Journal of Statistical Software

January 2009, Volume 29, Issue 2. *http://www.jstatsoft.org/*

LogConcDEAD: An R Package for Maximum Likelihood Estimation of a Multivariate Log-Concave Density

Madeleine Cule
University of Cambridge

Robert Gramacy
University of Cambridge

Richard Samworth
University of Cambridge

Abstract

In this article we introduce the R package **LogConcDEAD** (Log-concave density estimation in arbitrary dimensions). Its main function is to compute the nonparametric maximum likelihood estimator of a log-concave density. Functions for plotting, sampling from the density estimate and evaluating the density estimate are provided. All of the functions available in the package are illustrated using simple, reproducible examples with simulated data.

Keywords: log-concave density, multivariate density estimation, visualization, nonparametric statistics.

1. Introduction

1.1. About this document

This document is an introduction to the R package **LogConcDEAD** (log-concave density estimation in arbitrary dimensions). It aims to provide a detailed user guide based on simple, reproducible worked examples. This package is available from the Comprehensive R Archive Network at `http://CRAN.R-project.org/package=LogConcDEAD`.

LogConcDEAD depends on **akima** (Akima *et al.* 2006) for plotting, **MASS** (Venables and Ripley 2002) for some vector operations, and **geometry** (Grasman and Gramacy 2008) for convex hull computation. The package **rgl** (Adler and Murdoch 2007) is recommended for producing graphics.

This document was created using `Sweave` (Leisch 2002) and LaTeX (Lamport 1994) using R

(R Development Core Team 2008). This means that all of the code has been checked by R, and can be reproduced exactly by setting an appropriate seed (as given at the beginning of each example), or tested on different examples by using a different seed.

1.2. Log-concave density estimation

We address the fundamental statistical problem of estimating a probability density function f_0 from independent and identically distributed observations $X_1, \ldots, X_n$ taking values in $\mathbb{R}^d$.

If a suitable parametric model is available, a common method is to use maximum likelihood to estimate the parameters of the model. Otherwise, a standard nonparametric approach is based on kernel density estimation (Wand and Jones 1995), which has been implemented in the R function `density`. In common with many nonparametric methods, kernel density estimation requires the careful specification of a smoothing parameter. For multivariate data, the smoothing parameter is a bandwidth matrix with up to $\frac{1}{2}d(d+1)$ entries to choose, meaning that this method can be especially difficult to apply in practice.

An alternative to kernel density estimation or other estimation techniques based on smoothing (all of which require the selection of a smoothing parameter, which is nontrivial especially in the multivariate case) is to impose some qualitative shape restrictions on the density. If the shape restrictions are suitable, there is enough structure to guarantee the existence of a unique and fully automatic maximum likelihood estimate, even though the class of densities may be infinite-dimensional. This therefore avoids both the restrictions of a parametric model and the difficulty of bandwidth selection in kernel density estimation. The price is some restriction on the shape of the density. However, these restrictions are less severe than those imposed by a parametric model.

Shape-constrained maximum likelihood dates back to Grenander (1956), who treated monotone densities in the context of mortality data. Recently there has been considerable interest in alternative shape constraints, including convexity, k-monotonicity and log-concavity (Groeneboom *et al.* 2001; Dümbgen and Rufibach 2008; Balabdaoui and Wellner 2007). However, these works have all focused on the case of univariate data.

Log-concave densities

A function $g\colon \mathbb{R}^d \to [-\infty, \infty)$ is concave if

$$g(\lambda x + (1-\lambda)y) \geq \lambda g(x) + (1-\lambda)g(y)$$

for all $x, y \in \mathbb{R}^d$ and $\lambda \in (0,1)$. This corresponds to what Rockafellar (1997) calls a proper concave function. We say a probability density function f is log-concave if $\log f$ is a concave function. Several common parametric families of univariate densities are log-concave, such as Gaussian, logistic and Gumbel densities, as well as Weibull, Gamma and Beta densities for certain parameter values (An 1998). In fact, Cule *et al.* (2008) showed that even though the class of multivariate log-concave densities is large (infinite-dimensional), it still retains some of the simple and attractive properties of the class of Gaussian densities.

1-dimensional log-concave density estimation via maximum likelihood is discussed in Dümbgen and Rufibach (2008); computational aspects are treated in Rufibach (2007). It is in the multivariate case, however, where kernel density estimation is more difficult and parametric models less obvious, where a log-concave model may be most useful.

Theoretical and computational aspects of multivariate log-concave density estimation are treated in Cule *et al.* (2008). In particular, it is proved that if $Y_1, \ldots, Y_m$ are (distinct) independent and identically distributed observations from a distribution with log-concave density f_0 on $\mathbb{R}^d$, then (with probability 1) there is a unique log-concave density $\widehat{f}_m$ satisfying

$$\widehat{f}_m = \operatorname*{argmax}_{f \in \mathcal{F}} \frac{1}{m} \sum_{i=1}^{m} \log f(Y_i), \tag{1}$$

where $\mathcal{F}$ is the class of all log-concave densities on $\mathbb{R}^d$. Further, it is shown that this infinite-dimensional maximization problem can be reduced to that of maximizing over functions of the form $\bar{h}_y$ for some $y = (y_1, \ldots, y_m) \in \mathbb{R}^m$, where

$$\bar{h}_y(x) = \inf\{h(x) \colon h \text{ is concave, } h(Y_i) \geq y_i, \ i = 1, \ldots, m\}. \tag{2}$$

As discussed in Cule *et al.* (2008), we may think of $\bar{h}_y$ as the function obtained by placing a pole of height y_i at X_i and stretching a rubber sheet over the top of the poles.

Therefore, to completely specify the maximum likelihood estimator, we need only specify a suitable vector $\widehat{y} \in \mathbb{R}^m$, as this defines the entire function $\bar{h}_{\widehat{y}}$. A main feature of the **LogConcDEAD** package is that it provides an iterative algorithm for finding such an appropriate vector $\widehat{y}$.

From our knowledge of the structure of functions of the form (2), we may deduce some additional properties of $\widehat{f}_m$. It is zero outside the convex hull of the data, and strictly positive inside the convex hull. Moreover, we can find a triangulation of the convex hull into simplices (triangles when $d = 2$, tetrahedra when $d = 3$, and so on) such that $\log \widehat{f}_m$ is affine on each simplex (Rockafellar 1997).

In practice our observations will be made only to a finite precision, so the observations will not necessarily be distinct. However, the same method of proof shows that, more generally, if $X_1, \ldots, X_n$ are distinct points in $\mathbb{R}^d$ and $w_1, \ldots, w_n$ are strictly positive weights satisfying $\sum_{i=1}^{n} w_i = 1$, then there is a unique log-concave density $\widehat{f}_n$, which is of the form $\widehat{f}_n = \exp(\bar{h}_y)$ for some $y \in \mathbb{R}^n$, and which satisfies

$$\widehat{f}_n = \operatorname*{argmax}_{f \in \mathcal{F}} \sum_{i=1}^{n} w_i \log f(X_i). \tag{3}$$

The default case $w_i = \frac{1}{n}$ corresponds to the situation described above, and is appropriate for most situations. However, the generalization (3) obtained by allowing $w_i \neq \frac{1}{n}$ allows us to extend to binned observations. In more detail, if $Y_1, \ldots, Y_m$ are independent and identically distributed according to a density f_0, and distinct binned values $X_1, \ldots, X_n$ are observed, we may construct a maximum likelihood problem of the form given in (3), setting

$$w_i = \frac{\#\text{ of times value } X_i \text{ is observed}}{m}$$

and

$$\widehat{f}_n = \operatorname*{argmax}_{f \in \mathcal{F}} \sum_{i=1}^{n} w_i \log f(X_i).$$

This generalization may also be used for a multivariate version of a log-concave EM algorithm (Chang and Walther 2008, also discussed in Cule *et al.* 2008).

1.3. Outline of the remainder of this document

In Section 2, we outline the algorithm used to compute the maximum likelihood estimator, including various parameters used in the computation. This is essentially an adaptation of Shor's r-algorithm (Shor 1985, implemented as **SolvOpt** by Kappel and Kuntsevich 2000), and depends on the **Quickhull** algorithm for computing convex hulls (Barber *et al.* 1996). This section may be skipped on first reading.

In Section 3, we demonstrate the main features of the package through four simple examples (one with $d = 1$, two with $d = 2$ and one with $d = 3$). This section includes a description of all of the parameters used, as well as the output structures. We also introduce the plotting functions available, as well as functions for sampling from the density estimate and for evaluating the density at a particular point.

2. Algorithm

2.1. Introduction

Recall that the maximum likelihood estimator $\widehat{f}_n$ of f_0 may be completely specified by its values at the observations $X_1, \ldots, X_n$. Writing C_n for the convex hull of the data, Cule *et al.* (2008) showed that the problem of computing the estimator may be rephrased as one of finding

$$\operatorname*{argmin}_{y \in \mathbb{R}^n} \sigma(y) = -\sum_{i=1}^{n} w_i y_i + \int_{C_n} \exp\{\bar{h}_y(x)\}\, dx$$

for suitable chosen weights w_i, where

$$\bar{h}_y(x) = \inf\{h(x) : h \text{ is concave}, h(X_i) \geq y_i, i = 1, \ldots, n\}.$$

The function σ is convex, but not differentiable, so standard gradient-based convex optimization techniques such as Newton's method are not suitable. Nevertheless, the notion of a subgradient is still valid: a subgradient at y of σ is any direction which defines a supporting hyperplane to σ at y. Shor (1985) developed a theory of subgradient methods for handling convex, non-differentiable optimization problems. The r-algorithm, described in Shor (1985, Chapter 3) and implemented as **SolvOpt** in C by Kappel and Kuntsevich (2000), was found to work particularly well in practice. A main feature of the **LogConcDEAD** package is an implementation of an adaptation of this r-algorithm for the particular problem encountered in log-concave density estimation.

2.2. Shor's r-algorithm

Our adaptation of Shor's r-algorithm produces a sequence (y^t) with the property that

$$\sigma(y^t) \to \min_{y \in \mathbb{R}^n} \sigma(y)$$

as $t \to \infty$. At each iteration, the algorithm requires the evaluation $\sigma(y^t)$, and the subgradient at y^t, denoted $\partial\sigma(y^t)$, which determines the direction of the move to the next term y^{t+1} in the sequence.

Exact expressions for $\sigma(y^t)$ and $\partial\sigma(y^t)$ are provided in Cule *et al.* (2008). In practice, their computation requires the evaluation of convex hulls and triangulations of certain finite sets of points. This can be done in a fast and robust way via the **Quickhull** algorithm (Barber *et al.* 1996), available in R through the **geometry** package (Grasman and Gramacy 2008). Due to the presence of some removable singularities in the expressions for $\sigma(y^t)$ and $\partial\sigma(y^t)$, it is computationally more stable to use a Taylor approximation to the true values for certain values of y^t (Cule and Dümbgen 2008). The values for which a Taylor expansion (rather than direct evaluation) is used may be controlled by the argument `Jtol` to the **LogConcDEAD** function `mlelcd`. By default this is 10^{-3}; altering this parameter is not recommended.

Several parameters may be used to control the r-algorithm as detailed by Kappel and Kuntsevich (2000). In the function `mlelcd`, they may be controlled by the user via the arguments `stepscale1`, `stepscale2`, `stepscale3`, `stepscale4` and `desiredsize`. For a detailed description of these parameters, as well as of this implementation of the r-algorithm, see Kappel and Kuntsevich (2000).

Stopping criteria

The implementation of the r-algorithm used in the main function `mlelcd` terminates after the $(t+1)$th iteration if each of the following conditions holds:

$$|y_i^{t+1} - y_i^t| \leq \delta |y_i^t| \text{ for } i = 1, \ldots, n \tag{4}$$

$$|\sigma(y^{t+1}) - \sigma(y^t)| \leq \epsilon |\sigma(y^t)| \tag{5}$$

$$\left| \int_{C_n} \exp\{\bar{h}_{y^t}(x)\}\, dx - 1 \right| \leq \eta \tag{6}$$

for some small tolerances δ, ϵ and η.

(4) and (5) are the criteria suggested by Kappel and Kuntsevich (2000); (6) is based on the observation that the maximum likelihood estimator is density (Cule *et al.* 2008). By default, these values are $\delta = 10^{-4}$, $\epsilon = 10^{-8}$ and $\eta = 10^{-4}$, but they may be modified by the user as required, using the parameters `ytol`, `sigmatol` and `integraltol` respectively. The default parameters have been found to work well and it is not recommended to alter them.

3. Usage

In this section we illustrate the functions available in **LogConcDEAD** through several simple simulated data examples. These functions include `mlelcd`, which computes the maximum likelihood estimator, as well as graphics facilities and the function `rlcd` for sampling from the fitted density.

3.1. Example 1: 1-d data

For 1-dimensional data, the alternative active set algorithm from **logcondens** (Rufibach and Dümbgen 2006; Dümbgen *et al.* 2007) may be used to compute the log-concave maximum likelihood estimator. In this section we will compare the output of the two procedures.

For this example, we will use 200 points from a Gamma$(2, 1)$ distribution. The seed has been set (to 1), so this example can be reproduced exactly; you may also like to try with a different seed.

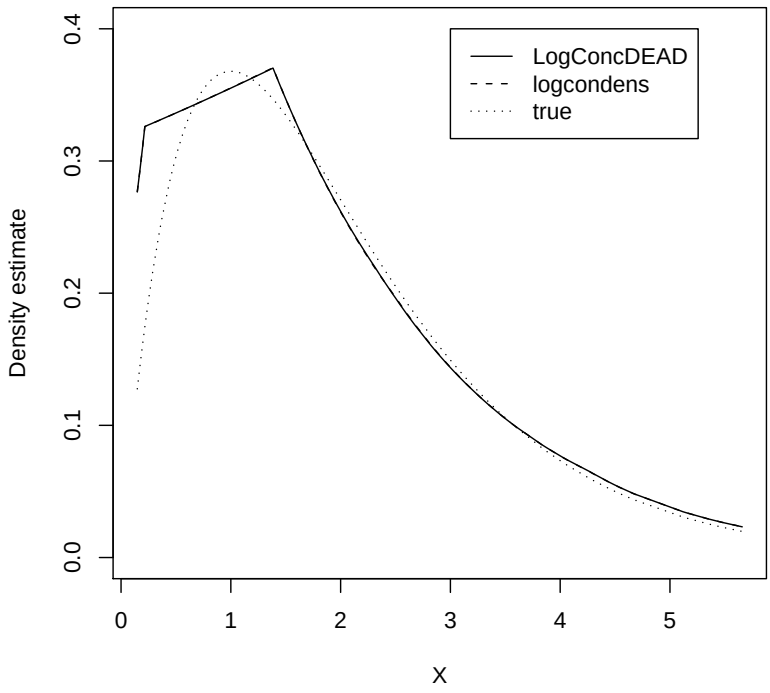

Figure 1: Density estimates (and true density) based on 200 i.i.d. observations from a Gamma(2,1) distribution.

```
R> library("logcondens")
R> set.seed(1)
R> n <- 200
R> x <- sort(rgamma(n, shape = 2))
R> out1 <- activeSetLogCon(x)
R> out2 <- mlelcd(x)
```

We can see from Figure 1 that, as expected, **logcondens** and **LogConcDEAD** produce the same output. This figure is produced using the following code:

```
R> ylim <- c(0, 0.4)
R> lgdtxt <- c("LogConcDEAD", "logcondens", "true")
R> lgdlty <- c(1, 2, 3)
R> plot(out2, ylim = ylim, lty = 1)
R> lines(x, exp(out1$phi), lty = 2)
R> lines(x, x * exp(-x), lty = 3)
R> legend(x = 3, y = 0.4, lgdtxt, lty = lgdlty)
```

Figure 2 also illustrates the structure of the log-concave maximum likelihood estimator: its logarithm is piecewise linear with changes of slope only at observation points. This figure is produced using the following code:

```
R> ylim <- c(-4, -1)
R> lgdtxt <- c("LogConcDEAD", "logcondens", "true")
```

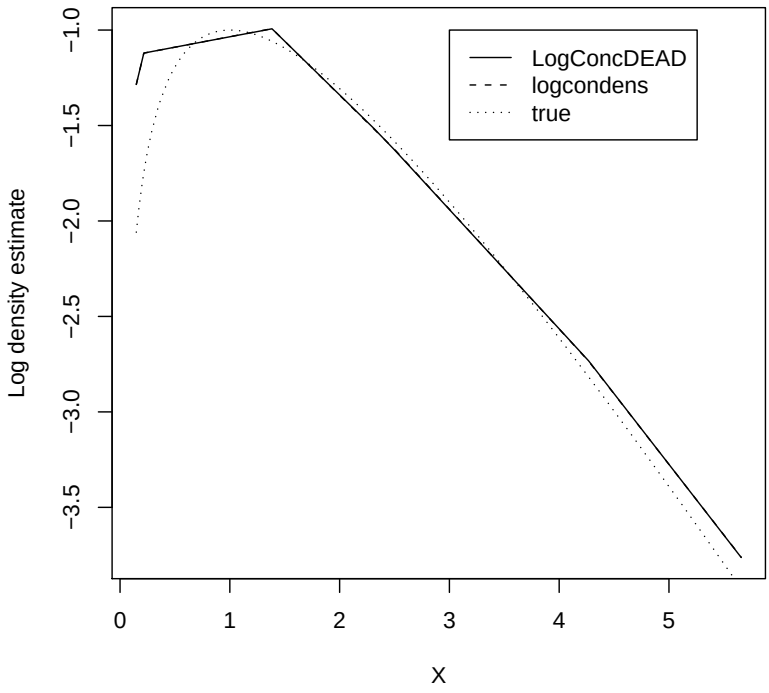

Figure 2: Log of density estimate (and true log-density) based on 200 i.i.d. observations from a Gamma(2,1) distribution.

```
R> lgdlty <- c(1, 2, 3)
R> plot(out2, uselog = TRUE, lty = 1)
R> lines(x, out1$phi, lty = 2)
R> lines(x, log(x) - x, lty = 3)
R> legend(x = 3, y = -1, lgdtxt, lty = lgdlty)
```

3.2. Example 2: 2-d normal data

For this section, we will generate 500 points from a bivariate normal distribution with independent components. Again, we have set the seed (to 22) for reproducibility.

```
R> set.seed(22)
R> d <- 2
R> n <- 500
R> x <- matrix(rnorm(n * d), ncol = d)
```

Basic usage

The basic command in this package is `mlelcd`, which computes the log-concave maximum likelihood estimate $\widehat{f}_n$. The `verbose` option controls the diagnostic output, which will be described in more detail below.

```
R> out <- mlelcd(x, verbose = 200)

Iter # ... Function Val ... Step Value ... Integral ... Grad Norm
   200          3.84062      0.0202458     0.989243      0.038331
Iter # ... Function Val ... Step Value ... Integral ... Grad Norm
   400          3.83207      0.0183776      1.00047      0.038719
Iter # ... Function Val ... Step Value ... Integral ... Grad Norm
   600          3.80796      0.0118178      1.00164      0.027426
Iter # ... Function Val ... Step Value ... Integral ... Grad Norm
   800          3.80767    0.000289599      1.00121      0.026765
Iter # ... Function Val ... Step Value ... Integral ... Grad Norm
  1000          3.80747    0.000715941      1.00041      0.028312
Iter # ... Function Val ... Step Value ... Integral ... Grad Norm
  1200          3.80744    0.000548852      1.00018      0.030912
SolvOpt: Normal termination.
```

The default `print` statement shows the value of the logarithm of the maximum likelihood estimator at the data points, the number of iterations of the subgradient algorithm required, and the total number of function evaluations required to reach convergence.

In the next two subsections, we will describe the input and output in more detail.

Input

The only input required is an $n \times d$ matrix of data points. One dimensional (vector) input will be converted to a matrix. Optionally a vector of weights `w`, corresponding to $(w_1, \ldots, w_n)$ in (3), may be specified. By default this is

$$\left(\frac{1}{n}, \ldots, \frac{1}{n}\right),$$

which is appropriate for independent and identically distributed observations.

A starting value `y` may be specified for the vector $(y_1, \ldots, y_n)$; by default a kernel density estimate (using a normal kernel and a diagonal bandwidth selected using a normal scale rule) is used. This is performed using the (internal) function `initialy`.

The parameter `verbose` controls the degree of diagnostic information provided by **SolvOpt**. The default value, -1, prints nothing. The value 0 prints warning messages only. If the value is $m > 0$, diagnostic information is printed every mth iteration. The printed information summarises the progress of the algorithm, displaying the iteration number, current value of the objective function, (Euclidean) length of the last step taken, current value of $\int \exp\{\bar{h}_y(x)\}\, dx$ and (Euclidean) length of the subgradient. The last column is motivated by the fact that 0 is a subgradient only at the minimum of σ (Rockafellar 1997, Chapter 27), and so for smooth functions a small value of the subgradient may be used as a stopping criterion. For nonsmooth functions, we may be close to the minimum even if this value is relatively large, so only the middle three columns form the basis of our stopping criteria, as described in Section 2.2.1.

The remaining optional arguments are generic parameters of the r-algorithm, and have already been discussed in Section 2.

Output

The output is an object of class `"LogConcDEAD"`, which has the following elements:

```
R> names(out)

 [1] "x"                   "w"                   "logMLE"
 [4] "NumberOfEvaluations" "MinSigma"            "b"
 [7] "beta"                "triang"              "verts"
[10] "vertsoffset"         "chull"               "outnorm"
[13] "outoffset"
```

The first two components `x` and `w` give the input data. The component `logMLE` specifies the logarithm of the maximum likelihood estimator, via its values at the observation points. In this example the first 5 elements are shown, corresponding to the first 5 rows of the data matrix `x`.

```
R> out$logMLE[1:5]

[1] -2.416011 -5.786652 -2.382350 -2.081424 -2.484434
```

As was mentioned in Sections 1 and 2, there is a triangulation of $C_n = \mathrm{conv}(X_1, \ldots, X_n)$, the convex hull of the data, such that $\log \widehat{f}_n$ is affine on each simplex in the triangulation. Each simplex in the triangulation is the convex hull of a subset of $\{X_1, \ldots, X_n\}$ of size $d+1$. Thus the simplices in the triangulation may be indexed by a finite set J of $(d+1)$-tuples, which are available via

```
R> out$triang[1:5, ]

     [,1] [,2] [,3]
[1,]  287  115  239
[2,]   15  483  236
[3,]  369  115  171
[4,]  369  287  115
[5,]  393  115  171
```

For each $j \in J$, there is a corresponding vector $b_j \in \mathbb{R}^d$ and $\beta_j \in \mathbb{R}$, which define the affine function which coincides with $\log \widehat{f}_n$ on the jth simplex in the triangulation. These values b_j and β_j are available in

```
R> out$b[1:5, ]

         [,1]      [,2]
[1,] 3.057233 -1.668416
[2,] 1.468925  6.372443
[3,] 2.506461 -1.238754
[4,] 2.751072 -1.206699
[5,] 2.506459 -1.238757
```

```
R> out$beta[1:5]

[1]  -3.913909 -10.421228  -2.120515  -2.698055  -2.120513
```

(In all of the above cases, only the first 5 elements are shown.) As discussed in Cule *et al.* (2008), for each $j \in J$ we may find a matrix A_j and a vector α_j such that the map $w \mapsto A_j w + \alpha_j$ maps the unit simplex in $\mathbb{R}^d$ to the jth simplex in the triangulation. The inverse of this map, $x \mapsto A_j^{-1}x - A_j^{-1}\alpha_j$, is required for easy evaluation of the density at a point, and for plotting. The matrix A_j^{-1} is available in `out$verts` and $A_j^{-1}\alpha_j$ is available in `out$vertsoffset`.

The `"LogConcDEAD"` object also provides some diagnostic information on the execution of the **SolvOpt** routine: the number of iterations required, the number of function evaluations needed, the number of subgradient evaluations required (in a vector `NumberOfEvaluations`), and the minimum value of the objective function σ attained (`MinSigma`).

```
R> out$NumberOfEvaluations

[1] 1373 4320 1374

R> out$MinSigma

[1] 3.807418
```

The indices of simplices in the convex hull C_n are available:

```
R> out$chull[1:5, ]

     [,1] [,2]
[1,]  239  287
[2,]  239   84
[3,]   46  259
[4,]  203  287
[5,]    2  259
```

In addition, an outward-pointing normal vector for each face of the convex hull C_n and an offset point (lying on the face of the convex hull) may be obtained.

```
R> out$outnorm[1:5, ]

           [,1]        [,2]
[1,] -0.8485752  0.52907483
[2,] -0.0767366  0.99705140
[3,]  0.9996818  0.02522528
[4,] -0.8588526 -0.51222283
[5,]  0.9264791 -0.37634623

R> out$outoffset[1:5, ]
```

```
          [,1]      [,2]
[1,] -1.930875  2.986373
[2,] -1.930875  2.986373
[3,]  3.253349  2.471396
[4,] -2.407876 -1.909542
[5,]  2.485184 -1.763713
```

This information may be used to test whether or not a point lies in C_n, as $x \in C_n$ if and only if $p^T(x - q) \leq 0$ for every face of the convex hull, where p denotes an outward normal and q an offset point.

When $d = 1$, the convex hull consists simply of the minimum and maximum of the data points, and `out$outnorm` and `out$outoffset` are `NULL`, although `out$chull` still takes on the appropriate values.

3.3. Graphics

Various aspects of the log-concave maximum likelihood estimator can be plotted using the `plot` command, applied to an object of class `"LogConcDEAD"`.

The plots are based on interpolation over a grid, which can be somewhat time-consuming. As several will be done here, we can save the results of the interpolation separately, using the function `interplcd`, and use it to make several plots. The number of grid points may be specified using the parameter `gridlen`. By default, `gridlen = 100`, which is suitable for most plots.

Where relevant, the colors were obtained by a call to `heat_hcl` in the package **colorspace** (Ihaka *et al.* 2008), following the recommendation of Zeileis *et al.* (2008).

```
R> g <- interplcd(out, gridlen = 200)
R> g1 <- interpmarglcd(out, marg = 1)
R> g2 <- interpmarglcd(out, marg = 2)
```

The plots in Figure 3 show a contour plot of the estimator and a contour plot of its logarithm for 500 points in 2 dimensions. Note that the contours of log-concave densities enclose convex regions. This figure is produced using

```
R> par(mfrow = c(1, 2), pty = "s", cex = 0.7)
R> plot(out, g = g, addp = FALSE, asp = 1)
R> plot(out, g = g, uselog = TRUE, addp = FALSE, asp = 1)
```

If $d > 1$, we can plot 1-dimensional marginals by setting the `marg` parameter. Note that the marginal densities of a log-concave density are log-concave (discussed in Cule *et al.* 2008, as a consequence of the theory of Prékopa 1973). This is illustrated by Figure 4 using the following code:

```
R> par(mfrow = c(1, 2), pty = "s", cex = 0.7)
R> plot(out, marg = 1, g.marg = g1)
R> plot(out, marg = 2, g.marg = g2)
```

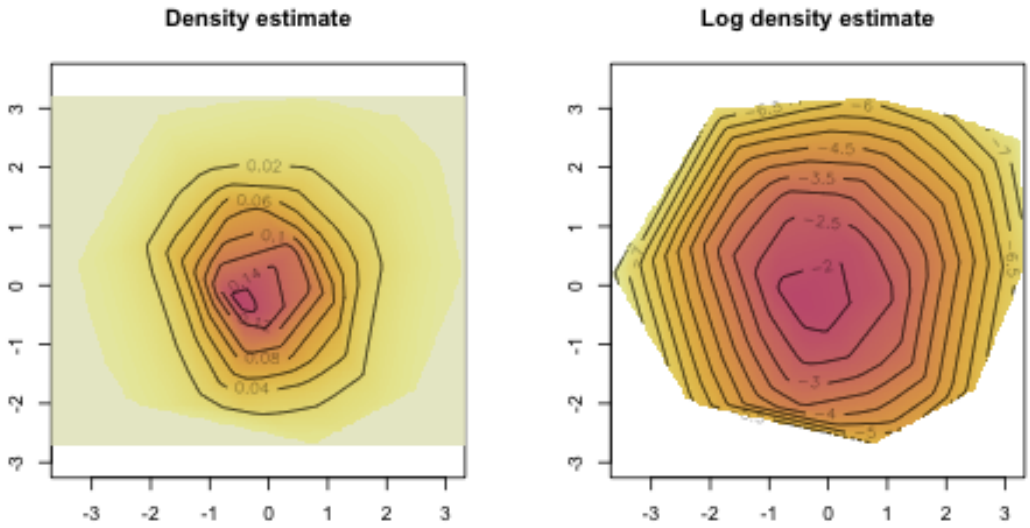

Figure 3: Plots based on 500 points from a standard bivariate normal distribution.

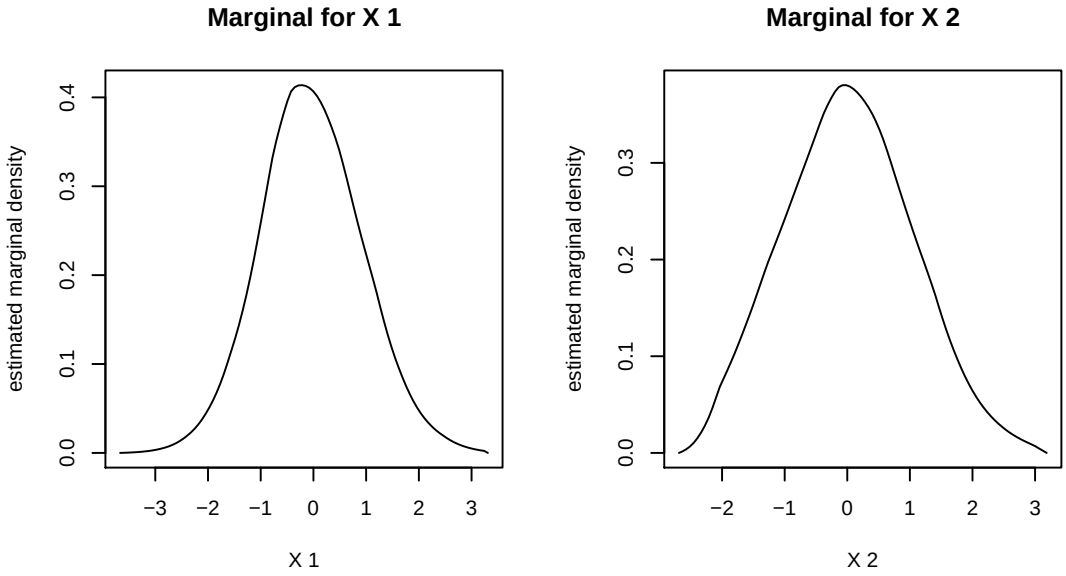

Figure 4: Plots of estimated marginal densities based on 500 points from a standard bivariate normal distribution.

The plot type is controlled by the argument `type`, which may take the values `"p"` (a perspective plot), `"i"`, `"c"` or `"ic"` (colour maps, contours or both), or `"r"` (a 3d plot using the **rgl** package Adler and Murdoch 2007). The default plot type is `"ic"`.

The **rgl** package allows user interaction with the plot (e.g., the plot can be rotated using the mouse and viewed from different angles). Although we are unable to demonstrate this feature on paper, Figure 5 shows the type of output produced by **rgl**, using the following code:

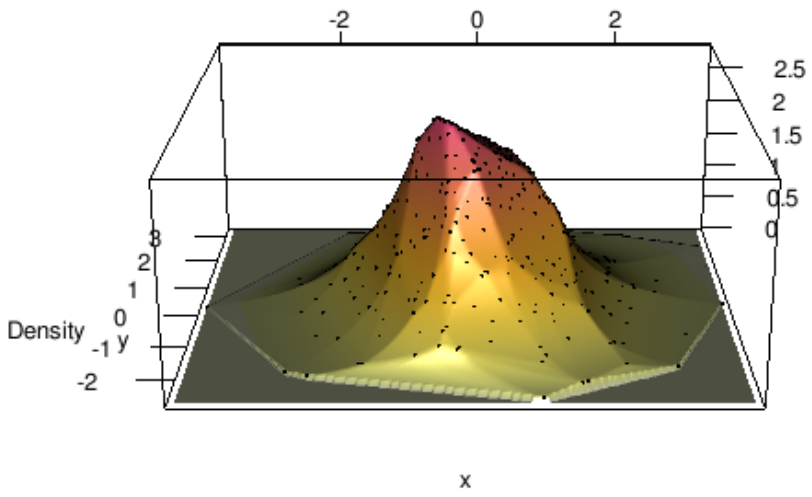

Figure 5: **rgl** output for Example 2.

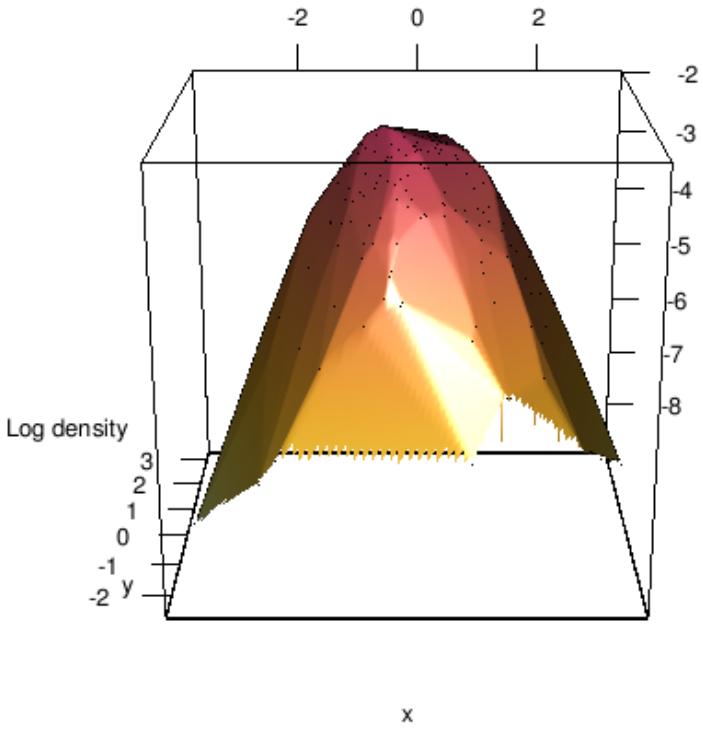

Figure 6: **rgl** output for Example 2 with `uselog = TRUE`.

```
R> plot(out, g = g, type = "r")
```

Figure 6 shows the output produced by setting `uselog = TRUE` to plot on the log scale. Here we can clearly see the structure of the log-concave density estimate. This is produced using the command

```
R> plot(out, g = g, type = "r", uselog = TRUE)
```

3.4. Other functions

In this section we will describe the use of the additional functions `rlcd` and `dlcd`.

Sampling from the MLE

Suppose we wish to estimate a functional of the form $\theta(f) = \int g(x)f(x)\,dx$, for example the mean or other moments, the differential entropy $-\int f(x)\log f(x)\,dx$, etc. Once we have obtained a density estimate $\widehat{f}_n$, such as the log-concave maximum likelihood estimator, we may use it as the basis for a plug-in estimate $\widehat{\theta}_n = \int g(x)\widehat{f}_n(x)\,dx$, which may be approximated using a simple Monte Carlo procedure even if an analytic expression is not readily available. In more detail, we generate a sample $Z_1, \ldots, Z_N$ drawn from $\widehat{f}_n$, and approximate $\widehat{\theta}_n$ by

$$\widetilde{\theta}_n = \frac{1}{N}\sum_{j=1}^{N} g(Z_j).$$

This requires the ability to sample from $\widehat{f}_n$, which may be achieved given an object of class `"LogConcDEAD"` as follows:

```
R> nsamp <- 1000
R> mysamp <- rlcd(nsamp, out)
```

Details of the function `rlcd`, which uses a straightforward rejection sampling scheme, are given in Cule *et al.* (2008).

Once we have a random sample, plug-in estimation of various functionals is straightforward.

```
R> apply(mysamp, 2, mean)

[1] -0.05823016 -0.03222685

R> cov(mysamp)

            [,1]       [,2]
[1,] 0.93223724 0.03134590
[2,] 0.03134590 1.10437356
```

Evaluation of fitted density

We may evaluate the fitted density at a point or matrix of points such as

```
R> m <- 10
R> mypoints <- 1.5 * matrix(rnorm(m * d), ncol = d)
```

using the command

```
R> dlcd(mypoints, out)
```

```
[1] 0.078864298 0.134037006 0.003302070 0.000000000 0.107737481
[6] 0.071336837 0.033358001 0.020749033 0.031100878 0.006285303
```

Note that, as expected, the density estimate is zero for points outside the convex hull of the original data.

The `dlcd` function may be used in conjunction with `rlcd` to estimate more complicated functionals such as a $100(1-\alpha)\%$ highest density region, defined by Hyndman (1996) as $R_\alpha = \{x \in \mathbb{R}^d : f(x) \geq f_\alpha\}$, where f_α is the largest constant such that $\int_{R_\alpha} f(x)\,dx \geq 1-\alpha$. Using the algorithm outlined in Hyndman (1996, Section 3.2), it is straightforward to approximate f_α as follows:

```
R> myval <- sort(dlcd(mysamp, out))
R> alpha <- c(0.25, 0.5, 0.75)
R> myval[(1 - alpha) * nsamp]

[1] 0.12070302 0.07863713 0.03931380
```

3.5. Example 3: 2-d binned data

In this section, we demonstrate the use of **LogConcDEAD** with binned data. The seed here has been set to 333 for reproducibility; you may wish to try these examples with other seeds. We generate some data from a normal distribution with correlation using the package **mvtnorm** (Genz *et al.* 2008).

```
R> library("mvtnorm")
R> set.seed(333)
R> sigma <- matrix(c(1, 0.2, 0.2, 1), nrow = 2)
R> d <- 2
R> n <- 500
R> y <- rmvnorm(n, sigma = sigma)
R> xall <- round(y, digits = 1)
```

The matrix `xall` therefore contains 500 observations, rounded to 1 decimal place; there are in total 411 distinct observations. In order to compute an appropriate log-likelihood, we will use the function `getweights` to extract a matrix of distinct observations and a vector of weights for use in `mlelcd`. The result of this has two parts: a matrix `x` consisting of the distinct observations, and a vector `w` of weights. We may also use `interplcd` as before to evaluate the estimator on a grid for plotting purposes.

```
R> tmpw <- getweights(xall)
R> outw <- mlelcd(tmpw$x, w = tmpw$w)
R> gw <- interplcd(outw, gridlen = 200)
```

In Figure 7 we plot density and log-density estimates using a contour plot as before. In contrast to the examples in Figure 3, we have `addp=TRUE` (the default), which superposes the observation points on the plots, and `drawlabels=FALSE`, which suppresses the contour labels. The code to do this is

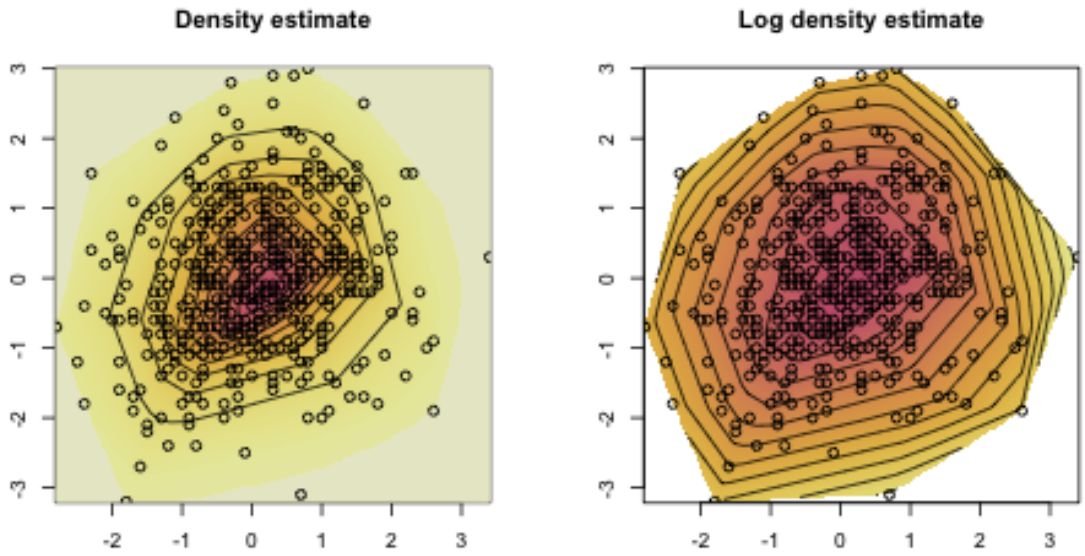

Figure 7: Density and log density estimate based on 500 points from a bivariate normal distribution in two dimensions, truncated to 1 decimal place (Example 3).

```
R> par(mfrow = c(1, 2), pty = "s", cex = 0.7)
R> plot(outw, g = gw, asp = 1, drawlabels = FALSE)
R> plot(outw, g = gw, uselog = TRUE, asp = 1, drawlabels = FALSE)
```

3.6. Example 4: Higher-dimensional data

In our final example we illustrate the use of the log-concave density estimate for higher-dimensional data. The seed has been set to 4444. The log-concave maximum likelihood estimator is defined and may be computed and evaluated in exactly the same way as the 2-dimensional examples in Sections 3.2 and 3.5. This estimate will be based on 500 points.

```
R> set.seed(4444)
R> d <- 3
R> n <- 500
R> x <- matrix(rgamma(n * d, shape = 2), ncol = d)
R> out3 <- mlelcd(x)
```

The function `dmarglcd` may be used to evaluate the marginal estimate, setting the parameter `marg` appropriately. Note that, as before, the estimate is 0 outside the convex hull of the observed data.

```
R> mypoints <- c(0, 2, 4)
R> dmarglcd(mypoints, out3, marg = 1)

[1] 0.00000000 0.28978620 0.07632828
```

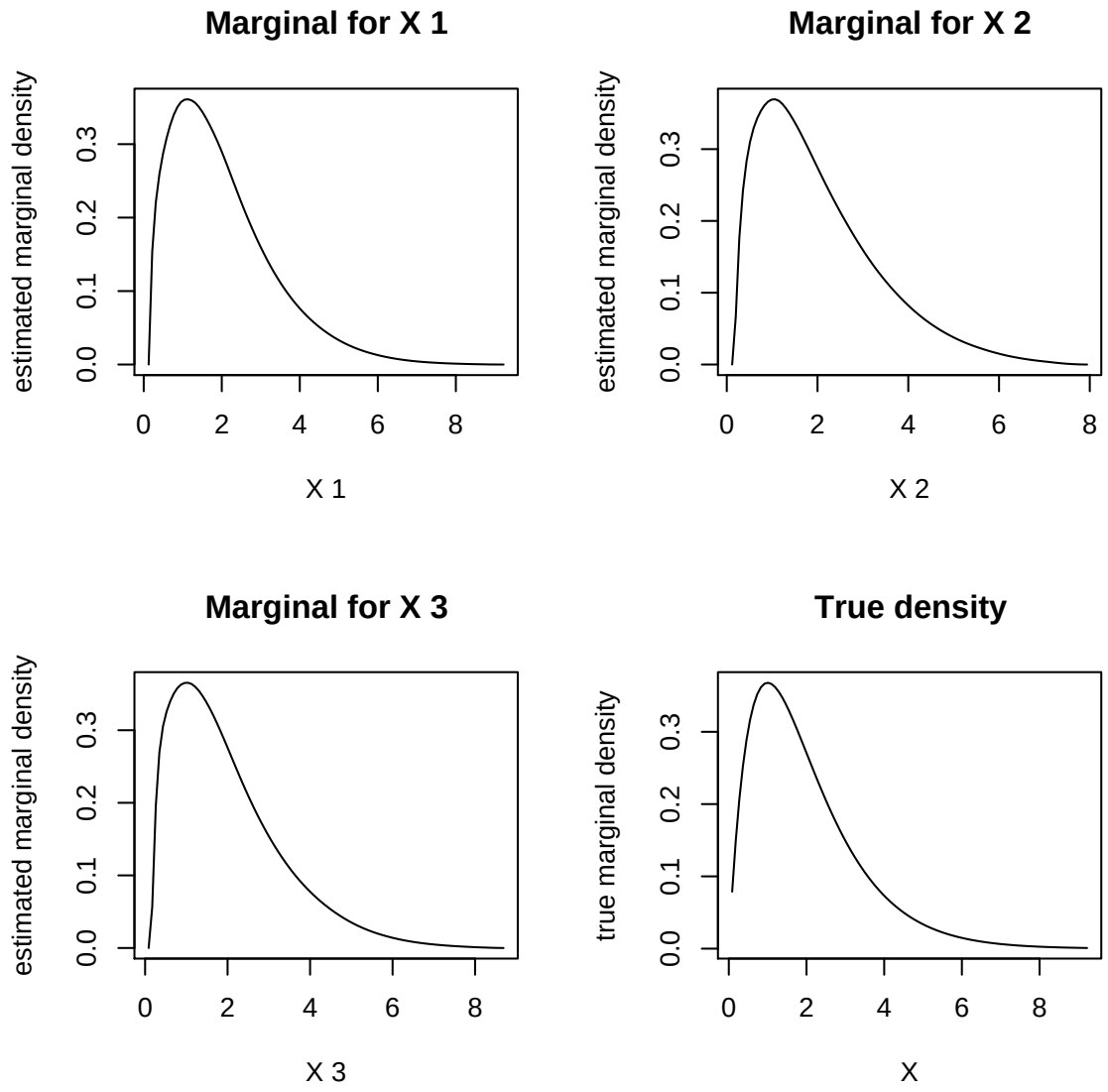

Figure 8: Marginal density estimates for 3-dimensional data (based on 500 points).

1-dimensional marginal distributions may be plotted easily, by setting the `marg` parameter to the appropriate margin as shown in Figure 8 using the following:

```
R> par(mfrow = c(2, 2), cex = 0.8)
R> plot(out3, marg = 1)
R> plot(out3, marg = 2)
R> plot(out3, marg = 3)
R> tmp <- seq(min(out3$x), max(out3$x), len = 100)
R> plot(tmp, dgamma(tmp, shape = 2), type = "l", xlab = "X",
+     ylab = "true marginal density")
R> title(main = "True density")
```

References

Adler D, Murdoch D (2007). ***rgl**: 3D Visualization Device System (**OpenGL**)*. R package version 0.75, URL http://CRAN.R-project.org/package=rgl.

Akima H, Gebhardt A, Petzoldt T, Maechler M (2006). ***akima**: Interpolation of Irregularly Spaced Data*. R package version 0.5-1, URL http://CRAN.R-project.org/package=akima.

An MY (1998). "Logconcavity Versus Logconvexity: A Complete Characterization." *Journal of Economic Theory*, **80**, 350–369.

Balabdaoui F, Wellner JA (2007). "Estimation of a k-Monotone Density: Limiting Distribution Theory and the Spline Connection." *The Annals of Statistics*, **35**, 2536–2564.

Barber CB, Dobkin DP, Huhdanpaa H (1996). "The Quickhull Algorithm for Convex Hulls." *ACM Transactions on Mathematical Software*, **22**, 469–483. URL http://www.qhull.org/.

Chang G, Walther G (2008). "Clustering with Mixtures of Log-Concave Distributions." *Computational Statistics & Data Analysis*. doi:10.1016/j.csda.2007.01.008. Forthcoming.

Cule ML, Dümbgen L (2008). "On an Auxiliary Function for Log-Density Estimation." *Technical Report 71*, Universität Bern. URL http://arxiv.org/abs/0807.4719/.

Cule ML, Samworth RJ, Stewart MI (2008). "Maximum Likelihood Estimation of a Multidimensional Log-Concave Density." Submitted, URL http://arxiv.org/abs/0804.3989/.

Dümbgen L, Hüsler A, Rufibach K (2007). "Active Set and EM Algorithms for Log-Concave Densities Based on Complete and Censored Data." *Technical report*, Universität Bern. URL http://arxiv.org/abs/0709.0334/.

Dümbgen L, Rufibach K (2008). "Maximum Likelihood Estimation of a Log-Concave Density: Basic Properties and Uniform Consistency." *Bernoulli*. Forthcoming.

Genz A, Bretz F, Miwa T, Mi X, Leisch F, Scheipl F, Hothorn T (2008). ***mvtnorm**: Multivariate Normal and t Distributions*. R package version 0.9-3, URL http://CRAN.R-project.org/package=mvtnorm.

Grasman R, Gramacy RB (2008). ***geometry**: Mesh Generation and Surface Tesselation*. R package version 0.1, URL http://CRAN.R-project.org/package=geometry.

Grenander U (1956). "On the Theory of Mortality Measurement II." *Skandinavisk Aktuarietidskrift*, **39**, 125–153.

Groeneboom P, Jongbloed G, Wellner JA (2001). "Estimation of a Convex Function: Characterizations and Asymptotic Theory." *The Annals of Statistics*, **29**, 1653–1698.

Hyndman RJ (1996). "Computing and Graphing Highest Density Regions." *The American Statistician*, **50**, 120–126.

Ihaka R, Murrell P, Hornik K, Zeileis A (2008). ***colorspace**: Color Space Manipulation*. R package version 1.0-0, URL http://CRAN.R-project.org/package=colorspace.

Kappel F, Kuntsevich A (2000). "An Implementation of Shor's r-Algorithm." *Computational Optimization and Applications*, **15**, 193–205.

Lamport L (1994). *LaTeX: A Document Preparation System*. 2nd edition. Addison-Wesley, Reading, Massachusetts.

Leisch F (2002). "Dynamic Generation of Statistical Reports Using Literate Data Analysis." In W Härdle, B Rönz (eds.), "COMPSTAT 2002 – Proceedings in Computational Statistics," pp. 575–580. Physica-Verlag, Heidelberg.

Prékopa A (1973). "On Logarithmically Concave Measures and Functions." *Acta Scientarium Mathematicarum*, **34**, 335–343.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Rockafellar RT (1997). *Convex Analysis*. Princeton University Press, Princeton, New Jersey.

Rufibach K (2007). "Computing Maximum Likelihood Estimators of a Log-Concave Density Function." *Journal of Statistical Computation and Simulation*, **77**, 561–574.

Rufibach K, Dümbgen L (2006). ***logcondens**: Estimate a Log-Concave Probability Density from i.i.d. Observations*. R package version 1.3.2, URL http://CRAN.R-project.org/package=logcondens.

Shor NZ (1985). *Minimization Methods for Non-Differentiable Functions*. Springer-Verlag, Berlin.

Venables WN, Ripley BD (2002). *Modern Applied Statistics with S*. Springer-Verlag, New York.

Wand MP, Jones MC (1995). *Kernel Smoothing*. Chapman and Hall, CRC Press, Florida.

Zeileis A, Hornik K, Murrell P (2008). "Escaping RGBland: Selecting Colors for Statistical Graphics." *Computational Statistics & Data Analysis*. Forthcoming.

Affiliation:

Madeleine Cule, Robert Gramacy, Richard Samworth
University of Cambridge
Statistical Laboratory
Centre for Mathematical Sciences
Wilberforce Road
Cambridge CB3 0WG, United Kingdom

E-mail: mlc40@statslab.cam.ac.uk
bobby@statslab.cam.ac.uk
rjs57@statslab.cam.ac.uk
URL: http://www.statslab.cam.ac.uk/~mlc40/
http://www.statslab.cam.ac.uk/~bobby/
http://www.statslab.cam.ac.uk/~rjs57/

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 29, Issue 2 *Submitted:* 2008-07-07
January 2009 *Accepted:* 2008-12-14