SFB 823

The benchden package: Benchmark densities for nonparametric density estimation

Thoralf Mildenberger, Henrike Weinert

Nr. 14/2011

Discussion Paper

SFB 823

The benchden Package: Benchmark Densities for Nonparametric Density Estimation

Thoralf Mildenberger
Department of Mathematics
University of Bayreuth
95440 Bayreuth, Germany
E-mail: thoralf.mildenberger@uni-bayreuth.de

Henrike Weinert
Faculty of Statistics
TU Dortmund University
44221 Dortmund, Germany
E-mail: weinert@statistik.tu-dortmund.de

March 17, 2011

Abstract

This article describes the benchden package which implements a set of 28 example densities for nonparametric density estimation in R. In addition to the usual functions that evaluate the density, distribution and quantile functions or generate random variates, a function designed to be specifically useful for larger simulation studies has been added. After describing the set of densities and the usage of the package, a small toy example of a simulation study conducted using the benchden package is given.

1 Introduction

In the last decades, nonparametric curve estimation has become an important field of research. Apart from the invention of new methods, there has been great progress in the theoretical analysis of the properties of nonparametric methods. However, many results are still to a large extent of an asymptotic nature, and are often derived under some restrictive conditions on the estimand. To make matters worse, these conditions are usually not empirically verifiable. To assess the performance of nonparametric methods for small or medium sized samples and for situations not covered by the conditions required for the theoretical results, simulation studies are needed. Indeed, most published articles suggesting a new method contain a simulation study in which the proposed method is compared to at least a few competitors. Since a comparison of methods on real-life data sets has the drawback that the correct solution is usually unknown, one often resorts to a comparison on artificial data sets generated under a completely known mechanism which allows for a more objective assessment of the behaviour of different methods.

Over the years, for many problems in nonparamteric statistics and related areas widely used test functions have evolved as a standard for comparison, most notably the Donoho-Johnstone functions (Blocks, Bumps, Doppler and HeaviSine) originally introduced in Donoho and Johnstone (1994) in the context of Wavelet denoising and the 'Lena' or 'Peppers' images frequently used in image analysis (see for example the 'miscellaneous' section of the USC-SIPI Image Database `http://sipi.usc.edu/database/database.php`). In both cases, the test functions have well-known features (discontinuities or certain textures, for example) that resemble the difficulties encountered in specific applications.

While generating artificial data sets is relatively easy for regression and image analysis – one just needs to add random noise to a discretized version of the function or image – conducting simulation studies for density estimation requires a bit more effort, since at least a function to evaluate the density and one to generate random samples from the density are required. Additionally, there seems to be no generally used set of test densities for density estimation, with the possible exception of the set of normal mixture densities proposed by Marron and Wand (1992) and implemented in the R package nor1mix (Mächler, 2010).

Our package benchden (Mildenberger et al., 2011), which is described in this article, aims at closing this gap. It implements the set of 28 test bed densities first introduced by Berlinet and Devroye (1994) and since used in Devroye (1997) and Rozenholc et al. (2010) in R (R Development Core Team, 2011). This set of 28 densities is sufficiently large to cover a wide variety of situations that are of interest for the comparison of different methods. Unlike the densities proposed by Marron and Wand (1992), which vary greatly in shape but are all normal mixtures, these densities also differ widely in their mathematical properties such as smoothness or tail behaviour and even include some densities with infinite peaks that are not square-integrable. They include both densities from standard families of distributions as well as some examples specifically constructed to pose special challenges to estimation procedures. Hence, this suite should be large enough to choose an appropriate subset of interesting cases for most simulation studies.

In addition to providing functions `dberdev`, `pberdev`, `qberdev` and `rberdev` for the evaluation of the density, distribution and quantile functions and for generating random variates, we offer a function `berdev` specially designed to be helpful in larger simulation studies. This function returns a list that contains some information about the densities, such as a string giving the name (useful for automatic generation of tables of results) and a vector containing the points of discontinuity of the density (which will be needed for many numerical integration methods).

In the implementation, we followed some basic principles to ensure suitability of the package – with respect to reliability, reproducibility and speed – for its use in simulation studies:

- The densities are implemented exactly in the versions given in Berlinet and Devroye (1994) with no further free parameters or options that affect location, scale or shape.
- In case a density has already been implemented in the standard stats package included with R, we use this implementation.
- Random variates are either generated by transformation of standard random variates already implemented in R or by an explicit inversion of the distribution function. Only for the caliper density (number 25), a rejection algorithm is used (in a fast, vectorized implementation).
- Quantiles are calculated using an explicit inversion of the distribution function whereever possible. For a few densities (numbers 15, 21-25 and 28), we use numerical inversion which makes the calculation of quantiles slower than for the other densities.
- Unless absolutely necessary, the implementation (especially w.r.t random variate generation) will not be changed in subsequent updates to ensure full reproducibility. This means that different versions of the package will produce exactly the same samples, given the same random seed.

The paper is organized as follows: in the second section, we describe the Berlinet and Devroye set of densities and some of their properties as well as some issues of the implementation. The third section describes the usage of the functions in benchden, while the fourth section contains a toy example of a simulation study implemented using the package. The fifth section contains a few concluding remarks.

2 The densities

We now give a detailed description of the densities. The list basically follows Berlinet and Devroye (1994), but is supplemented with some additional information and a few details on random variate generation:

1. **Uniform:** The density of the uniform distribution on $[0, 1]$. The standard R implementation from the stats package is used.
2. **Exponential:** The density of the Exp(1) exponential distribution. The standard R implementation from the stats package is used.
3. **Maxwell:** The density is given by $f(x) = x \exp(-\frac{x^2}{2})$ on $[0, \infty)$. Random variates are generated by inversion of the distribution function.
4. **Double Exponential:** The standard double exponential distribution with density given by $f(x) = \frac{1}{2}\exp(-|x|)$.
5. **Logistic:** The standard logistic distribution with density given by $f(x) = \frac{\exp(-x)}{(1+\exp(-x))^2}$. The standard R implementation from the stats package is used.
6. **Cauchy:** The density of the Cauchy(0,1)-Distribution. The standard R implementation from the stats package is used.
7. **Extreme value:** The density of an extreme value distribution with distribution function $F(x) = \exp(-\exp(-x))$ and density $f(x) = \exp(-x - \exp(-x))$. Random variates are generated by inversion of the distribution function.
8. **Infinite peak:** A distribution with density $f(x) = (2\sqrt{x})^{-1}$ on $(0, 1)$. Due to the infinite peak in 0, the density is not in L_2 (and hence not in L_∞). Since this is also the density of U^2, where U is a standard uniform random variable, random variates are generated by squaring standard uniform random variates.
9. **Pareto:** The Pareto distribution with parameter 3/2: $f(x) = (2x^{3/2})^{-1}$ on $[1, \infty)$. Random variates from this heavy-tailed distribution are generated by inversion of the distribution function.
10. **Symmetric Pareto:** A translated and symmetrized version of density 9. The density function is $f(x) = (4(1 + |x|)^{3/2})^{-1}$.
11. **Normal:** The density of a N(0,1)-Distribution. The standard R implementation from the stats package is used.
12. **Lognormal:** The standard Lognormal distribution with density function given by $f(x) = (x\sqrt{2\pi})^{-1}\exp(-(\log x)^2/2)$ on $[0, \infty)$. The standard R implementation from the stats package is used.
13. **Uniform scale mixture:** A mixture of two uniform distributions with overlapping support: $\frac{1}{2}U[-\frac{1}{2}, \frac{1}{2}] + \frac{1}{2}U[-5, 5]$.
14. **Matterhorn:** Density of $S\exp(-2/U)$ with $P(S = -1) = P(S = 1) = \frac{1}{2}$ and U uniformly distributed on $[0, 1]$. The density is $(|x|(\log(|x|))^2)^{-1}$ on $[-e^{-2}, e^{-2}]$ and it is neither in L_2 nor L_∞. Due to limited computational accuracy, the infinte peak effectively is a small point mass at zero, and larger samples generated from this distribution may contain a few realizations equal to zero.
15. **Logarithmic peak:** The density of UV, where U and V are independently $U[0, 1]$-distributed. The density is $f(x) = -\log(x)$ on $(0, 1)$ and although it has an infinte peak, it is in L_2 (but not in L_∞). Quantiles are calculated by numerical inversion of the distribution function.
16. **Isosceles triangle:** The density of a triangle distribution $f(x) = (1 - |x|)_+$.

17. **Beta (2,2):** The Beta(2,2)-distribution with density given by $f(x) = 6x(1-x)$ on $[0,1]$.

18. **Chi-square (1):** The χ^2-Distribution with 1 degree of freedom. The density is $(\sqrt{2\pi x})^{-1}\exp(-\frac{x}{2})$ for $x > 0$ and is not in L_2 and L_∞.

19. **Normal cubed:** The density of N^3, where N is standard Gaussian. The density is $f(x) = \frac{\sqrt{2}}{6\sqrt{\pi}}x^{-2/3}\exp\left(-\frac{1}{2}x^{2/3}\right)$ and is not in L_2 and L_∞.

20. **Inverse exponential:** Distribution of E^{-2}, where E is Exp(1)-distributed. The density is $f(x) = \frac{1}{2}x^{-3/2}\exp(-\frac{1}{\sqrt{x}})$ on $[0,\infty)$.

21. **Marronite:** A very well separated mixture of two normals. The density is given by $f(x) = \frac{1}{3}\phi(\frac{x+20}{1/4}) + \frac{2}{3}\phi(x)$, where ϕ is the standard normal density. Quantiles are calculated by numerical inversion of the distribution function.

22. **Skewed bimodal:** A mixture of two normals with density given by $f(x) = \frac{3}{4}\phi(x) + \frac{1}{4}\phi(\frac{x-1.5}{1/3})$, where ϕ is the standard normal density. Identical to density number 8 in Marron and Wand (1992). Quantiles are calculated by numerical inversion of the distribution function.

23. **Claw:** A mixture of six normals with density given by $f(x) = \frac{1}{2}\phi(x) + \frac{1}{10}\phi(\frac{x+1}{0.1}) + \frac{1}{10}\phi(\frac{x+0.5}{0.1}) + \frac{1}{10}\phi(\frac{x}{0.1}) + \frac{1}{10}\phi(\frac{x-0.5}{0.1}) + \frac{1}{10}\phi(\frac{x-1}{0.1})$, where ϕ is the standard normal density. Identical to density number 10 in Marron and Wand (1992). Quantiles are calculated by numerical inversion of the distribution function.

24. **Smooth comb:** A mixture of six normals with density given by $f(x) = \frac{32}{63}\phi(\frac{x+31/21}{32/63}) + \frac{16}{63}\phi(\frac{x-17/21}{16/63}) + \frac{8}{63}\phi(\frac{x-41/21}{8/63}) + \frac{4}{63}\phi(\frac{x-53/21}{4/63}) + \frac{2}{63}\phi(\frac{x-59/21}{2/63}) + \frac{1}{63}\phi(\frac{x-62/21}{1/63})$, where ϕ is the standard normal density. Identical to density number 14 in Marron and Wand (1992). Quantiles are calculated by numerical inversion of the distribution function.

25. **Caliper:** The Density of $S(X+0.1)$, where $P(S=-1) = P(S=1) = \frac{1}{2}$ and X has the density of $f(x) = 4(1-x^{1/3})$ on $[0,1]$. The random variate X is generated via a simple rejection algorithm that was implemented in a vectorized version. Quantiles are calculated by numerical inversion of the distribution function.

26. **Trimodal uniform:** The density of a mixture of three uniform distributions with disjoint support $\frac{1}{2}U[-1,1] + \frac{1}{4}U[-20.1,-20] + \frac{1}{4}U[20,20.1]$.

27. **Sawtooth:** The density of $N + X$, where N uniformly distributed on $\{-9,-7,-5,-3,-1,1,3,5,7,9\}$ and X has the isosceles triangular density on $[-1,1]$ (see no. 16).

28. **Bilogarithmic peak:** The density is $f(x) = -\frac{1}{2}\log(x(1-x))$ on $[0,1]$ and is in L_2, but not in L_∞. Quantiles are calculated by numerical inversion of the distribution function.

All densities are depicted in Figure 1. In Table 1 we give an overview of the densities and a few of their properties: We indicate whether the density is in L_2 and L_∞, and whether it is continous or even differentiable on the whole real line (not just on the support). Furthermore, we categorize the densities with respect to whether the support is the whole real line, an interval of the type $[a,\infty)$ for some real a ('half line'), a compact interval ('compact') or a union of disjoint compact intervals with gaps in between ('gaps'). We then distinguish three different types of tail behaviour: if the support is compact, the density has no tails. If the support is unbounded, we say that the tails are 'light' iff $f(x) = O(\exp(-x))$ for $|x| \to \infty$ and 'heavy' otherwise. The last column of the table gives the number of modes of the densities. If a local maximum is attained on a whole interval, we count this as one mode (i.e., the uniform density has one mode and not infinitely many).

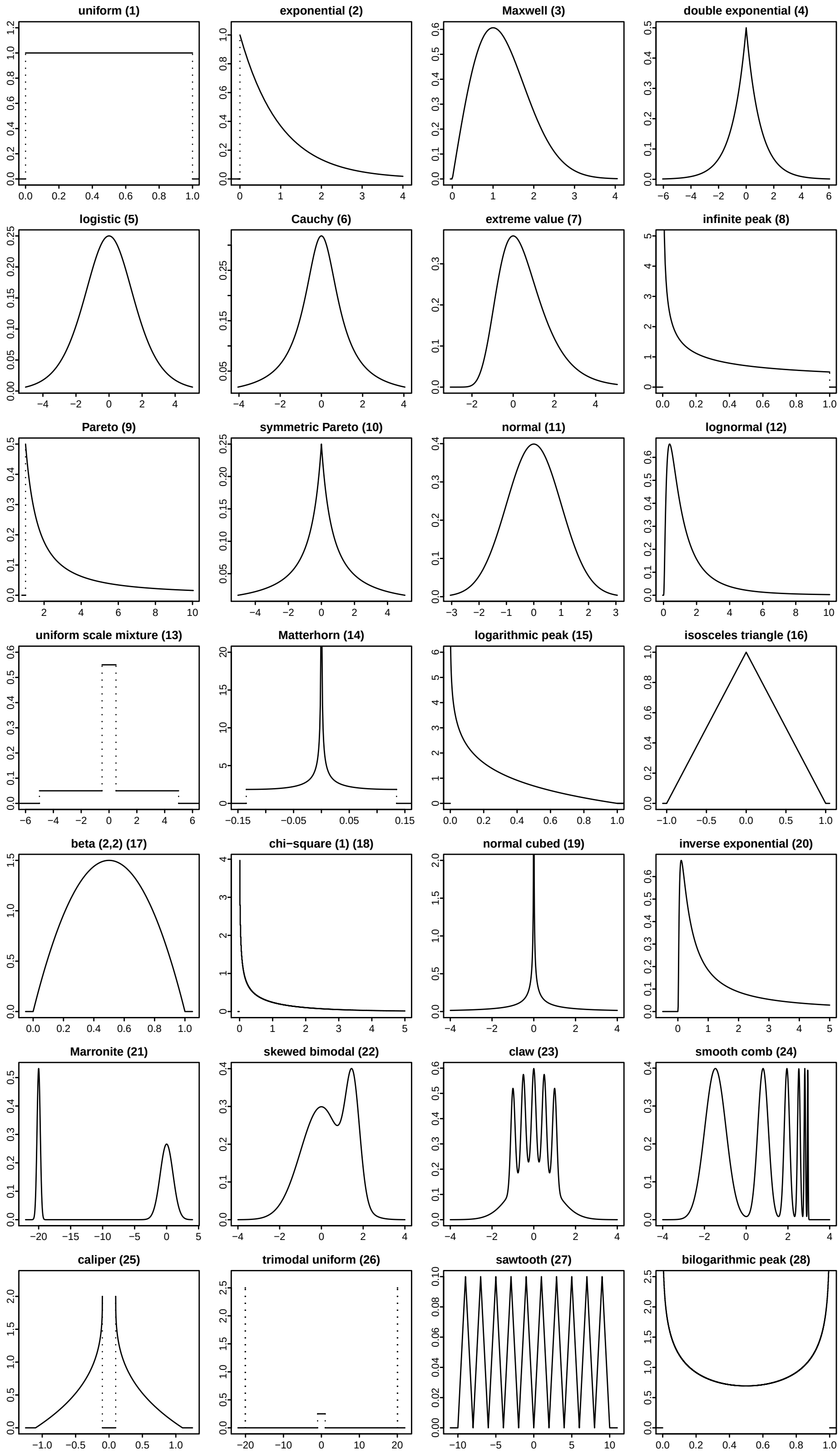

Figure 1: The 28 test bed densities

	name	L_2	L_∞	cont.	diff.	support	tails	modes
1	Uniform	yes	yes	no	no	compact	none	1
2	Exponential	yes	yes	no	no	half line	light	1
3	Maxwell	yes	yes	yes	no	half line	light	1
4	Double Exponential	yes	yes	yes	no	real line	light	1
5	Logistic	yes	yes	yes	yes	real line	light	1
6	Cauchy	yes	yes	yes	yes	real line	heavy	1
7	Extreme value	yes	yes	yes	yes	real line	light	1
8	Infinite peak	no	no	no	no	compact	none	1
9	Pareto	yes	yes	no	no	half line	heavy	1
10	Symmetric Pareto	yes	yes	yes	no	real line	heavy	1
11	Normal	yes	yes	yes	yes	real line	light	1
12	Lognormal	yes	yes	yes	yes	half line	heavy	1
13	Uniform scale mixture	yes	yes	no	no	compact	none	1
14	Matterhorn	no	no	no	no	gaps	none	1
15	Logarithmic peak	yes	no	no	no	compact	none	1
16	Isosceles triangle	yes	yes	yes	no	compact	none	1
17	Beta (2,2)	yes	yes	yes	no	compact	none	1
18	Chi-square (1)	no	no	no	no	half line	heavy	1
19	Normal cubed	no	no	no	no	real line	heavy	1
20	Inverse exponential	yes	yes	yes	yes	half line	heavy	1
21	Marronite	yes	yes	yes	yes	real line	light	2
22	Skewed bimodal	yes	yes	yes	yes	real line	light	2
23	Claw	yes	yes	yes	yes	real line	light	5
24	Smooth comb	yes	yes	yes	yes	real line	light	6
25	Caliper	yes	yes	no	no	gaps	none	2
26	Trimodal uniform	yes	yes	no	no	gaps	none	3
27	Sawtooth	yes	yes	yes	no	compact	none	10
28	Bilogarithmic peak	yes	no	no	no	compact	none	2

Table 1: The 28 densities and some of their properties.

As can be seen from the table, this set of densities contains a large number of densities modelling various difficulties encountered in practice. It should be rich enough to choose a subset of interesting cases for most applications, depending on the goal of estimation and the methods under consideration. Additionally, the benchden package contains four histogram densities which we will not describe here, but see Rozenholc et al. (2009).

3 Usage

Once the benchden package has been loaded, the density, the distribution function, the quantile function and a random sample from the distribution can be obtained by calling the functions

```
dberdev(x,dnum=1)
pberdev(q,dnum=1)
qberdev(p,dnum=1)
```
and
```
rberdev(n,dnum=1)
```

repectively, just like for any other distribution implemented in R. The argument `dnum` is an integer between `1` and `28` giving the number of the distribution as described in section 2, `x` and `q` are vectors of quantiles, `p` is a vector of probabilities and `n` the number of observations. Since the densities are meant to provide a standard for comparison in simulations studies, there are no further free parameters that affect location, scale or shape.

Additionally, the package provides a function `berdev` giving some information which could be relevant in simulation studies.

The usage is
`berdev(dnum)`
where the number of the density `dnum` is the only argument. The function returns a list with the four components `name`, `peaks`, `support` and `breaks`.

The first entry of the list gives the name of the distribution as a character string which may be useful when automatically generating pictures or tables.

The `peaks` component of the list contains an ordered vector of the positions of the peaks or modes of each density which is needed in simulations for situations where the modes of an estimate should be compared with those of the true density. Especially for the multimodal densities (numbers 21 - 28) this is helpful since the positions of their modes could not be seen at first sight. For example, the modes of the claw density are

```
>> berdev(23)$peaks
[1] -0.9969638 -0.4978001  0.0000000  0.4978001  0.9969638
```

If a local maximum is taken on an interval, the location is given as the midpoint of this interval. For example, for the standard uniform density, the single mode is the midpoint of the support:

```
>> berdev(1)$peaks
[1] 0.5
```

Using `berdev(dnum)$support` one can obtain the support of the densities. The support is given as matrix, with the first column giving the left and the second column giving the right end of an (possibly infinite) interval. The matrix contains several rows if the support is the union of disjoint intervals (which is only the case for densities 25 and 26) and only one row if the support consists of just one interval. For example, the trimodal uniform density has support

```
>> berdev(26)$support
      [,1]  [,2]
[1,] -20.1 -20.0
[2,]  -1.0   1.0
[3,]  20.0  20.1
```

while the inverse exponential (number 20) on $[0, \infty)$ has support:

```
>> berdev(20)$support
     [,1] [,2]
[1,]    0  Inf
```

To compute a measure of distance between the true and the estimated density, one usually has to use some type of numerical integration scheme. For this, it is often necessary to split up the range of integration into intervals where both the true density and the estimate are sufficiently smooth. The fourth component of the list, `berdev(dnum)$breaks` is a vector of points where the density is not continuous or not differentiable. These points can be used as boundary points for piecewise integration. For an example see section 4.

For backward compatibility, the package contains two functions `nberdev` and `bberdev`, which just return the `name` and `breaks` components of the list returned by `berdev`.

4 An example

We now give a toy example of how one might conduct a simulation study of different density estimators using the benchden package. For this, we compare the kernel density estimator implemented in the function `density` in the stats package using two different bandwidth selectors (a plug-in rule and cross-validation) on three densities and two sample sizes. As a measure of quality we use the L_1-risk. First, we need to load the benchden and xtable packages:

```
library("benchden")
library("xtable")
```

The latter is only used in the last step to generate a nice table and is not needed for the actual calculations.

Given a density f and an estimate $\hat{f}$, the L_1 loss for a single simulation run is calculated using numerical integration of $|f - \hat{f}|$. The integral over an interval $[a, b]$ of a function g that is continous on that interval may be numerically evaluated using a trapezoidal rule:

$$\int_a^b g(x)dx \approx \frac{b-a}{m}\left(\frac{1}{2}g(x_0) + g(x_1) + \cdots + g(x_{m-1}) + \frac{1}{2}g(x_m)\right)$$

with gridpoints $x_j := a + j\frac{b-a}{m}$, $j = 0, \ldots, m$. To apply the trapezoidal rule to $g := |f - \hat{f}|$, we need to partition the real line into intervals that do not contain points where either f or $\hat{f}$ is not continous. Given an interval $[a, b]$ without discontinuities of g in the interior (but the boundary points may be discontinuities), we can use the trapezoidal rule on the interval $[a+\varepsilon, b-\varepsilon]$ for some small $\varepsilon > 0$ to approximate the integral over the open interval (a, b). The following function evaluates $\int_a^b |f(x) - \hat{f}(x)|dx$. The first arguments of `integ.interval` are the sample `x` and the number of the density `dnum`. Since both methods compared are kernel density estimators differing only in the method of bandwidth selection, the function takes the selected bandwidth `h` as third argument, followed by a vector `bounds` giving the left and right endpoints of the interval under consideration (which are then slightly moved to the interior to ensure integration over an open interval, since the endpoints may be discontinuity points). Finally `gridsize` gives the number of grid points for the trapezoidal rule:

```
integ.interval<-function(x, dnum, h, bounds, m = 1000) {
  a <- bounds[1] + 10^(-11)
  b <- bounds[2] - 10^(-11)
  esteval <- density(x, bw = h, n = m + 1, from = a, to = b)
  gridpoints<-esteval$x
  denseval<-dberdev(gridpoints, dnum=dnum)
  g <- abs (denseval - esteval$y)
  (b - a) / m * sum (0.5 * g[1] + sum( g[2:m] ) + 0.5 * g[m + 1])
}
```

Now the real line is partitioned into intervals where the true density is continous. Since we use the `density` function with the default Gaussian kernel, the estimated density is always continous and the only discontinuity points of $g = |f - \hat{f}|$ are the discontinuity points of the true density f. The `berdev` function returns a list including the entry `breaks` containing the vector of discontinuity points (and additionally the points of nondifferentiability, but including these causes no harm). We add the boundary points of the support (replaced with extreme quantiles in case the support is unbounded) and suitable cut-off points for the estimate (to the left of the minimum value of the sample and to the right of the maximum value of the sample). We thus end up with a partition of a finite subinterval of the real line which contains most of the mass of both the estimate and the true density. The elements of this partition are by construction intervals with no discontinuities of g in the interior.

The function `eval.loss`, which takes the sample, the density number and the chosen bandwidth as arguments, now goes through this list of intervals and calls `integ.interval` for each one. The contributions to the overall loss are then added up and their sum returned.

```
eval.loss<-function(x, dnum, h) {
  loss<-0
  x<-sort(x)
  n<-length(x)
  q<-qnorm(1 - 10^ (- 4) / n, sd=h)

  bd <- berdev(dnum = dnum)
  breaks<-c(x[1] - q, x[n] + q, bd$support[is.finite(bd$support)], bd$breaks)
  if (bd$support[1]==-Inf)
breaks <- c(breaks, qberdev(10 ^ - 10, dnum))
  if (bd$support[length(bd$support)]==Inf)
breaks <- c(breaks, qberdev(1 - 10 ^ - 10, dnum))
  breaks <- unique(sort(breaks))
  k<-length(breaks)

  for (i in 1 : (k - 1))      {
      bnd<-breaks[i:(i + 1)]
      if (bnd[2]-bnd[1] > 10 ^ - 8)
loss <- loss+integ.interval(x = x, h = h, dnum = dnum, bounds = bnd)
 }
 loss
}
```

Depending on the loss or risk function and the estimator, densities with heavy tails or infinite peaks require some extra care as numerical integration is problematic in these cases. Simple solutions are to cut out a small intervall containing the infinite peak and to take special care to use a sufficiently rich grid of evaluation points in the tails when these are heavy. Both require some experimentation to make sure that accurate results are obtained. In the case of kernel density estimators, Devroye (1997) suggests an interesting method of evaluating the L_1 loss based on the distribution functions rather than the densities. This method circumvents some of the problems, but there is no obvious way to adapt it to different loss functions.

For our small simulation study, the main program consists of several nested loops. In our case, we evaluate two kernel density estimators using different bandwidths (the `density` function in the `stats` package with `bw = ''nrd0''` and `bw = ''nrd0''`), for three different densities (numbers 1, 2 and 11) and two different sample sizes ($n = 100$ and $n = 250$). The L_1-risk is estimated by averaging the L_1-loss from 10 simulation runs. The results are stored in a three-dimensional array `results` such that `results[i,j,k]` contains the result for the i-th density, the j-th sample size and the k-th method:

```
dens <- c(1, 2, 11)
sizes <- c(100, 250)
replications <- 10
set.seed(0)
results<-array(0, dim=c(length(dens), length(sizes), 2))
```

Density	n	nrd0	ucv
uniform	100	0.245	0.2566
	250	0.1919	0.1749
exponential	100	0.3369	0.3079
	250	0.2641	0.252
normal	100	0.1458	0.1597
	250	0.1084	0.1072

Table 2: Small example of an automatically generated results table.

```
for (i in 1:length(dens)) {
  for (j in 1:length(sizes)) {
    loss <- c(0, 0)
    for (r in 1:replications) {
      x <- rberdev(sizes[j],dnum=dens[i])
      h1 <- density(x,bw="nrd0", n=1)$bw
      h2 <- density(x,bw="ucv", n=1)$bw
      loss <- loss +
        c(eval.loss(x, dnum=dens[i], h=h1), eval.loss(x, dnum=dens[i], h=h2))
    }
    results[i,j,] <- loss / replications
  }
}
```

With the results stored in an array, the xtable package can be used to generate a nice LaTEX table. The following code was used to generate Table 2:

```
table<-matrix(0, nrow=(length(dens) * length(sizes)), ncol=2)
densevec<-c("Density")
nvec<-c("n")

for (j in 1:length(dens)) {
  densevec<-c(densevec, berdev(dens[j])$name, rep("",(length(sizes) - 1)) )
  for (k in 1:length(sizes)) {
    nvec<-c(nvec,sizes[k])
    table[(j - 1) * length(sizes) + k,]<-round(results[j, k, ] * 10000) / 10000
  }
}

table<-rbind(c("nrd0", "ucv"),table)
table<-cbind(densevec, nvec, table)

cap<-"Small example of an automatically generated results table."
hlvec<-c(1,((1:length(dens)) - 1)*length(sizes) + 1, dim(table)[1] )

print(xtable(table, caption = cap, align = "lllll"), include.colnames = F,
  include.rownames = F, hline.after = hlvec, caption.placement = "bottom")
```

For details on the printing options for the `xtable` function see the documentation of the xtable package (Dahl, 2009).

5 Concluding remarks

The benchden package is designed to make life easier for researchers working in nonparametric density estimation. We provide an implementation of the suite of 28 test densities proposed by Berlinet and Devroye (1994) which should be rich enough to contain intersting examples for most problems in density estimation. In addition to the usual functions for evaluating the density, distribution and quantile functions and for random variate generation, our package includes a function giving further information on the density which is specifically designed for use in simulation studies. Our hope is that the package will encourage the use of the Berlinet and Devroye set of densities in simulation studies.

Acknowledgement

This work has been supported by the Collaborative Research Center 'Statistical modelling of nonlinear dynamic processes' (SFB 823, Projects B1 and C1) of the German Research Foundation (DFG). The authors also wish to thank Yves Rozenholc for introducing us to the Berlinet and Devroye set of densities, Luc Devroye for providing his implementaion for testing purposes and Sebastian Tiemeyer for programming assistance.

References

Berlinet, A. and L. Devroye (1994). A comparison of kernel density estimates. *Publications de l'Institute de Statistique de L'Universite de Paris 38*, 3–59.

Dahl, D. B. (2009). *xtable: Export tables to LaTeX or HTML.* R package version 1.5-6.

Devroye, L. (1997). Universal smoothing factor selection in density estimation: Theory and practice (with discussion). *Test 6*, 223–320.

Donoho, D. L. and I. Johnstone (1994). Ideal spatial adaption by wavelet shrinkage. *Biometrika 81*, 425–455.

Mächler, M. (2010). *nor1mix: Normal (1-d) Mixture Models (S3 Classes and Methods).* R package version 1.1-2.

Marron, J. and M. Wand (1992). Exact mean integrated squared error. *The Annals of Statistics 20*, 712–736.

Mildenberger, T., H. Weinert, and S. Tiemeyer (2011). *benchden: 28 benchmark densities from Berlinet/Devroye (1994).* R package version 1.0.4.

R Development Core Team (2011). *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing. ISBN 3-900051-07-0.

Rozenholc, Y., T. Mildenberger, and U. Gather (2009). Combining regular and irregular histograms by penalized likelihood. Discussion Paper 31/2009, SFB 823, Technische Universitt Dortmund.

Rozenholc, Y., T. Mildenberger, and U. Gather (2010). Combining regular and irregular histograms by penalized likelihood. *Computational Statistics and Data Analysis 54*, 3313–3323.