathrm{d}\boldsymbol{x}$. Since $M \neq M_1$, their minimisers are also not equal in general. The default optimal bandwidth for the density gradient estimate in the `eks` package is the plug-in bandwidth (Chacón and Duong 2010) obtained from a call to `ks::Hpi(x, deriv.order=1)`. For the air quality measurements for 13:00 to 14:00, this bandwidth matrix is [441.0, 59.5; 59.5, 305.0]. In comparison, the optimal bandwidth matrix for the density estimate is [495.0, 88.0; 88.0, 245.0]. For the *G. yorkrakinensis* data, the optimal plug-in bandwidth matrix for the density gradient estimate is [4.39e8, −4.36e8; −4.36e8, 7.73e8], whereas for the density estimate, it is [8.84e8, −8.33e8; −8.33e8, 1.36e9].

5 Density-based clustering (unsupervised learning)

The goal of clustering is to discover homogeneous groups within a data set in a trade-off between similarity/dissimilarity: members of the same cluster are similar to each other while members of different clusters are dissimilar to each other. If the q unknown population clusters are $\{C_1, \ldots, C_q\}$, then the cluster labelling function is $c(\boldsymbol{x}) = j$ whenever a candidate point $\boldsymbol{x}$ belongs to cluster C_j. Whilst we are able to estimate the cluster labelling function for all candidate points, for the vast majority of data analysis cases, it is sufficient to compute $\hat{c}(\mathbf{X}_1), \ldots, \hat{c}(\mathbf{X}_n)$ for the data sample $\mathbf{X}_1, \ldots, \mathbf{X}_n$. Since the cluster labels are unknown, then this is an unsupervised learning problem.

Many clustering algorithms have been proposed in the literature. Our chosen approach is density-based clustering, where a cluster is a data-rich region (high density values) which is separated from another data-rich region by a data-poor region (low density values). Thus we associate each data point to its 'most representative' data-rich region. In the `eks` package, this association is carried out with a mean shift algorithm (Fukunaga and Hostetler 1975). For a data point $\mathbf{X}_i$, we initialise a sequence with $\mathbf{X}_{i,0} = \mathbf{X}_i$, then we iterate the recurrence equation

$$\mathbf{X}_{i,k+1} = \mathbf{X}_{i,k} + \mathbf{H}^{-1} \mathsf{D}\hat{f}_{\mathbf{H}}(\mathbf{X}_k) \big/ \hat{f}_{\mathbf{H}}(\mathbf{X}_k),$$

where $\hat{f}_{\mathbf{H}}$ is a density estimate and $\mathsf{D}\hat{f}_{\mathbf{H}}$ is a density gradient estimate. This recurrence equation is closely related to the well-known gradient ascent algorithm, with the improvement that accelerates the convergence of the recurrence iterations in regions of low data density. A more computationally stable form of the mean shift recurrence equation, since it avoids the explicit computation of the density estimate $\hat{f}_{\mathbf{H}}$ and density derivative estimate $\mathsf{D}\hat{f}_{\mathbf{H}}$, is

$$\mathbf{X}_{i,k+1} = \mathbf{X}_{i,k} + \boldsymbol{\beta}_{\mathbf{H}}(\mathbf{X}_{i,k}) = \frac{\sum_{\ell=1}^{n} \mathbf{X}_\ell \, g\big((\mathbf{X}_{i,k} - \mathbf{X}_\ell)^\top \mathbf{H}^{-1} (\mathbf{X}_{i,k} - \mathbf{X}_\ell)\big)}{\sum_{\ell=1}^{n} g\big((\mathbf{X}_{i,k} - \mathbf{X}_\ell)^\top \mathbf{H}^{-1} (\mathbf{X}_{i,k} - \mathbf{X}_\ell)\big)} \tag{4}$$

where $g(x) = x \exp(-\frac{1}{2}x)$ and $\boldsymbol{\beta}_{\mathbf{H}}(\boldsymbol{x}) = \dfrac{\sum_{\ell=1}^{n} \mathbf{X}_\ell g((\boldsymbol{x} - \mathbf{X}_\ell)^\top \mathbf{H}^{-1} (\boldsymbol{x} - \mathbf{X}_\ell))}{\sum_{\ell=1}^{n} g((\boldsymbol{x} - \mathbf{X}_\ell)^\top \mathbf{H}^{-1} (\boldsymbol{x} - \mathbf{X}_\ell))} - \boldsymbol{x}$. This $\boldsymbol{\beta}_{\mathbf{H}}$ is known as the mean shift, since it is the difference between the current iterate and a weighted mean of all data points. For our stopping rule, we iterate the recurrence in Eq. (4) until either we reach a maximum number of iterations (400) or that the distance between subsequent iterations is less than 0.001 times the minimal marginal IQR (interquartile range) of the input data. This heuristic stopping rule gives sensible results in most cases.

The result is a sequence of points $\{\mathbf{X}_{i,0}, \mathbf{X}_{i,1}, \dots\}$ which traces out a path, along the steepest ascent of the density gradient, from the data point $\mathbf{X}_i$ to the mode of the associated data-rich region. The data-rich regions are the 'basins of attraction' of the density gradient ascent. If the data points are associated with the same mode, then they are considered to be members of the same cluster. Thus the number of clusters is equal to the number of these basins of attraction. For more details on mean shift and other forms of density-based clustering, see Chacón and Duong (2018, Sect. 6.2).

5.1 Tidy clustering

The result of the mean shift clustering on the $n = 1280$ air quality measurements from 13:00 to 14:00 into 5 clusters is displayed on the left panel in Fig. 8. Observe that we do not need to specify the number of clusters in advance, and the clusters can be of any arbitrary shape. These represent two important advantages over k-means clustering, which requires an a priori number of clusters, and whose cluster shapes are more restricted than those in mean shift clustering. Cluster #4 (blue crosses) and #5 (magenta boxed crosses) are the most separate from the other clusters. The

points at the edges of cluster #3 (green squares), cluster #1 (red circles) and cluster #2 (khaki triangles) are close to together, and *k*-means clustering tends to assign them to the same cluster, whereas the directionality of the mean shift assigns them to different clusters. Since the mean shift relies on the density gradient ascent paths, we overlay the arrows of the quiver plot of the density gradient on the convex hulls of the mean shift clusters on the right of Fig. 8. We observe that the gradient ascent arrows within each cluster are oriented towards the modes.

The command for mean shift clustering for tidy data is `tidy_kms`. The output is similar to that for a single density estimate, except that the data points are returned rather than the estimation grid points, and that `estimate` indicates the estimated cluster label rather than the density estimate value.

```
R> ## tidy mean shift clusters
R> t7 <- tidy_kms(air3)
R> ggplot(t7, aes(x=co2, y=pm10)) + geom_point(aes(colour=estimate))
```

Since the direction along which the data points are shifted is directly related to the density gradient, the default bandwidth for mean shift clustering in `tidy_kms` is the plug-in bandwidth computed by `ks::Hpi(x, deriv.order=1)`. For the air quality measurements for 13:00 to 14:00, this bandwidth matrix is [441.0, 59.5; 59.5, 305.0]. The bandwidth choice is made with the goal of optimal identification of the density gradient ascent paths. It is also supported by the results that the optimal bandwidth for estimating the mode of a density is closely related to the optimal bandwidth for density gradient estimation (Chacón and Duong 2018, p. 138).

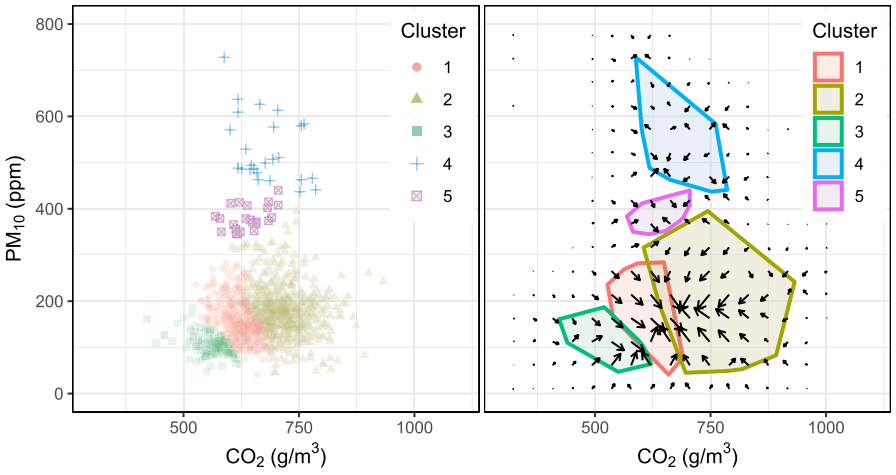

Fig. 8 Mean shift clusters for the air quality measurements 13:00–14:00 ($n = 1280$). (Left) Cluster members. (Right) Cluster convex hulls, superposed over the quiver plot of its density gradient estimate

5.2 Geospatial clustering

The result of the mean shift clustering on the $n = 93$ *G. yorkrakinensis* locations into 4 clusters is displayed on the left panel in Fig. 9. Cluster #4 (magenta crosses) is the most northerly and most separate from the other clusters. Cluster #2 (green triangles) forms the most southerly cluster and is also well-separated. The points on the right edge of cluster #1 (red circles) are close to those on the left edge of cluster #3 (cyan squares), though the directionality of the mean shift, as indicated by the black arrows of the density gradient, assigns them to different clusters.

The command for mean shift clustering for geospatial data is `st_kms`. To produce the mean shift clusters, with at least 3 members in each cluster, the code snippet is:

```
R> ## geospatial mean shift clusters geom_sf plot
R> s7 <- st_kms(yorkr, min.clust.size=3)
R> ggplot(s7) + geom_sf(aes(colour=estimate))
R> ## base R plot
R> plot(s7, pch=16)
```

The default bandwidth for mean shift clustering in `st_kms` is the plug-in bandwidth computed by `ks::Hpi(x, deriv.order=1)`. For the *G. yorkrakinensis* locations, this is [4.39e8, −4.36e8; −4.36e8, 7.73e8].

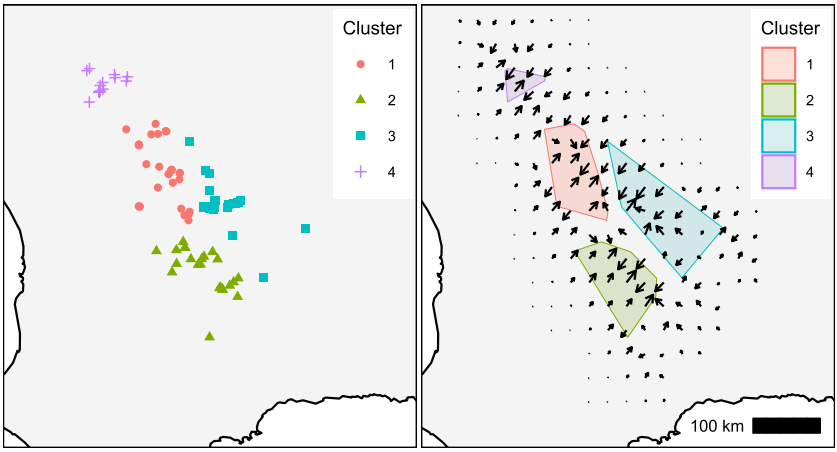

Fig. 9 Mean shift clusters for *G. yorkrakinensis* ($n = 93$). (Left) Cluster members. (Right) Cluster convex hulls, superposed over the quiver plot of its density gradient estimate

6 Software

6.1 Other data analysis settings

All the functionality in the `ks` package that involve 1- and 2-dimensional kernel smoothers are implemented for tidy data in the `eks` package. In addition to `tidy_kde`, `tidy_kda`, `tidy_kdde`, and `tidy_kms` in Sects. 2–5, these functions include

`tidy_kde_boundary`	Boundary density estimate where the kernel function K is modified explicitly in the boundary region
`tidy_kde_truncate`	Truncated density estimate where the standard density estimate $\hat{f}$ is truncated and rescaled to give unit integral over the boundary region
`tidy_kde_sp`	Sample point density estimate where the bandwidth $\mathbf{H}(\cdot)$ varies with the data point $\mathbf{X}_i$
`tidy_kde_balloon`	Balloon density estimate where the bandwidth $\mathbf{H}(\cdot)$ varies with the estimation point $\boldsymbol{x}$
`tidy_kdcde`	Deconvolved density estimate for data $\mathbf{X}_i$ observed with error
`tidy_kcde`	Cumulative distribution estimate $\hat{F}$
`tidy_kcopula`	Copula estimate with uniformly distributed marginal distributions
`tidy_kroc`	ROC (receiver operating characteristic) curve of the 2-sample comparison of the marginal distribution estimates $\hat{F}_1, \hat{F}_2$
`tidy_kdr`	Density ridge estimate which is a generalisation of principal components to lower dimensional manifolds
`tidy_kde_local_test`	Significance testing for the 2-sample comparison of the difference of the density estimates $\hat{f}_1 - \hat{f}_2$
`tidy_kfs`	Significance testing for modal regions where the second derivative (Hessian matrix) of the density estimate $\hat{f}$ is positive definitive.

All of the above functions (except `tidy_kcopula`) are implemented for 2-dimensional geospatial data as `st_k*`. All of these utilise the appropriate default bandwidth selector from the `ks` package. For brevity, we do not illustrate them here: their usage is demonstrated in their help pages contained in the `eks` package, and the details of the statistical framework in these data analysis settings are provided in Chacón and Duong (2018).

6.2 Export to external GIS

The ability to export the geospatial kernel estimates to standard vectorial geospatial data formats extends the functionality of the `eks` package to GIS software. The commands to export to the geopackage format are:

```
R> ## export to vectorial geospatial format
R> sf::write_sf(yorkr, dsn="grevillea.gpkg", layer="yorkr")
R> sf::write_sf(st_get_contour(s1), dsn="grevillea.gpkg", layer="yorkr_cont")
R> sf::write_sf(s6$sf, dsn="grevillea.gpkg", layer="yorkr_quiver")
```

The `grevillea.gpkg` geopackage consists of four layers: `yorkr` for the point geometries of the *G. yorkrakinensis* locations, `yorkr_cont` for the multi-polygons of the quartile contour regions of the density estimate, and `yorkr_quiver` for the linestrings of the density gradient flows.

This `grevillea.gpkg` geopackage can be subsequently employed in QGIS (QGIS.org 2021), which is an industry standard software for GIS practitioners since it offers features that are not available in R. For example, it has an interactive point-and-click interface, and it incorporates fast rendering of the OpenStreetMap base maps. A screenshot from a QGIS analysis for a quiver plot overlaid on a density estimate is given in the left panel of Fig. 10. Recall that quiver plots can be difficult to produce with geospatial data in `ggplot2` graphics, since the arrows require trial and error to display suitably with `ggplot2::geom_segment`. In contrast, quiver plots are straightforward in QGIS since rescaleable arrows are a native feature.

In addition, QGIS efficiently handles raster geospatial data. Whilst the `grid` field of a kernel estimate consists of the rectangular polygons for each pixel of the estimation grid, it can be converted to a raster via the `stars` package. The heat map of the converted raster is displayed in QGIS on the right of Fig. 10.

```
R> ## export to raster format
R> stars::write_stars(stars::st_rasterize(s1$grid), dsn="grevillea.gpkg",
   options=c("APPEND_SUBDATASET=YES", "RASTER_TABLE=yorkr_raster"))
```

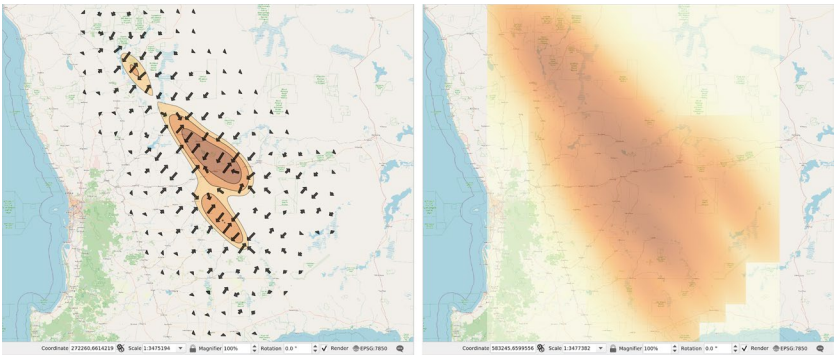

Fig. 10 Screenshots of QGIS analysis for *G. yorkrakinensis* ($n = 93$). (Left) Contour plot of vectorial density estimate and quiver plot of vectorial density gradient estimate. (Right) Heat map of raster density estimate

7 Conclusion

We have introduced a new R package `eks` which serves as a bridge from the comprehensive suite of kernel smoothers in the `ks` package to the tidyverse and geospatial analysis. A wide range of kernel smoothing methods are available, which (i) improve on the existing kernel density estimates, and (ii) widen the accessibility to more complex kernel-based data analyses, such as density gradient estimation, density-based classification (supervised learning) and mean shift clustering (unsupervised learning). The `eks` package provides practitioners with additional tools to create compelling statistical visualisations from kernel smoothers, whether they are using tidy or geospatial data, or whether they are using base R or tidyverse graphics.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s00180-024-01543-9).

References

Baíllo A, Chacón JE (2021) Chapter 1—Statistical outline of animal home ranges: an application of set estimation. In: Rao AS, Rao C (eds) Data science: theory and applications. Elsevier, Amsterdam, pp 3–37

Beck G, Duong T, Azzag H, Lebbah M (2016) Distributed mean shift clustering with approximate nearest neighbours. In: Proceedings of the 2016 international conference on neural networks (IJCNN), pp 3110–3115

Béranger B, Duong T, Perkins-Kirkpatrick SE, Sisson SA (2019) Tail density estimation for exploratory data analysis using kernel methods. J Nonparametric Stat 31:144–174

Bowman AW, Foster P (1993) Density based exploration of bivariate data. Stat Comput 3:171–177

Chacón JE, Duong T (2010) Multivariate plug-in bandwidth selection with unconstrained bandwidth matrices. Test 19:375–398

Chacón JE, Duong T (2018) Multivariate kernel smoothing and its applications. Chapman and Hall/CRC, Boca Raton

Chacón JE, Duong T, Wand MP (2011) Asymptotics for general multivariate kernel density derivative estimators. Stat Sin 21:807–840

Devroye L, Györfi L, Lugosi G (1996) A probabilistic theory of pattern recognition. Springer, Berlin

Duong T (2007) ks: kernel density estimation and kernel discriminant analysis for multivariate data in R. J Stat Softw 21(7):1–16
Duong T (2023) eks: tidy and geospatial kernel smoothing. R package version 1.0.3
Duong T, Hazelton ML (2003) Plug-in bandwidth matrices for bivariate kernel density estimation. J Nonparametric Stat 15:17–30
Fukunaga K, Hostetler L (1975) The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Trans Inf Theory 21:32–40
Hyndman RJ (1996) Computing and graphing highest density regions. Am Stat 50:120–126
Kalair K, Connaughton C (2021) Anomaly detection and classification in traffic flow data from fluctuations in the flow-density relationship. Transp Res Part C Emerg Technol 127:103178
Myers N, Mittermeier RA, Mittermeier CG, Da Fonseca GAB, Kent J (2000) Biodiversity hotspots for conservation priorities. Nature 403:853–858
OGC (2010) OpenGIS implementation standard for geographic information—simple feature access—part 1: common architecture. Version 1.2.1
O'Hara-Wild M (2019) ggquiver: quiver plots for 'ggplot2'. R package version 0.2.0
Otto J, Kahle D (2023) ggdensity: interpretable bivariate density visualization with 'ggplot2'. R package version 1.0.0
Pebesma E (2018) Simple features for R: standardized support for spatial vector data. R J 10:439–446
QGIS.org (2021) QGIS geographic information system. QGIS Association
RATP (2016) Qualité de l'air mesurée dans la station Châtelet. Régie autonome des transports parisiens, Département Développement, Innovation et Territoires. https://data.iledefrance.fr/explore/dataset/qualite-de-lair-mesuree-dans-la-station-chatelet. Accessed 27 Sept 2017
Rudis B, Bolker B, Schulz J (2017) ggalt: extra coordinate systems, 'geoms', statistical transformations, scales and fonts for 'ggplot2'. R package version 0.4.0
Schauer K, Duong T, Bleakley K, Bardin S, Bornens M, Goud B (2010) Probabilistic density maps to study global endomembrane organization. Nat Methods 7:560–568
Wickham H (2014) Tidy data. J Stat Softw 59(10):1–23
Wickham H (2016) ggplot2: elegant graphics for data analysis. Springer-Verlag, New York