Using ks for bivariate kernel density estimation

Tarn Duong
Department of Statistics, University of New South Wales
Sydney Australia

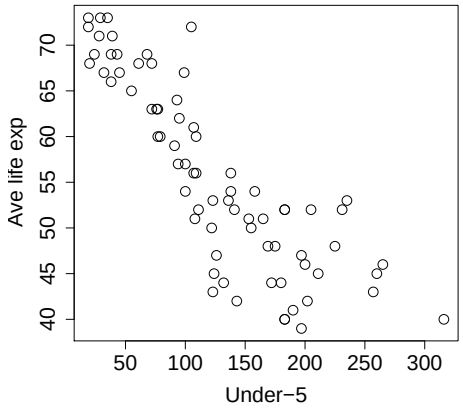
19 March 2007

1 Introduction

Kernel density estimation has become a popular tool for visualising the distribution of data. See Simonoff (1996), for example, for an overview. When multivariate kernel density estimation is considered it is usually in the constrained context with diagonal bandwidth matrices, e.g. in the R packages **sm** (Bowman and Azzalini, 2005) and **KernSmooth** (Wand, 2006). We introduce a new R package **ks** for kernel smoothing which implements diagonal and unconstrained data-driven bandwidth matrices. The main theoretical advances are in the development of new methods for the latter. It is able to analyse 1- to 6-dimensional data with graphical visualisation for 1- to 3-dimensional data. Currently it is the most comprehensive kernel density estimation package available in R. This vignette focuses on kernel density estimation for the 2-dimensional case.

The `unicef` dataset is included in the **ks** package. It contains the number of deaths of children under 5 years of age per 1000 live births and the average life expectancy (in years) at birth for 73 countries with GNI (Gross National Income) less than 1000 US dollars per annum per capita. The scatterplot is below. A major goal of kernel density estimation is to find a description which summarises the important characteristics of the data.

```
> library(ks)
> data(unicef)
> plot(unicef)
```



2 Kernel density estimation

For a bivariate sample $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n$ drawn from a density f, the kernel density estimate is

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

where $\boldsymbol{x} = (x_1, x_2)^T$ and $\boldsymbol{X}_i = (X_{i1}, X_{i2})^T, i = 1, 2, \ldots, n$. Here $K(\boldsymbol{x})$ is the bivariate kernel (which we assume to be a probability density function); $\mathbf{H} = \begin{bmatrix} h_1^2 & h_{12} \\ h_{12} & h_2^2 \end{bmatrix}$ is the bandwidth matrix which is symmetric and positive-definite; and $K_{\mathbf{H}}(\boldsymbol{x}) = |\mathbf{H}|^{-1/2} K(\mathbf{H}^{-1/2}\boldsymbol{x})$. The choice of K is not crucial – we take $K(\boldsymbol{x}) = (2\pi)^{-1} \exp(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{x})$ the standard normal throughout. On the other hand, the choice of $\mathbf{H}$ *is* crucial in determining the performance of $\hat{f}$.

We measure the performance of $\hat{f}$ (in common with the majority of researchers in this field) using the Mean Integrated Squared Error (MISE) criterion,

$$\operatorname{MISE}(\mathbf{H}) = \mathbb{E} \int_{\mathbb{R}^2} [\hat{f}(\boldsymbol{x}; \mathbf{H}) - f(\boldsymbol{x})]^2 \, d\boldsymbol{x}.$$

Our aim in bandwidth selection is to estimate

$$\mathbf{H}_{\operatorname{MISE}} = \operatorname*{argmin}_{\mathbf{H}} \operatorname{MISE}(\mathbf{H}),$$

over the space of all symmetric, positive definite 2×2 matrices. It is well known that the optimal bandwidth $\mathbf{H}_{\operatorname{MISE}}$ does not have a closed form. To make progress it is usual to employ an asymptotic approximation, known as the AMISE (Asymptotic MISE):

$$\operatorname{AMISE}(\mathbf{H}) = n^{-1}(4\pi)^{-1}|\mathbf{H}|^{-1/2} + \tfrac{1}{4}(\operatorname{vech}^T \mathbf{H})\boldsymbol{\Psi}_4(\operatorname{vech} \mathbf{H}) \tag{2}$$

where $R(K) = \int_{\mathbb{R}^2} K(\boldsymbol{x})^2 \, d\boldsymbol{x} = (4\pi)^{-1}$ and vech is the vector half operator i.e.

$$\operatorname{vech} \mathbf{H} = \operatorname{vech} \begin{bmatrix} h_1^2 & h_{12} \\ h_{12} & h_2^2 \end{bmatrix} = \begin{bmatrix} h_1^2 \\ h_{12} \\ h_2^2 \end{bmatrix}.$$

See Wand and Jones (1995, p. 98) for the general expression of the $\frac{1}{2}d(d+1) \times \frac{1}{2}d(d+1)$ matrix $\boldsymbol{\Psi}_4$. For $d = 2$, we can show that

$$\boldsymbol{\Psi}_4 = \begin{bmatrix} \psi_{40} & 2\psi_{31} & \psi_{22} \\ 2\psi_{31} & 4\psi_{22} & 2\psi_{13} \\ \psi_{22} & 2\psi_{13} & \psi_{04} \end{bmatrix} \tag{3}$$

where the integrated density derivative functional is

$$\psi_{r_1, r_2} = \int_{\mathbb{R}^2} f^{(r_1, r_2)}(\boldsymbol{x}) f(\boldsymbol{x}) \, d\boldsymbol{x}$$

and the partial derivatives of f are

$$f^{(r_1, r_2)}(\boldsymbol{x}) = \frac{\partial^4}{\partial_{x_1}^{r_1} \partial_{x_2}^{r_2}} f(\boldsymbol{x}).$$

The subscript 4 on $\mathbf{\Psi}$ relates to the order of the derivatives involved. We make use of the tractability of AMISE by seeking

$$\mathbf{H}_{\text{AMISE}} = \operatorname*{argmin}_{\mathbf{H}} \text{AMISE}\,(\mathbf{H}).$$

For the next step we estimate the MISE or AMISE. A data-driven bandwidth selector is either

$$\hat{\mathbf{H}} = \operatorname*{argmin}_{\mathbf{H}} \widehat{\text{MISE}}\,(\mathbf{H}) \quad \text{or} \quad \hat{\mathbf{H}} = \operatorname*{argmin}_{\mathbf{H}} \widehat{\text{AMISE}}\,(\mathbf{H}) \tag{4}$$

Different selectors arise from the different methods used in the estimation step.

3 Plug-in bandwidth selectors

The most well-known univariate plug-in selector is due to Sheather and Jones (1991). Plug-in selectors require pilot estimates of the ψ_{r,r_2} functionals which comprise $\mathbf{\Psi}_4$. This in turn produces an estimate of the AMISE

$$\text{PI}(\mathbf{H}) = n^{-1}(4\pi)^{-1}|\mathbf{H}|^{-1/2} + \tfrac{1}{4}(\text{vech}^T\,\mathbf{H})\hat{\mathbf{\Psi}}_4(\text{vech}\,\mathbf{H}) \tag{5}$$

that can be numerically minimised to give the plug-in bandwidth matrix, $\hat{\mathbf{H}}_{\text{PI}}$. To compute $\hat{\mathbf{\Psi}}_4$ we need to use a helper or 'pilot' bandwidth matrix $\mathbf{G}$. Like $\mathbf{H}$, we need to choose a sensible value for $\mathbf{G}$. If we note that $\psi_{r_1,r_2} = \mathbb{E}\, f^{(r_1,r_2)}(\boldsymbol{X})$ where $\boldsymbol{X} \sim f$, then a natural estimator of ψ_{r_1,r_2} is

$$\hat{\psi}_{r_1,r_2}(\mathbf{G}) = n^{-1}\sum_{i=1}^{n} \hat{f}^{(r_1,r_2)}(\boldsymbol{X}_i;\mathbf{G}) = n^{-2}\sum_{i=1}^{n}\sum_{j=1}^{n} K_{\mathbf{G}}^{(r_1,r_2)}(\boldsymbol{X}_i - \boldsymbol{X}_j). \tag{6}$$

3.1 AMSE pilot bandwidth selectors

We consider pilot bandwidth matrices of the form $\mathbf{G} = g^2\mathbf{I}$. It may appear that we are contradicting ourselves since we advocate using unconstrained matrices for $\mathbf{H}$. It turns out that a restricted form on $\mathbf{G}$ has less effect on the performance of $\hat{f}$ than a restricted form on $\mathbf{H}$. We can reduce the effect of the restricted form of $\mathbf{G}$ by using an appropriate pre-transformation (we return to this topic in Section 6).

The MSE (Mean Squared Error) for $\hat{\psi}_{\boldsymbol{r}}(g)$ is

$$\text{MSE}\,\hat{\psi}_{\boldsymbol{r}}(g) = \mathbb{E}[\hat{\psi}_{\boldsymbol{r}}(g) - \psi_{\boldsymbol{r}}]^2.$$

The bandwidths which are optimal for the Asymptotic MSE are (a) if all elements of $\boldsymbol{r}$ are even then

$$g_{\boldsymbol{r},\text{AMSE}} = \left[\frac{A_{\boldsymbol{r}}}{n}\right]^{1/(d+6)} \tag{7}$$

and (b) if at least one of $\boldsymbol{r}$ is odd then

$$g_{\boldsymbol{r},\text{AMSE}} = \left[\frac{B_{\boldsymbol{r}}}{n^2}\right]^{1/(d+12)}. \tag{8}$$

See Wand and Jones (1994) for explicit expressions for $A_{\boldsymbol{r}}$ and $B_{\boldsymbol{r}}$. They depend on the density f and the kernel K but not on the sample size n. In fact, they depend on f only via higher-order $\psi_{\boldsymbol{r}}$ functionals. These functionals can be estimated by a normal reference estimate or by

another kernel estimate with another pilot bandwidth. The former is known as a 1-stage pilot selector, the latter as 2-stage. See Duong and Hazelton (2003) for a full description of the pilot selector algorithms. Intuitively we can think of 2 estimation stages as a strategy to reduce the dependence on the normality assumptions for 1 stage estimation.

The $\hat{\mathbf{\Psi}}_4$ estimated in this element-wise way uses a different pilot bandwidth for each unique element (5 in total). Then $\hat{\mathbf{\Psi}}_4$ is not guaranteed to be positive definite. Hence using appropriate estimators of each element of a matrix will not necessarily lead to an appropriate estimator of the matrix as a whole. Positive-definiteness can be guaranteed by using a single, common pilot bandwidth.

3.2 SAMSE pilot bandwidth selectors

Modifying AMSE pilot selectors, we derive a SAMSE (Sum of Asymptotic Mean Squared Error) pilot selector. This type of selector has been specially devised to maintain the positive definiteness of $\hat{\mathbf{\Psi}}_4$ which is crucial to the numerical minimisation of the plug-in criterion PI. This method is also simpler and more parsimonious than AMSE selectors.

The fourth order SAMSE can be expressed as

$$\text{SAMSE}_4(g) = n^{-2}g^{-2d-8}A_1 + n^{-1}g^{-d-2}A_2 + \tfrac{1}{4}g^4 A_3 \tag{9}$$

where A_1 and A_3 are positive constants and A_2 is a negative constant. This has a minimum at

$$g_{4,\text{SAMSE}} = \left[\frac{A_4}{n}\right]^{1/(d+6)}. \tag{10}$$

Explicit expressions for A_1, A_2, A_3 and A_4 can be found in Duong and Hazelton (2003). Equation (10) is the single SAMSE optimal pilot bandwidth: only one pilot bandwidth is required to compute $\hat{\mathbf{\Psi}}_4$. Again we have the choice of either 1 or 2 stages for estimating the pilot bandwidth.

3.3 R examples

Use `Hpi` for full plug-in selectors and `Hpi.diag` for diagonal plug-in selectors. There are two arguments which further specify the plug-in selector used: `nstage` is the number of pilot estimation stages (1 or 2) and `pilot` is the type of pilot estimation (`"amse"` or `"samse"`). The other argument `pre` involves the pre-transformations outlined in Section 6.

```
> Hpi1 <- Hpi(x = unicef, nstage = 1, pilot = "amse", pre = "scale")

          [,1]      [,2]
[1,] 391.02859 -34.73347
[2,] -34.73347   9.89807

> Hpi2 <- Hpi(x = unicef, nstage = 2, pilot = "samse", pre = "sphere")

          [,1]       [,2]
[1,]  810.9140 -108.73376
[2,] -108.7338   19.79100

> Hpi3 <- Hpi.diag(x = unicef, nstage = 2, pilot = "amse", pre = "scale")
```

```
         [,1]     [,2]
[1,] 201.5118 0.000000
[2,]   0.0000 6.242821
```

```
> Hpi4 <- Hpi.diag(x = unicef, nstage = 2, pilot = "samse", pre = "scale")
```

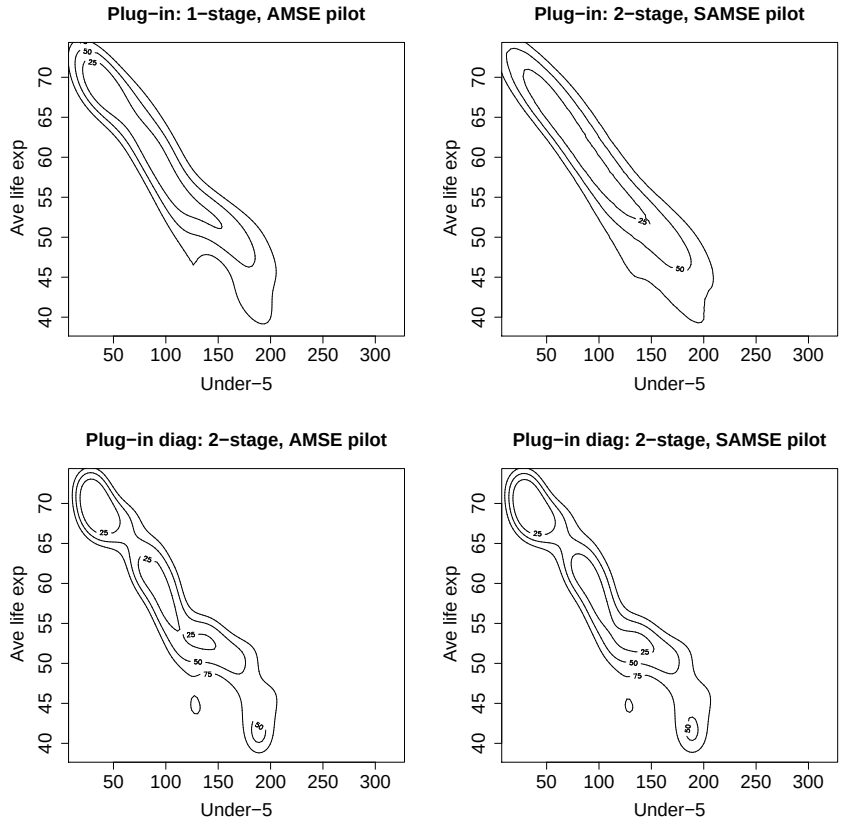
```
         [,1]     [,2]
[1,] 227.0192 0.000000
[2,]   0.0000 6.179491
```

To compute a kernel density estimate (Equation (1)), the command is `kde`.

```
> fhat1 <- kde(x = unicef, H = Hpi1)
> fhat2 <- kde(x = unicef, H = Hpi2)
> fhat3 <- kde(x = unicef, H = Hpi3)
> fhat4 <- kde(x = unicef, H = Hpi4)
```

We can use the `plot` command to display these kernel density estimates. The default is a contour plot with the upper 25%, 50% and 75% contours.

```
> plot(fhat1, main = "Plug-in: 1-stage, AMSE pilot")
> plot(fhat2, main = "Plug-in: 2-stage, SAMSE pilot")
> plot(fhat3, main = "Plug-in diag: 2-stage, AMSE pilot")
> plot(fhat4, main = "Plug-in diag: 2-stage, SAMSE pilot")
```



4 Cross validation bandwidth selectors

Cross-validation selectors are the main alternative to plug-in selectors. There are three main types of cross-validation selectors: least squares, biased and smoothed.

4.1 Least squares cross validation

The bivariate version of the least squares cross validation (LSCV) criterion of Rudemo (1982) and Bowman (1984) is

$$\mathrm{LSCV}(\mathbf{H}) = \int_{\mathbb{R}^2} \hat{f}(\boldsymbol{x}; \mathbf{H})^2 \, d\boldsymbol{x} - 2n^{-1} \sum_{i=1}^{n} \hat{f}_{-i}(\boldsymbol{X}_i; \mathbf{H}),$$

where the leave-one-out estimator is

$$\hat{f}_{-i}(\boldsymbol{x}; \mathbf{H}) = (n-1)^{-1} \sum_{\substack{j=1 \\ j \neq i}}^{n} K_{\mathbf{H}}(\boldsymbol{x} - \boldsymbol{X}_j).$$

The LSCV selector $\hat{\mathbf{H}}_{\mathrm{LSCV}}$ is the minimiser of $\mathrm{LSCV}(\mathbf{H})$. We can rewrite LSCV as

$$\mathrm{LSCV}(\mathbf{H}) = n^{-1}(4\pi)^{-1}|\mathbf{H}|^{-1/2} + n^{-1}(n-1)^{-1} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} (K_{2\mathbf{H}} - 2K_{\mathbf{H}})(\boldsymbol{X}_i - \boldsymbol{X}_j). \tag{11}$$

We can show that $\mathbb{E}[\mathrm{LSCV}(\mathbf{H})] = \mathrm{MISE}(\mathbf{H}) - R(f)$, indicating that LSCV estimates the MISE directly.

4.2 Biased cross validation

Plug-in methods use a pilot bandwidth matrix $\mathbf{G}$, which is independent of $\mathbf{H}$, to estimate $\boldsymbol{\Psi}_4$. For BCV, we set $\mathbf{G} = \mathbf{H}$ and use slightly different estimators. There are two versions of BCV, depending on the estimator of ψ_{r_1,r_2} with $r_1 + r_2 = 4$, see Sain et al. (1994). We can use

$$\check{\psi}_{r_1,r_2}(\mathbf{H}) = n^{-2} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} K_{2\mathbf{H}}^{(r_1,r_2)}(\boldsymbol{X}_i - \boldsymbol{X}_j) \tag{12}$$

or we could use

$$\tilde{\psi}_{r_1,r_2}(\mathbf{H}) = n^{-1} \sum_{i=1}^{n} \hat{f}_{-i}^{(r_1,r_2)}(\boldsymbol{X}_i; \mathbf{H}) = n^{-1}(n-1)^{-1} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} K_{\mathbf{H}}^{(r_1,r_2)}(\boldsymbol{X}_i - \boldsymbol{X}_j). \tag{13}$$

The estimates $\check{\boldsymbol{\Psi}}_4$ and $\tilde{\boldsymbol{\Psi}}_4$ are obtained from $\boldsymbol{\Psi}_4$ by substituting $\check{\psi}_{r_1,r_2}$ and $\tilde{\psi}_{r_1,r_2}$ for ψ_{r_1,r_2}. The BCV1 function is the version of BCV with $\check{\boldsymbol{\Psi}}_4$

$$\mathrm{BCV1}(\mathbf{H}) = n^{-1}(4\pi)^{-1}|\mathbf{H}|^{-1/2} + \tfrac{1}{4}(\mathrm{vech}^T \mathbf{H})\check{\boldsymbol{\Psi}}_4(\mathrm{vech}\, \mathbf{H}) \tag{14}$$

and the BCV2 function is the version with $\tilde{\boldsymbol{\Psi}}_4$

$$\mathrm{BCV2}(\mathbf{H}) = n^{-1}(4\pi)^{-1}|\mathbf{H}|^{-1/2} + \tfrac{1}{4}(\mathrm{vech}^T \mathbf{H})\tilde{\boldsymbol{\Psi}}_4(\mathrm{vech}\, \mathbf{H}). \tag{15}$$

The BCV selectors $\hat{\mathbf{H}}_{\mathrm{BCV}}$ are the minimisers of the appropriate BCV function.

4.3 Smoothed cross validation

Smoothed cross validation (SCV), introduced by Hall et al. (1992), can be thought of as a hybrid of LSCV and BCV. The SCV criterion takes the asymptotic integrated variance from the BCV but attempts to estimate the integrated squared bias exactly rather than using its asymptotic form

$$\mathrm{SCV}(\mathbf{H}) = n^{-1}(4\pi)^{-1}|\mathbf{H}|^{-1/2} + n^{-2}\sum_{i=1}^{n}\sum_{j=1}^{n}(K_{2\mathbf{H}+2\mathbf{G}} - 2K_{\mathbf{H}+2\mathbf{G}} + K_{2\mathbf{G}})(\boldsymbol{X}_i - \boldsymbol{X}_j). \quad (16)$$

The SCV selector $\hat{\mathbf{H}}_{\mathrm{SCV}}$ is the minimiser of $\mathrm{SCV}(\mathbf{H})$.

Again, we consider pilot bandwidth matrices of the form $\mathbf{G} = g^2\mathbf{I}$. We generalise the process of Jones et al. (1991) to find an optimal pilot bandwidth selector. We can show that the pilot bandwidth which minimises

$$Q(g) = \sum_{i=1}^{d}\sum_{j=i}^{d}\mathbb{E}[\hat{h}_{\mathrm{SCV},ij} - h_{\mathrm{AMISE},ij}]^2,$$

where $\hat{h}_{\mathrm{SCV},ij}$ is the (i,j)-th element of $\hat{\mathbf{H}}_{\mathrm{SCV}}$ and $h_{\mathrm{AMISE},ij}$ is the (i,j)-th element of $\mathbf{H}_{\mathrm{AMISE}}$, is

$$g_{\mathrm{AMSE}} = \left[\frac{B}{n}\right]^{1/(d+6)} \quad (17)$$

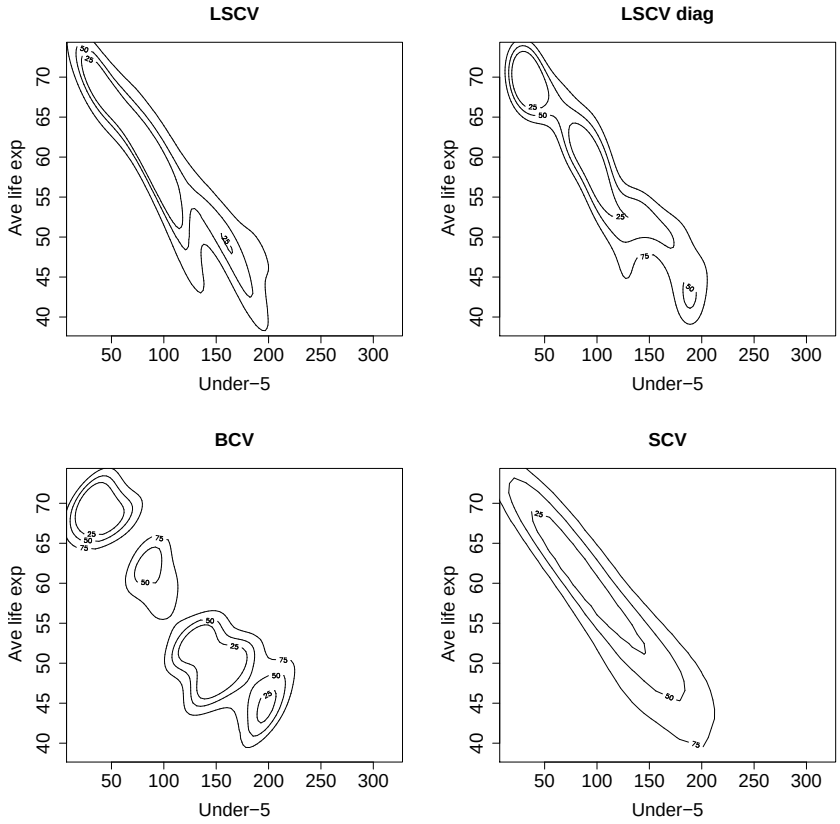
where B is a constant which depends on f, K but not n. The details of the derivation of this pilot bandwidth is available in Duong and Hazelton (2005), along with a full description of the SCV selector algorithm.

4.4 R examples

We continue with the `unicef` data. `Hlscv` and `Hlscv.diag` are the full and diagonal LSCV selectors. (The `Hstart` argument in the below code specifies the initial value for the numerical minimisation - it can be specified for any of the bandwidth selector functions). `Hbcv` implements both BCV1 and BCV2. The default is BCV1; set `whichbcv=2` to use BCV2. Their diagonal counterpart is `Hbcv.diag`. `Hscv` is the full SCV selector (no diagonal version is available). Its argument `pre` is the same as for `Hpi` and `Hpi.diag` in Section 3.3.

```
> Hlscv1 <- Hlscv(unicef)

          [,1]      [,2]
[1,] 388.18250 -83.34084
[2,] -83.34084  25.12909

> Hlscv2 <- Hlscv.diag(unicef, Hstart = Hlscv1)

         [,1]     [,2]
[1,] 194.4292  0.00000
[2,]   0.0000 11.11751

> Hbcv1 <- Hbcv(unicef)
```

```
          [,1]      [,2]
[1,] 1087.0682 135.33067
[2,]  135.3307  23.58613

> Hbcv2 <- Hbcv.diag(unicef, whichbcv = 2)

         [,1]     [,2]
[1,] 1072.781 0.000000
[2,]    0.000 9.298466

> Hscv1 <- Hscv(unicef, pre = "sphere")

          [,1]      [,2]
[1,] 1323.3189 -191.9278
[2,] -191.9278   35.0105

> Hscv2 <- Hscv(unicef, pre = "scale")

          [,1]      [,2]
[1,] 694.14462 -73.09935
[2,] -73.09935  17.49451

> fhat1 <- kde(unicef, Hlscv1)
> fhat2 <- kde(unicef, Hlscv2)
> fhat3 <- kde(unicef, Hbcv1)
> fhat4 <- kde(unicef, Hscv1)

> plot(fhat1, main = "LSCV")
> plot(fhat2, main = "LSCV diag")
> plot(fhat3, main = "BCV")
> plot(fhat4, main = "SCV")
```



5 More graphics

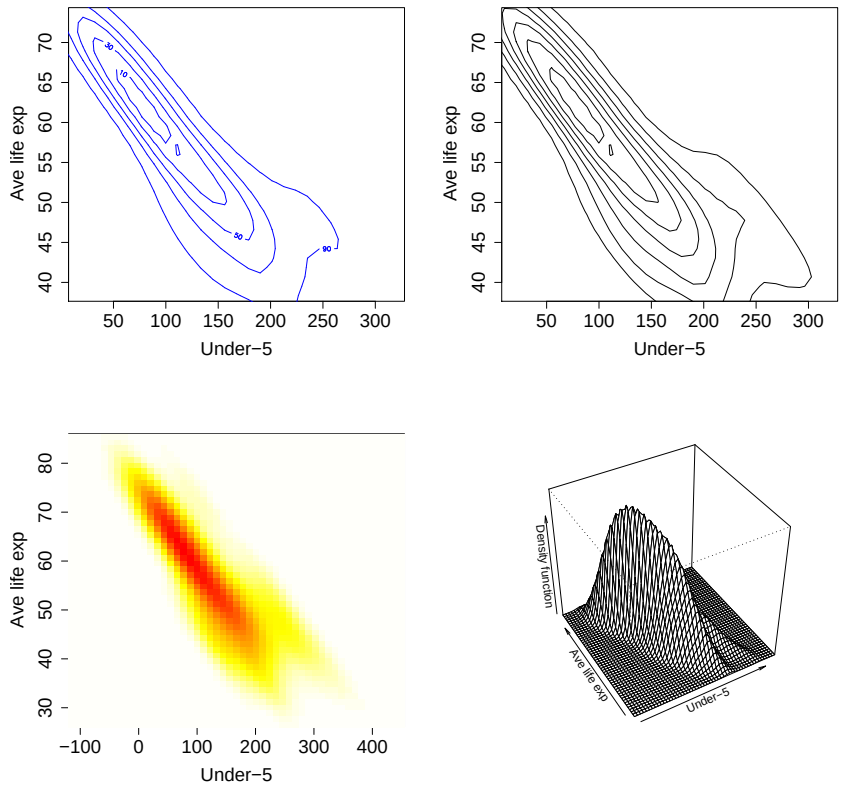
Slice or contour plots are the default or can be explicitly called by setting `display="slice"`. The way the contours are plotted is controlled by `cont` or `ncont`: only one of these needs to be set. The argument `cont` takes a vector of percentages and produces a set of contours at the levels corresponding to the percentages of the maximum height of the density estimate. The argument `ncont` takes a number and `R` tries to compute a pretty set of `ncont` contours. The colour(s) of the contour lines is `lcol` and the colour(s) of the plotting symbols is `ptcol`. The logical flags `drawlabels` and `drawpoints` indicate whether to draw the labels for the contours levels or the data points.

An alternative to contour plots are the image or heat plots, called by `display="image"`. These are similar to contour plots except that the heights of the density estimate are represented by different colours rather than with different level curves. The default colours is `heat.colors` but we use instead `rev(heat.colors)` which gives us the high values as red and the low values (close to zero) as white with yellow/orange as intermediate.

The other alternative is a perspective or wire-frame plot, called by `display="persp"`. This is an attempt to capture the three-dimensional structure more directly than the image or contour plots.

```
> plot(fhat4, lcol = "blue", ptcol = "black", cont = seq(10, 90,
+     by = 20))
> plot(fhat4, ncont = 8, drawlabels = FALSE, drawpoints = FALSE)
```

```
> plot(fhat4, display = "image", col = rev(heat.colors(100)))
> plot(fhat4, display = "persp")
```



6 Pre-transformations

The plug-in and SCV selector functions use pilot bandwidths of the form $\mathbf{G} = g^2\mathbf{I}$ which will be inappropriate if the dispersion of the data differs markedly between the two coordinate directions. We estimate using the transformed data $\boldsymbol{X}_1^*, \boldsymbol{X}_2^*, \ldots, \boldsymbol{X}_n^*$, where the transformation is either *sphering*

$$\boldsymbol{X}^* = \mathbf{S}^{-1/2}\boldsymbol{X}$$

where $\mathbf{S}$ is the sample covariance matrix of the untransformed data; or *scaling*

$$\boldsymbol{X}^* = \mathbf{S}_D^{-1/2}\boldsymbol{X}$$

where $\mathbf{S}_D = \text{diag}(s_1^2, s_2^2)$ and s_1^2, s_2^2 are the marginal sample variances. The bandwidth matrix $\hat{\mathbf{H}}^*$ for the sphered or scaled data can be back transformed to the original scale by $\hat{\mathbf{H}} = \mathbf{S}^{1/2}\hat{\mathbf{H}}^*\mathbf{S}^{1/2}$ or $\hat{\mathbf{H}} = \mathbf{S}_D^{1/2}\hat{\mathbf{H}}^*\mathbf{S}_D^{1/2}$, as appropriate.

7 Large sample sizes and binned kernel estimation

For large sample sizes, direct computation of kernel estimates becomes computationally difficult. One common technique for increasing computational speed for these large samples is binned

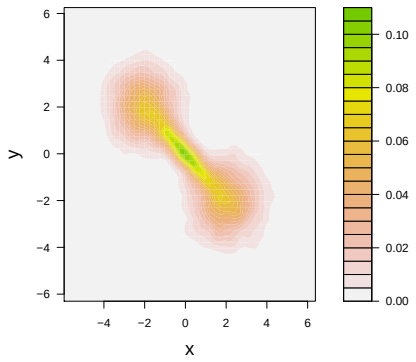
kernel estimation, see Wand and Jones (1994, Appendix D). Unfortunately binned estimation is only defined with diagonal bandwidth matrices. So applicable cases include computing the pilot bandwidth matrices (which are parameterised as $g^2\mathbf{I}$) for general plug-in and SCV selectors; and for kernel density estimators with diagonal bandwidth matrices.

We generate a 10 000 sample from the 'dumbbell' density, used in Duong and Hazelton (2005):

```
> mus <- rbind(c(-2, 2), c(0, 0), c(2, -2))
> Sigmas <- rbind(diag(2), 0.8 * invvech(c(1, -0.9, 1)), diag(2))
> cwt <- 3/11
> props <- c((1 - cwt)/2, cwt, (1 - cwt)/2)
> x <- rmvnorm.mixt(10000, mus, Sigmas, props)

> H.pidiag <- Hpi.diag(x, binned = TRUE)

           [,1]      [,2]
[1,] 0.02724536 0.0000000
[2,] 0.00000000 0.0281903

> H.pi <- Hpi(x, binned = TRUE)

           [,1]       [,2]
[1,]  0.0803607 -0.0692478
[2,] -0.0692478  0.0824435
```

For large sample sizes, we don't recommend plotting with slice/contour plots because computing the relative contour heights is computationally intensive. Instead, image, perspective or filled contour plots are more efficient.

```
> fhat.diag <- kde(x, H = H.pidiag, binned = TRUE)
> plot(fhat.diag, display = "filled", col = rev(terrain.colors(30)))
```



8 General recommendations

It is generally advisable to try a few different selectors and visually examine the resulting density estimates. The different bandwidth selectors available in **ks** may now pose a problem of *too* much

choice. The full bandwidth selectors will be better than their diagonal counterparts when the data have large mass oriented obliquely to the co-ordinate axes (as is the case for the `unicef` data). Amongst the full selectors, we advise against using the BCV selector. The LSCV selector is useful in some cases though its performance is known to be highly variable. The 2-stage plug-in and the SCV selectors can be viewed as generally recommended bandwidth selectors.

9 Acknowledgements

The author would like to acknowledge the support of Martin Hazelton as supervisor during the author's Ph.D. candidature, during which much of this research was carried out. The candidature was supported financially by an Australian Postgraduate Award.

References

Bowman A. W., Azzalini A. (2005). *sm: Kernel Smoothing Methods: Bowman and Azzalini (1997)*. R package version 2.0-14. Ported to R by B. D. Ripley.

Bowman, A. W. (1984). An alternative method of cross-validation for the smoothing of density estimates. *Biometrika*, **71**, 353–360.

Duong, T. and Hazelton, M. L. (2003). Plug-in bandwidth matrices for bivariate kernel density estimation. *Journal of Nonparametric Statistics*, **15**, 17–30.

Duong, T. and Hazelton, M. L. (2005). Cross-validation bandwidth matrices for multivariate kernel density estimation. *Scandinavian Journal of Statistics*, **32**, 485–506.

Hall, P., Marron, J. S., and Park, B. U. (1992). Smoothed cross-validation. *Probability Theory and Related Fields*, **92**, 1–20.

Jones, M. C., Marron, J. S., and Park, B. U. (1991). A simple root n bandwidth selector. *The Annals of Statistics*, **19**, 1919–1932.

Rudemo, M. (1982). Empirical choice of histograms and kernel density estimators. *Scandinavian Journal of Statistics. Theory and Applications*, **9**, 65–78.

Sain, S. R., Baggerly, K. A., and Scott, D. W. (1994). Cross-validation of multivariate densities. *Journal of the American Statistical Association*, **89**, 807–817.

Sheather, S. J. and Jones, M. C. (1991). A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society. Series B. Methodological*, **53**, 683–690.

Simonoff, J. S. (1996). *Smoothing Methods in Statistics*. Springer-Verlag, New York.

Wand, M. P. (2006). *KernSmooth: Functions for kernel smoothing for Wand & Jones (1995)*. R package version 2.22-19. R port by Brian Ripley.

Wand, M. P. and Jones, M. C. (1994). Multivariate plug-in bandwidth selection. *Computational Statistics*, **9**, 97–116.

Wand, M. P. and Jones, M. C. (1995). *Kernel Smoothing*. Chapman & Hall/CRC, London.