Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *doi: 10.18637/jss.v000.i00*

npbr: A Package for Nonparametric Boundary Regression in R

Abdelaati Daouia
Toulouse School of Economics
& Université Catholique de Louvain

Thibault Laurent
Toulouse
School of Economics

Hohsuk Noh
Sookmyung Women's
University

Abstract

The package **npbr** is the first free specialized software for data edge and frontier analysis in the statistical literature. It provides a variety of functions for the best known and most innovative approaches to nonparametric boundary estimation. The selected methods are concerned with empirical, smoothed, unrestricted as well as constrained fits under both single and multiple shape constraints. They also cover data envelopment techniques as well as robust approaches to outliers. The routines included in **npbr** are user friendly and afford a large degree of flexibility in the estimation specifications. They provide smoothing parameter selection for the modern local linear and polynomial spline methods as well as for some promising extreme value techniques. Also, they seamlessly allow for Monte Carlo comparisons among the implemented estimation procedures. This package will be very useful for statisticians and applied researchers interested in employing nonparametric boundary regression models. Its use is illustrated with a number of empirical applications and simulated examples.

Keywords: boundary curve, concavity, extreme-values, kernel smoothing, linear programming, local linear fitting, monotonicity, multiple shape constraints, piecewise polynomials, spline smoothing, R.

1. Introduction

In the standard regression model

$$y_i = \varphi(x_i) + \varepsilon_i, \quad i = 1, \ldots, n,$$

where the data (x_i, y_i) are observed, a variety of programs specializing in nonparametric and semi-parametric estimation have recently appeared. Prominent among these routines is the popular **np** package (Hayfield and Racine 2008), which allows R (R Core Team 2017) users

to conduct, for instance, nonparametric mean and quantile regression. In the non-standard boundary regression model, in contrast to classical theory, the regression errors (ε_i) are not assumed to be centred, but to have a one-sided support $(-\infty, 0]$, and the regression function φ describes some boundary curve. The present **npbr** package (Daouia, Laurent, and Noh 2017) is a collection of functions that perform a variety of nonparametric estimation techniques of the frontier function φ in the statistical software environment R. Specifically, suppose that we have n pairs of observations $(x_i, y_i),\ i = 1, \ldots, n$, from a bivariate distribution with a density $f(x, y)$ in $\mathbb{R}^2$. The support Ψ of f is assumed to be of the form

$$\begin{aligned} \Psi = \{(x,y)|y \leq \varphi(x)\} &\supseteq \{(x,y)|f(x,y) > 0\}, \\ \{(x,y)|y > \varphi(x)\} &\subseteq \{(x,y)|f(x,y) = 0\}, \end{aligned}$$

where the graph of φ corresponds to the locus of the curve above which the density f is zero. More specifically, this graph is the extremal regression quantile curve corresponding to the probability level 1. We consider the estimation of the frontier function φ based on the sample $\{(x_i, y_i),\ i = 1, \ldots, n\}$. This problem has increasing usage in various fields such as classification, cluster analysis, economics, education, finance, management, physics, public policy, scatter-point image analysis, and other arenas. For example, in image reconstruction, the frontier-or-edge is typically the interface of areas of different intensities or differing color tones, perhaps black above the boundary (where no observations are recorded) and grey below (see Park 2001, for a nice summary and an extensive bibliography).

In most applications, the frontier function φ is assumed to be monotone or concave (convex) monotone. This naturally occurs when analyzing, for instance, the reliability of nuclear reactors where x_i represents the temperature of the reactor pressure vessel material i and y_i represents its fracture toughness. The main goal is to estimate the highest fracture toughness φ as a function of the temperature. From a physical point of view, this master curve is known to be increasing and is believed to be concave (see Daouia, Girard, and Guillou 2014; Daouia, Noh, and Park 2016).

According to the micro-economic theory of the firm, the support boundary is interpreted as the set of the most efficient businesses or industries that are optimally using inputs x_i (labor, energy, capital, etc.) to produce their outputs y_i (produced goods or services). Econometrics considerations often lead to the assumption that the cost/production function φ is monotone nondecreasing with/without concavity. The concavity assumption is not always valid, although it is widely used in economics. For example, the production set Ψ might admit increasing returns to scale, that is, the outputs might increase faster than the inputs (see, *e.g.,* Daouia, Girard, and Guillou 2014). Another related field of application where monotone boundaries and convex supports naturally appear is portfolio management. In the Capital Assets Pricing Models, the upper support extremity gives a benchmark relative to which the performance of an investment portfolio can be measured. Here, x_i measures the risk (volatility or variance) of a portfolio, y_i its averaged return, and φ is required to be both monotone and concave (see, *e.g.,* Gijbels, Mammen, Park, and Simar 1999). Such examples are abundant in economics and related fields.

Nonparametric boundary regression is clearly a problem involving extreme value theory. Already in the case of production econometrics, Hendricks and Koenker (1992) stated, "In the econometric literature on the estimation of production technologies, there has been considerable interest in estimating so called frontier production models that correspond closely to models for extreme quantiles of a stochastic production surface". Chernozhukov (2005) and

Daouia, Gardes, and Girard (2013) may be viewed as the first attempt to actually implement theoretically the idea of Hendricks and Koenker, respectively, in a linear regression model and in a general nonparametric model. However, their approaches aim to estimate an extreme regression quantile curve instead of the true full frontier φ. Thereby the use of high regression quantiles might be viewed as an exploratory tool, rather than as a method for final frontier analysis. To this end, one may employ the R packages **cobs** (Ng and Maechler 2007), **quantreg** (Koenker 2017) and **splines** (R Core Team 2017), to name a few.

There is a vast literature on nonparametric frontier estimation, including extreme-value methods (de Haan and Resnick 1994; Hall, Nussbaum, and Stern 1997; Gijbels and Peng 2000; Girard and Jacob 2003, 2004; Daouia, Florens, and Simar 2010), projection techniques (Jacob and Suquet 1995), piecewise polynomials (Korostelev and Tsybakov 1993; Härdle, Park, and Tsybakov 1995), local polynomials (Hall and Park 2004; Hall, Park, and Stern 1998; Knight 2001). It is often assumed that the joint density of the data $f(x, y)$ is an algebraic function of the distance from the upper support extremity with a power $\beta_x > -1$, *i.e.*,

$$f(x, y) = c_x \{\varphi(x) - y\}^{\beta_x} + o(\{\varphi(x) - y\}^{\beta_x}) \quad \text{as} \quad y \uparrow \varphi(x),$$

with c_x being a strictly positive function in x. The quantity $\beta_x \neq 0$ describes the rate at which the density decays to zero smoothly ($\beta_x > 0$) or rises up to infinity ($\beta_x < 0$) as it approaches the boundary. The power $\beta_x = 0$ corresponds to a jump of the density at the boundary $\varphi(x)$. The cases $\beta_x > 0$, $\beta_x = 0$ and $\beta_x < 0$ are referred to as "non-sharp boundary", "sharp boundary" and "default-type boundary", respectively. For instance, the more realistic case of non-sharp boundaries has been studied in Härdle *et al.* (1995), where piecewise polynomials are utilized for estimating $\varphi(x)$. The particular range $\beta_x > 1$ has been considered in Hall *et al.* (1997), where the estimation of $\varphi(x)$ is based on an increasing number of large order statistics generated by the y_i values of observations falling into a strip around x. The case of general β_x has been handled by Gijbels and Peng (2000), where the maximum of all y_i values of observations falling into a strip around x and another extreme-value estimator based on three upper order statistics of these y_i's are considered.

All of the elegant approaches mentioned above do not rely, however, on the inherent shape constraints of monotonicity and concavity/convexity. There are two common empirical approaches for estimating monotone data edges: the free disposal hull (FDH) estimator (Deprins, Simar, and Tulkens 1984) and the data envelopment analysis (DEA) estimator (Farrell 1957) which relies on the additional assumption of concavity/convexity of the boundary curve. Despite the simple nature of these two estimators, their full asymptotic theory has been elucidated only during the last decade (see, *e.g.,* Simar and Wilson 2008).

An improved version of the FDH estimator, referred to as the linearized FDH (LFDH), has been considered in Hall and Park (2002) and Jeong and Simar (2006). Although the FDH, LFDH and DEA estimators provide the fitted values at the observed predictor with monotonicity or monotone concavity, they undersmooth the data and underestimate the true frontier. To reduce these defects, Daouia *et al.* (2016) suggested to combine spline smoothing with constrained estimation under both separate and simultaneous shape constraints. Modern kernel smoothing fits have also been proposed by Parmeter and Racine (2013) to estimate the smooth frontier function, based on recent advances in constrained kernel estimation by Hall and Huang (2001). More recently, Noh (2014) improved the kernel smoothing device of Parmeter and Racine (2013) by considering more adequate optimization criteria and bandwidth selection strategy for the estimator.

Most of the available empirical and smooth estimation techniques are, however, based on envelopment ideas, and hence are very non-robust to outliers and/or extremes. Efforts to remedy such a deficiency have appeared in some nonparametric frontier models (see, *e.g.*, Daouia and Simar 2005; Daouia and Ruiz-Gazen 2006; Daouia and Gijbels 2011; Daouia, Florens, and Simar 2012). Prominent among these recent developments are the contributions of Daouia *et al.* (2010, 2012). Instead of using only the top observations lying on the sample boundary to estimate the true frontier, they show how other extreme observations could help to build robust frontier estimators by using the ideas from Dekkers, Einmahl, and de Haan (1989) and Dekkers and de Haan (1989). Moreover, they provide different useful asymptotic confidence bands for the boundary function under the monotonicity constraint in the case of general β_x. However, such techniques are not without their disadvantages. As it is often the case in extreme-value theory, they require a large sample size to ensure acceptable results.

The overall objective of the present package is to provide a large variety of functions for the best known approaches to nonparametric boundary regression, including the vast class of methods employed in both Monte Carlo comparisons of Daouia *et al.* (2016) and Noh (2014) as well as other promising nonparametric devices, namely the extreme-value techniques of Gijbels and Peng (2000), Daouia *et al.* (2010) and Daouia *et al.* (2012). The various functions in the **npbr** package are summarized in Table 1. We are not aware of any other existing set of statistical routines more adapted to data envelope fitting and robust frontier estimation. Only the classical nonsmooth FDH and DEA methods can be found in some available packages dedicated to the economic literature on measurements of the production performance of enterprises, such as the R package **Benchmarking** (Bogetoft and Otto 2011). Other contributions to the econometric literature on frontier analysis by Parmeter and Racine (2013) can be found at `http://socserv.mcmaster.ca/racinej/Gallery/Home.html`. The package **npbr** is actually the first free specialized software for the statistical literature on nonparametric frontier analysis. The routines included in **npbr** are user friendly and highly flexible in terms of estimation specifications. They allow the user to filter out noise in edge data by making use of both empirical and smooth fits as well as (un)constrained estimates under separate and simultaneous multiple shape constraints. They also provide smoothing parameter selection for the innovative methods based on local linear techniques, polynomial splines, extreme values and kernel smoothing, though the proposed selection procedures can be computationally demanding. To solve the different involved optimization problems, we mainly use the **Rglpk** package (Theussl and Hornik 2017, version >= 0.6-2) based on the C library **GLPK** (Makhorin 2017), version 4.61.

In addition, the package will be very useful for researchers and practitioners interested in employing nonparametric boundary regression methods. On one hand, such methods are very appealing because they rely on very few assumptions and benefit from their modeling flexibility, function approximation power and ability to detect the boundary structure of data without recourse to any *a priori* parametric restrictions on the shape of the frontier and/or the distribution of noise. On the other hand, the package offers R users and statisticians in this active area of research simple functions to compute the empirical mean integrated squared error, the empirical integrated squared bias and the empirical integrated variance of various frontier estimators. This seamlessly allows the interested researcher to reproduce the Monte Carlo estimates obtained in the original articles and, perhaps most importantly, to easily compare the quality of any new proposal with the competitive existing methods. The package **npbr** is available from the Comprehensive R Archive Network (CRAN) at http://CRAN.R-

project.org/package=npbr.

Section 2 presents, briefly, five unrelated motivating data examples concerned with annual sport records, the master curve prediction in the reliability programs of nuclear reactors and with the optimal cost/production assessment in applied econometrics. Section 3 describes in detail the implemented functions of the package and provides practical guidelines to effect the necessary computations. In Section 4, we provide some computational tips that facilitate Monte-Carlo comparisons among frontier estimation methods in a similar way to the simulation studies undertaken by Daouia *et al.* (2016) and Noh (2014).

Function	Description	Reference
`dea_est`	DEA, FDH	Farrell (1957), Deprins *et al.* (1984),
	and linearized FDH	Hall and Park (2002), Jeong and Simar (2006)
`loc_est`	Local linear fitting	Hall *et al.* (1998), Hall and Park (2004)
`loc_est_bw`	Bandwidth choice for local linear fitting	Hall and Park (2004)
`poly_est`	Polynomial estimation	Hall *et al.* (1998)
`poly_degree`	Optimal polynomial degree selection	Daouia *et al.* (2016)
`dfs_momt`	Moment type estimation	Daouia *et al.* (2010), Dekkers *et al.* (1989)
`dfs_pick`	Pickands type estimation	Daouia *et al.* (2010), Dekkers and de Haan (1989)
`rho_momt_pick`	Conditional tail index estimation	Daouia *et al.* (2010), Dekkers *et al.* (1989), Dekkers and de Haan (1989)
`kopt_momt_pick`	Threshold selection for moment/Pickands frontiers	Daouia *et al.* (2010)
`dfs_pwm_regul`	Nonparametric frontier regularization	Daouia *et al.* (2012)
`loc_max`	Local constant estimation	Gijbels and Peng (2000)
`pick_est`	Local extreme-value estimation	Gijbels and Peng (2000)
`quad_spline_est`	Quadratic spline fitting	Daouia *et al.* (2016)
`quad_spline_kn`	Knot selection for quadratic spline fitting	Daouia *et al.* (2016)
`cub_spline_est`	Cubic spline fitting	Daouia *et al.* (2016)
`cub_spline_kn`	Knot selection for cubic spline fitting	Daouia *et al.* (2016)
`kern_smooth`	Nonparametric kernel boundary regression	Parmeter and Racine (2013), Noh (2014)
`kern_smooth_bw`	Bandwidth choice for kernel boundary regression	Parmeter and Racine (2013), Noh (2014)

Table 1: **npbr** functions.

2. Empirical applications

In this section, we illustrate the use of the **npbr** package via five different empirical applications taken from the recent literature. Each dataset is chosen to highlight the specifics of a class of estimation methods:

- The dataset `records` is concerned with the yearly best men's outdoor 1500-metre run times starting from 1966. These annual records, depicted in Figure 1 (a), display some interesting features. Following Jirak, Meister, and Reiss (2014), the lower boundary can be interpreted as the best possible time for a given year. This boundary steadily decreases from 1970 until around the year 2000, followed by a sudden increase. This event leaves room for speculations given that, until the year 2000, it had been very difficult to distinguish between the biological and synthetical EPO. Here, the boundary is not believed to be shape constrained and can be estimated by the polynomial, local linear, spline or kernel smoothing methods described in Sections 3.1 and 3.3.

- The dataset `nuclear` from the US Electric Power Research Institute (EPRI) consists of 254 toughness results obtained from non-irradiated representative steels. For each steel i, fracture toughness y_i and temperature x_i were measured. The scatterplot is given in Figure 1 (b). The objective is to estimate the lower and upper limits of fracture toughness for the reactor pressure vessel materials as a function of the temperature. Given that the nuclear reactors' data are measured accurately, it is natural and more realistic for practitioners to rely on data envelopment estimation techniques that we regroup in Sections 3.1-3.3. Here, the lower support boundary is believed to be both increasing and convex, while the upper extremity is only known to be monotone nondecreasing (see Daouia *et al.* 2014, 2016).

- The dataset `air` is concerned with the assessment of the efficiency of 37 European Air Controllers. The performance of each controller can be measured by its "distance" from the upper support boundary, or equivalently, the set of the most efficient controllers. This dataset is taken from Mouchart and Simar (2002). The scatterplot of the controllers in the year 2000 is given in Figure 1 (c), where their activity is described by one input (an aggregate factor of different kinds of labor) and one output (an aggregate factor of the activity produced, based on the number of controlled air movements, the number of controlled flight hours, etc.). Given the very small sample size and the sparsity in data, only the class of polynomials, piecewise polynomials and spline approximations seems to provide satisfactory fits in this applied setting. This class includes the families of empirical and smooth estimation methods described in Section 3.1. Note also that the efficient frontier here is monotone and can be assumed to be in addition concave (see Daouia, Florens, and Simar 2008; Daouia *et al.* 2016).

- The dataset `post` about the cost of the delivery activity of the postal services in France was first analyzed by Cazals, Florens, and Simar (2002) and then by Aragon, Daouia, and Thomas-Agnan (2005) and Daouia, Florens, and Simar (2010) among others. There are 4000 post offices observed in 1994. For each post office i, the input x_i is the labor cost measured by the quantity of labor, which accounts for more than 80% of the total cost of the delivery activity. The output y_i is defined as the volume of delivered mail (in number of objects). As can be seen from the scatterplot in Figure 1 (d), some

observations look so isolated in the output direction that they seem hardly related to the other observations. As a matter of fact, this dataset is known to contain outliers and it would then look awkward for practitioners to rely on estimation techniques based on data envelopment ideas (see Daouia and Gijbels 2011). This motivated the quest for robust frontier estimation methods in Section 3.4. It should be clear that only these methods allow one to construct valid asymptotic confidence intervals for the unknown support boundary.

- The dataset `green` consists of 123 American electric utility companies. As in the set-up of Gijbels, Mammen, Park, and Simar (1999), we used the measurements of the variables $y_i = \log(q_i)$ and $x_i = \log(c_i)$, where q_i is the production output of the company i and c_i is the total cost involved in the production. A detailed description and analysis of these data can be found in Christensen and Greene (1976). The scatterplot is given in Figure 1 (e). Here, the assumption of both monotonicity and concavity constraints is well accepted and any restricted data envelopment technique such as, for instance, kernel smoothing in Section 3.3 can be applied. Also, in the absence of information on whether these data are recorded accurately, one may favor robust frontier estimation. We caution the user that the robust methods based on extreme-value ideas may require a large sample size of the order of thousands to achieve acceptable fits and confidence intervals.

To help users navigate the methods in the **npbr** package, we describe in Table 2 the type of estimation and shape constraints allowed by each method.

Function	Type of estimator	Allowed constraints
`dea_est`	envelope, piecewise linear	monotonicity, concavity
`loc_est`	envelope, local linear	unconstrained
`poly_est`	envelope, polynomial	unconstrained
`dfs_momt`	robust, extreme quantile	monotonicity
`dfs_pick`	robust, extreme quantile	monotonicity
`dfs_pwm_regul`	robust, probability-weighted moment	monotonicity
`loc_max`	envelope, local constant, local DEA	unconstrained monotonicity, concavity
`pick_est`	robust/envelope, extreme quantile	unconstrained
`quad_spline_est`	envelope, quadratic spline	unconstrained, monotonicity, concavity
`cub_spline_est`	envelope, cubic spline	unconstrained, concavity
`kern_smooth`	envelope, kernel smoother	unconstrained, monotonicity, concavity

Table 2: Characteristics of the estimation methods in **npbr**.

For our illustration purposes, each of the five datasets contains only two variables: one input and one output.

```
R> require("npbr")
R> data("records", "nuclear", "air", "post", "green")
```

The following code will generate Figure 1.

```
R> plot(result ~ year, data = records, xlab = "year", ylab = "1500m record")
R> plot(ytab ~ xtab, data = nuclear, xlab = "temp. of the reactor vessel",
+   ylab = "fracture toughness")
R> plot(ytab ~ xtab, data = air, xlab = "input", ylab = "output")
R> plot(yprod ~ xinput, data = post, xlab = "quantity of labor",
+   ylab = "volume of delivered mail")
R> plot(log(OUTPUT) ~ log(COST), data = green)
```

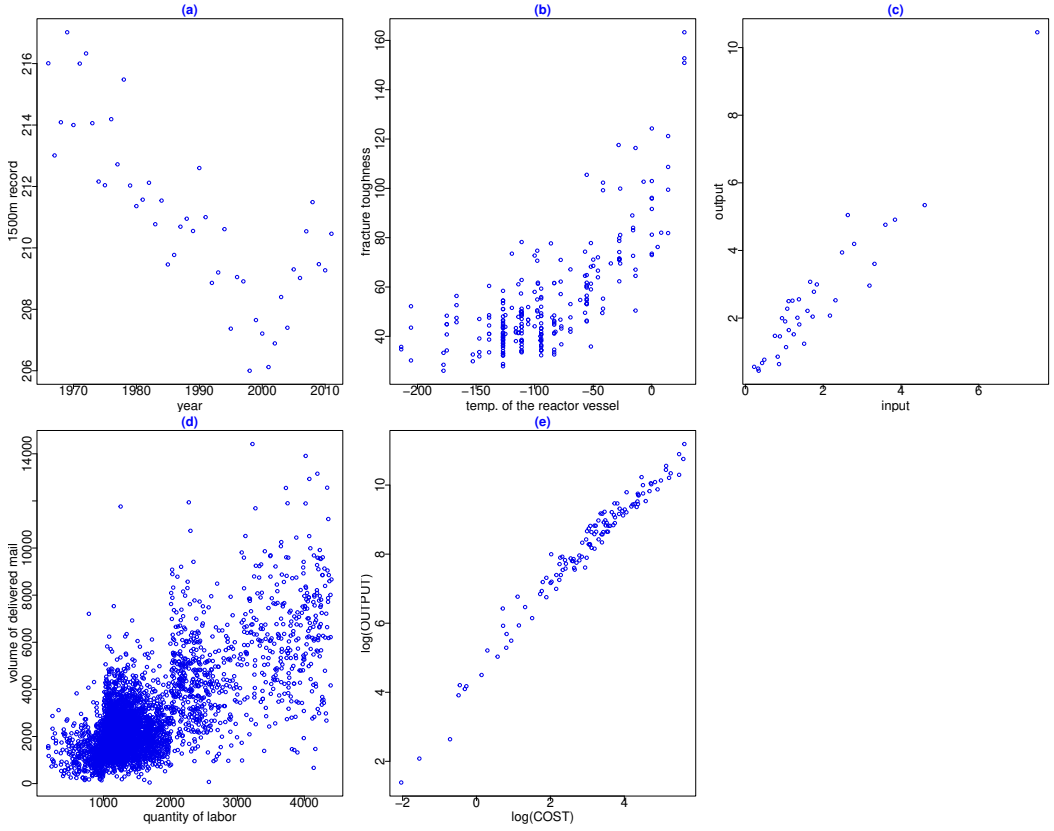

Figure 1: From left to right and from top to bottom, the scatterplots of the yearly best men's outdoor 1500-metre run times in seconds, the 254 nuclear reactors' data, the 37 European Air Controllers, the 4000 European post offices and the 123 American electric utility companies.

3. Main functions

This section describes in detail the main functions of the **npbr** package. The two first arguments of these functions correspond to the observed inputs $x_1, \ldots, x_n$ and the observed

outputs $y_1, \ldots, y_n$. The third argument is a numeric vector of evaluation points at which the estimator is to be computed. Basically, the user can generate a regular sequence of size 100, or any finer grid of points, from the minimum value of inputs x_i to their maximum value. The other arguments of the functions depend on the underlying statistical methods.

We do not presume that the user is familiar with nonparametric frontier modeling hence briefly describe the underlying estimation methodology and tuning parameters selection for each method. Section 3.1 is concerned with piecewise polynomial fitting, Section 3.2 with local polynomial estimation, Section 3.3 with kernel smoothing techniques, and Section 3.4 with robust regularization approaches.

3.1. Piecewise polynomial fitting

We commence with the traditional empirical DEA, FDH and Linearized FDH estimators. We then proceed to polynomial boundary estimators (Hall, Park, and Stern 1998), and finally to constrained spline estimators (Daouia, Noh, and Park 2016).

DEA, FDH and LFDH frontiers

The function `dea_est` implements the empirical FDH, LFDH and DEA frontier estimators programmed earlier in the **Benchmarking** package (Bogetoft and Otto 2011). There are two popular methods for preserving monotonicity in the frontier setting: the free disposal hull (FDH) introduced by Deprins *et al.* (1984) and the data envelopment analysis (DEA) proposed by Farrell (1957). The FDH boundary is the lowest "stair-case" monotone curve covering all the data points

$$\varphi_{fdh}(x) := \max\{y_i,\, i : x_i \leq x\}.$$

An improved version of this estimator, referred to as the linearized FDH (LFDH), is obtained by drawing the polygonal line smoothing the staircase FDH curve. It has been considered in Hall and Park (2002) and Jeong and Simar (2006). When the joint support of the data is in addition convex, the DEA estimator is defined as the least concave majorant of the FDH frontier. Formally, the DEA estimator of the joint support Ψ is defined by

$$\hat{\Psi} = \left\{ (x,y) | y \leq \sum_{i=1}^{n} \gamma_i y_i;\; x \geq \sum_{i=1}^{n} \gamma_i x_i \quad \text{for some} \quad (\gamma_1, \ldots, \gamma_n), \right.$$
$$\left. \text{such that} \quad \sum_{i=1}^{n} \gamma_i = 1;\; \gamma_i \geq 0;\; i = 1, \ldots, n \right\}.$$

Then the DEA estimator of the frontier function φ at x is defined by

$$\varphi_{dea}(x) := \sup\{y | (x,y) \in \hat{\Psi}\}.$$

Note that the FDH, LFDH and DEA estimators are well defined whenever there exists an x_i such that $x_i \leq x$. To illustrate the difference between these three empirical frontiers, we consider the `air` and `green` data. First, we generate a vector of evaluation points.

```
R> x.air <- seq(min(air$xtab), max(air$xtab), length.out = 101)
R> x.green <- seq(min(log(green$COST)), max(log(green$COST)),
+   length.out = 101)
```

Then, we compute the DEA, FDH and LFDH estimates.

```
R> y.dea.green = dea_est(log(green$COST), log(green$OUTPUT), x.green,
+   type = "dea")
R> y.fdh.green = dea_est(log(green$COST), log(green$OUTPUT), x.green,
+   type = "fdh")
R> y.lfdh.green = dea_est(log(green$COST), log(green$OUTPUT), x.green,
+   type = "lfdh")
R> y.dea.air <- dea_est(air$xtab, air$ytab, x.air, type = "dea")
R> y.fdh.air <- dea_est(air$xtab, air$ytab, x.air, type = "fdh")
R> y.lfdh.air <- dea_est(air$xtab, air$ytab, x.air, type = "lfdh")
```

Figure 2 plots the resulting piecewise linear curves. The following code will generate Figure 2.

```
R> plot(y.dea.green ~ x.green,  lty = 4, col = "cyan", type = "l",
+   xlab = "log(cost)", ylab = "log(output)")
R> lines(x.green, y.fdh.green, lty = 1, col = "green")
R> lines(x.green, y.lfdh.green, lty = 2, col = "magenta")
R> legend("topleft", legend = c("DEA", "FDH", "LFDH"), bty = "n",
+   col = c("cyan", "green", "magenta"), lty = c(4, 1, 2))
R> points(log(OUTPUT) ~ log(COST), data = green)
R> plot(x.air, y.dea.air,  lty = 4, col = "cyan",
+   type = "l", xlab = "input", ylab = "output")
R> lines(x.air, y.fdh.air, lty = 1, col = "green")
R> lines(x.air, y.lfdh.air, lty = 2, col = "magenta")
R> legend("topleft", legend = c("DEA", "FDH", "LFDH"), bty = "n",
+   col = c("cyan", "green", "magenta"), lty = c(4, 1, 2))
R> points(ytab ~ xtab, data = air)
```

Polynomial estimators

The function `poly_est` is an implementation of the unconstrained polynomial-type estimators of Hall, Park, and Stern (1998) for support frontiers and boundaries.

Here, the data edge is modeled by a single polynomial $\varphi_\theta(x) = \theta_0 + \theta_1 x + \ldots + \theta_p x^p$ of known degree p that envelopes the full data and minimizes the area under its graph for $x \in [a, b]$, with a and b being respectively the lower and upper endpoints of the design points $x_1, \ldots, x_n$. The function is the estimate $\hat{\varphi}_{n,P}(x) = \hat{\theta}_0 + \hat{\theta}_1 x + \ldots + \hat{\theta}_p x^p$ of $\varphi(x)$, where $\hat{\theta} = (\hat{\theta}_0, \hat{\theta}_1, \ldots, \hat{\theta}_p)^\top$ minimizes $\int_a^b \varphi_\theta(x)\, dx$ over $\theta \in \mathbb{R}^{p+1}$ subject to the envelopment constraints $\varphi_\theta(x_i) \geq y_i$, $i = 1, \ldots, n$. The polynomial degree p has to be fixed by the user in the 4th argument of the function.

Selection of the polynomial degree

As the degreee p determines the dimensionality of the approximating function, we may view the problem of choosing p as model selection by calling the function `poly_degree`. By analogy to the information criteria proposed by Daouia *et al.* (2016) in the boundary regression

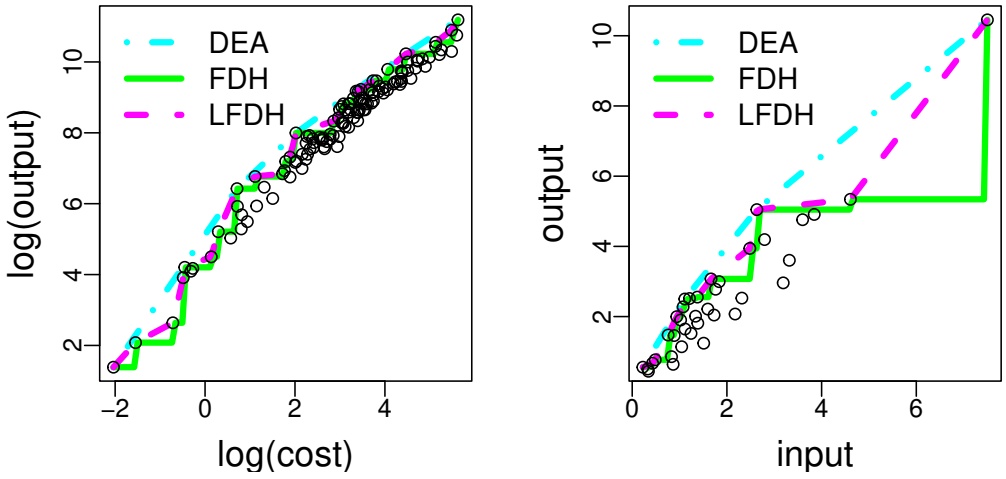

Figure 2: DEA, FDH and LFDH estimates of the optimal frontier for the 37 European air controllers (left) and the 123 American electric utility companies (right).

context, we obtain the optimal polynomial degree by minimizing

$$AIC(p) = \log\left(\sum_{i=1}^{n}(\hat{\varphi}_n(x_i) - y_i)\right) + (p+1)/n,$$
$$BIC(p) = \log\left(\sum_{i=1}^{n}(\hat{\varphi}_n(x_i) - y_i)\right) + \log n \cdot (p+1)/(2n).$$

The first one (option `type = "AIC"`) is similar to the famous Akaike information criterion (Akaike 1973) and the second one (option `type = "BIC"`) to the Bayesian information criterion (Schwartz 1978). They aim to balance the fidelity to data and the complexity of the fit in the boundary regression context. There are several ways to motivate the use of the total absolute residuals in these criteria instead of the standard residual sum of squares. For instance, it can be derived directly assuming exponential errors as motivated by Daouia *et al.* (2016) in Section 2.1.

Practical guidelines

By way of example, we consider the `records`, `air` and `nuclear` datasets. To determine the optimal polynomial degrees via the AIC criterion, we employ the commands

```
R> (p.aic.records <- poly_degree(records$year, 1/records$result,
+   prange = 0:12, type = "AIC"))

[1] 11

R> (p.aic.air <- poly_degree(air$xtab, air$ytab, type = "AIC"))

[1] 3
```

```
R> (p.aic.nuc <- poly_degree(nuclear$xtab, nuclear$ytab, type = "AIC"))

[1] 2
```

We find the same degrees by applying the BIC criterion. The R specifications for the corresponding polynomial boundaries to be estimated are given by

```
R> x.records<-seq(min(records$year), max(records$year), length.out = 101)
R> y.poly.records <- poly_est(records$year, 1/records$result, x.records,
+   deg = p.aic.records)
R> y.poly.air <- poly_est(air$xtab, air$ytab, x.air, deg = p.aic.air)
R> x.nucl <- seq(min(nuclear$xtab), max(nuclear$xtab), length.out = 101)
R> y.poly.nuc <- poly_est(nuclear$xtab, nuclear$ytab, x.nucl,
+   deg = p.aic.nuc)
```

The following code can be used to construct the plots of the resulting estimators appearing in Figure 3.

```
R> plot(x.records, 1/y.poly.records, type = "l", col = "green")
R> points(result ~ year, data = records)
R> legend("bottomleft", legend = paste("degree =", p.aic.records),
+   col = "green", lty = 1, bty = "n")
R> plot(x.air, y.poly.air, type = "l", col = "magenta")
R> points(ytab ~ xtab, data = air)
R> legend("topleft", legend = paste("degree =", p.aic.air),
+   col = "magenta", lty = 1, bty = "n")
R> plot(y.poly.nuc ~ x.nucl, type = "l", col = "cyan",
+   ylim = range(nuclear$ytab))
R> points(ytab ~ xtab, data = nuclear)
R> legend("topleft", legend = paste("degree =", p.aic.nuc),
+   col = "cyan", lty = 1, bty = "n")
```

Quadratic spline smoothers

The function `quad_spline_est` is an implementation of the (un)constrained quadratic spline estimates proposed by Daouia *et al.* (2016).

Unconstrained quadratic fit

Let a and b be, respectively, the minimum and maximum of the design points $x_1, \ldots, x_n$. Denote a partition of $[a, b]$ by $a = t_0 < t_1 < \ldots < t_{k_n} = b$ (see below the selection process of k_n and $\{t_j\}$). Let $N = k_n + 2$ and $\pi(x) = (\pi_0(x), \ldots, \pi_{N-1}(x))^\top$ be the vector of normalized B-splines of order 3 based on the knot mesh $\{t_j\}$ (see, *e.g.*, Schumaker 2007). The unconstrained (option `method = "u"`) quadratic spline estimate of the frontier function $\varphi(x)$ is defined as $\tilde{\varphi}_n(x) = \pi(x)^\top \tilde{\alpha}$, where $\tilde{\alpha}$ minimizes $\int_0^1 \pi(x)^\top \alpha \, dx = \sum_{j=0}^{N-1} \alpha_j \int_0^1 \pi_j(x) \, dx$ over $\alpha \in \mathbb{R}^N$ subject to the envelopment constraints $\pi(x_i)^\top \alpha \geq y_i$, $i = 1, \ldots, n$. A simple way of choosing the knot mesh in this unconstrained setting is by considering the j/k_nth quantiles $t_j = x_{[jn/k_n]}$

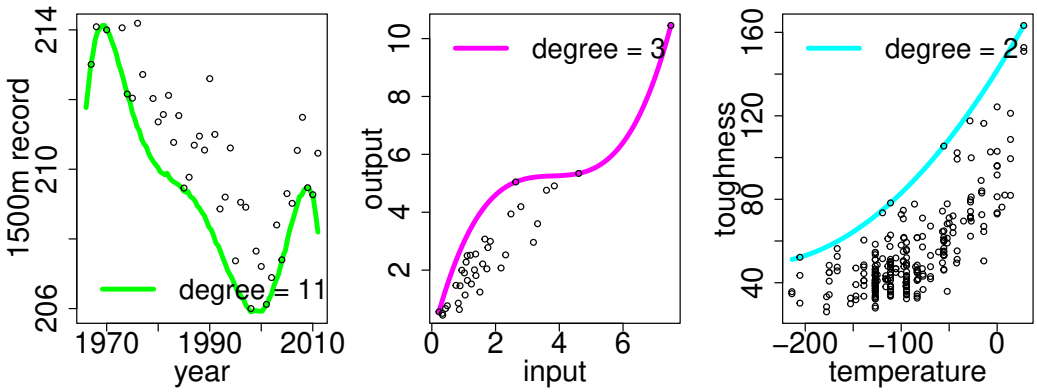

Figure 3: Polynomial boundary estimators for the 46 annual sport records (left), the 37 European air controllers (middle) and the 254 nuclear reactors' data (right).

of the distinct values of x_i for $j = 1, \ldots, k_n - 1$. Then, the choice of the number of inter-knot segments k_n is viewed as model selection by making use of the function `quad_spline_kn` (option `method = "u"`) described in a separate paragraph below.

Monotonicity constraint

When the true frontier $\varphi(x)$ is known or required to be monotone nondecreasing (option `method = "m"`), its constrained quadratic spline estimate is defined by $\hat{\varphi}_n(x) = \pi(x)^\top \hat{\alpha}$, where $\hat{\alpha}$ minimizes the same objective function as $\tilde{\alpha}$ subject to the same envelopment constraints and the additional monotonicity constraints $\pi'(t_j)^\top \alpha \geq 0$, $j = 0, 1, \ldots, k_n$, with π' being the derivative of π. Considering the special connection of the spline smoother $\hat{\varphi}_n$ with the traditional FDH frontier φ_{fdh} (see the function `dea_est`), Daouia *et al.* (2016) propose a refined way of choosing the knot mesh. Let $(\mathcal{X}_1, \mathcal{Y}_1), \ldots, (\mathcal{X}_\mathcal{N}, \mathcal{Y}_\mathcal{N})$ be the observations (x_i, y_i) lying on the FDH boundary (*i.e.*, $y_i = \varphi_{fdh}(x_i)$). The basic idea is to pick out a set of knots equally spaced in percentile ranks among the $\mathcal{N}$ FDH points $(\mathcal{X}_\ell, \mathcal{Y}_\ell)$ by taking $t_j = \mathcal{X}_{[j\mathcal{N}/k_n]}$, the j/k_nth quantile of the values of $\mathcal{X}_\ell$ for $j = 1, \ldots, k_n - 1$. The optimal number k_n is then obtained by using the function `quad_spline_kn` (option `method = "m"`).

Concavity constraint

When the monotone boundary $\varphi(x)$ is also believed to be concave (option `method = "mc"`), its constrained fit is defined as $\hat{\varphi}^\star_n(x) = \pi(x)^\top \hat{\alpha}^\star$, where $\hat{\alpha}^\star \in \mathbb{R}^N$ minimizes the same objective function as $\hat{\alpha}$ subject to the same envelopment and monotonicity constraints and the additional concavity constraints $\pi''(t^*_j)^\top \alpha \leq 0$, $j = 1, \ldots, k_n$, where π'' is the constant second derivative of π on each inter-knot interval and t^*_j is the midpoint of $(t_{j-1}, t_j]$. Regarding the choice of knots, the same scheme as for $\hat{\varphi}_n$ is applied by replacing the FDH points $(\mathcal{X}_1, \mathcal{Y}_1), \ldots, (\mathcal{X}_\mathcal{N}, \mathcal{Y}_\mathcal{N})$ with the DEA points $(\mathcal{X}^*_1, \mathcal{Y}^*_1), \ldots, (\mathcal{X}^*_\mathcal{M}, \mathcal{Y}^*_\mathcal{M})$, that is, the observations $(x_i, y_i = \varphi_{dea}(x_i))$ lying on the piecewise linear DEA frontier (see the function `dea_est`). Alternatively, the strategy of just using all the DEA points as knots also works quite well for datasets of modest size as shown in Daouia *et al.* (2016). In this case, the user

has to choose the option `all.dea = TRUE`.

Optimal number of inter-knot segments

The function `quad_spline_kn` computes the optimal number k_n for the quadratic spline fits proposed by Daouia *et al.* (2016). For the implementation of the unconstrained quadratic spline smoother $\tilde{\varphi}_n$, based on the knot mesh $\{t_j = x_{[jn/k_n]} : j = 1, \ldots, k_n - 1\}$, the user has to employ the option `method = "u"`. Since the number k_n determines the complexity of the spline approximation, its choice may be viewed as model selection via the minimization of the following Akaike (option `type = "AIC"`) or Bayesian (option `type = "BIC"`) information criteria:

$$\begin{aligned} A\tilde{I}C(k) &= \log\left(\sum_{i=1}^{n}(\tilde{\varphi}_n(x_i) - y_i)\right) + (k+2)/n, \\ B\tilde{I}C(k) &= \log\left(\sum_{i=1}^{n}(\tilde{\varphi}_n(x_i) - y_i)\right) + \log n \cdot (k+2)/(2n). \end{aligned}$$

For the implementation of the monotone (option `method = "m"`) quadratic spline smoother $\hat{\varphi}_n$, the authors first suggest using the set of knots $\{t_j = \mathcal{X}_{[j\mathcal{N}/k_n]},\ j = 1, \ldots, k_n - 1\}$ among the FDH points $(\mathcal{X}_\ell, \mathcal{Y}_\ell)$, $\ell = 1, \ldots, \mathcal{N}$, as described above. Then, they propose to choose k_n by minimizing the following AIC (option `type = "AIC"`) or BIC (option `type = "BIC"`) information criteria:

$$\begin{aligned} A\hat{I}C(k) &= \log\left(\sum_{i=1}^{n}(\hat{\varphi}_n(x_i) - y_i)\right) + (k+2)/n, \\ B\hat{I}C(k) &= \log\left(\sum_{i=1}^{n}(\hat{\varphi}_n(x_i) - y_i)\right) + \log n \cdot (k+2)/(2n). \end{aligned}$$

A small number of knots is typically needed as elucidated by the asymptotic theory.

For the implementation of the monotone and concave (option `method = "mc"`) spline estimator $\hat{\varphi}_n^\star$, just apply the same scheme as above by replacing the FDH points $(\mathcal{X}_\ell, \mathcal{Y}_\ell)$ with the DEA points $(\mathcal{X}_\ell^*, \mathcal{Y}_\ell^*)$.

Practical guidelines

We describe here how to construct the necessary computations of the (un)constrained quadratic spline fits under both separate and simultaneous shape constraints. By way of example we consider the `air` and `green` data. To conduct the unconstrained estimation, we first determine the optimal number of inter-knot segments via the BIC criterion.

```
R> (kn.bic.air.u <- quad_spline_kn(air$xtab, air$ytab,
+   method = "u", type = "BIC"))

[1] 12

R> (kn.bic.green.u <- quad_spline_kn(log(green$COST), log(green$OUTPUT),
+   method = "u", type = "BIC"))
```

```
[1] 14
```

When applying the AIC criterion, we get the optimal values 12 and 20 of k_n, respectively. The R specification for the unconstrained spline estimate $\tilde{\varphi}_n$ to be calculated is given by

```
R> y.quad.air.u <- quad_spline_est(air$xtab, air$ytab, x.air,
+   kn = kn.bic.air.u, method = "u")
R> y.quad.green.u <- quad_spline_est(log(green$COST), log(green$OUTPUT),
+   x.green, kn = kn.bic.green.u, method = "u")
```

When only the monotonicity constraint is of interest, we calculate the optimal number k_n via the following specification:

```
R> (kn.bic.air.m <- quad_spline_kn(air$xtab, air$ytab,
+   method = "m", type = "BIC"))
```

```
[1] 6
```

```
R> (kn.bic.green.m <- quad_spline_kn(log(green$COST), log(green$OUTPUT),
+   method = "m", type = "BIC"))
```

```
[1] 10
```

Note that we find the values 6 and 19 of the optimal number k_n when applying the AIC criterion. The monotonic spline $\hat{\varphi}_n$ can then be produced by employing the command

```
R> y.quad.air.m <- quad_spline_est(air$xtab, air$ytab, x.air,
+   kn = kn.bic.air.m, method = "m")
R> y.quad.green.m <- quad_spline_est(log(green$COST), log(green$OUTPUT),
+   x.green, kn = kn.bic.green.m, method = "m")
```

When the concavity constraint is also of interest, we obtain the optimal number k_n via the BIC criterion and the corresponding constrained spline $\hat{\varphi}_n^\star$ by proceeding as follows:

```
R> (kn.bic.air.mc <- quad_spline_kn(air$xtab, air$ytab,
+   method = "mc", type = "BIC"))
```

```
[1] 2
```

```
R> (kn.bic.green.mc <- quad_spline_kn(log(green$COST), log(green$OUTPUT),
+   method = "mc", type = "BIC"))
```

```
[1] 1
```

When applying the AIC criterion, we get the optimal values 2 and 7 of k_n, respectively. To compute the smoother $\hat{\varphi}_n^\star$ by utilizing all the DEA points as knots, we use the command

```
R> y.quad.air.mc <- quad_spline_est(air$xtab, air$ytab, x.air,
+   kn = kn.bic.air.mc, method = "mc", all.dea = TRUE)
R> y.quad.green.mc <- quad_spline_est(log(green$COST), log(green$OUTPUT),
+   x.green, kn = kn.bic.green.mc, method = "mc", all.dea = TRUE)
```

The resulting unrestricted and two constrained estimates of the econometric frontiers (*i.e.*, the sets of the most efficient companies and controllers) are graphed in Figure 4 for each dataset. The following code will generate Figure 4.

```
R> plot(y.quad.air.u ~ x.air, lty = 1, col = "green", type = "l",
+   xlab = "input", ylab = "output")
R> lines(x.air, y.quad.air.m, lty = 2, col = "cyan")
R> lines(x.air, y.quad.air.mc, lty = 3, col = "magenta")
R> points(ytab ~ xtab, data = air)
R> legend("topleft", col = c("green", "cyan", "magenta"),
+   lty = c(1, 2, 3), bty = "n", lwd = 4,
+   legend = c("unconstrained", "monotone", "monotone + concave"))
R> plot(y.quad.green.u ~ x.green, lty = 1, col = "green", type = "l",
+   xlab = "log(COST)", ylab = "log(OUTPUT)")
R> lines(x.green, y.quad.green.m, lty = 2, col = "cyan")
R> lines(x.green, y.quad.green.mc, lty = 3, col = "magenta")
R> points(log(OUTPUT) ~ log(COST), data = green)
R> legend("topleft", col = c("green", "cyan", "magenta"),
+   bty = "n", lty = c(1, 2, 3),
+   legend = c("unconstrained", "monotone", "monotone + concave"))
```

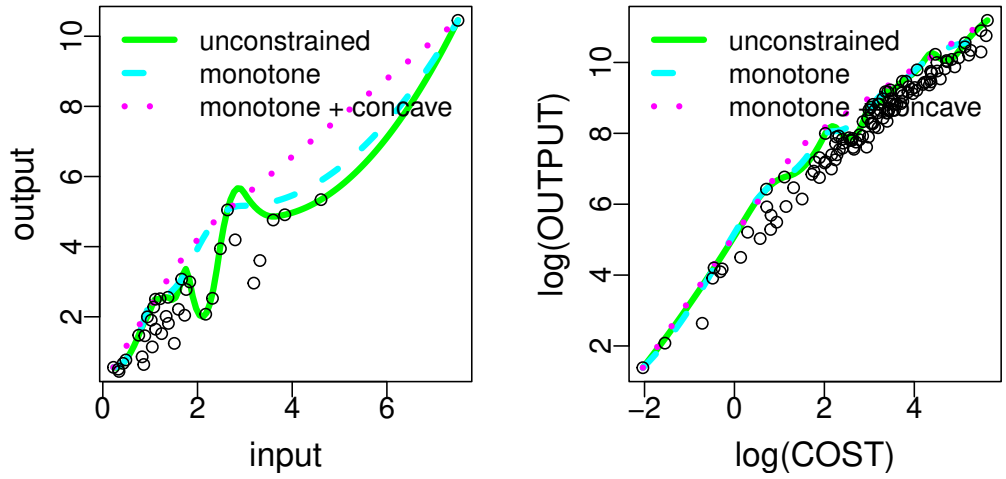

Figure 4: The quadratic spline frontiers $\tilde{\varphi}_n$, $\hat{\varphi}_n$ and $\hat{\varphi}_n^\star$ for the 37 European air controllers (left) and the 123 American electric utility companies (right).

Cubic spline frontiers

The function `cub_spline_est` is an implementation of the (un)constrained cubic spline estimates proposed by Daouia *et al.* (2016).

As in the quadratic spline setting, let a and b be respectively the minimum and maximum of the design points $x_1, \ldots, x_n$, and denote a partition of $[a, b]$ by $a = t_0 < t_1 < \ldots < t_{k_n} = b$. Here, $N = k_n + 3$ and $\pi(x) = (\pi_0(x), \ldots, \pi_{N-1}(x))^\top$ is the vector of normalized B-splines of order 4 based on the knot mesh $\{t_j\}$. The unconstrained (option `method = "u"`) cubic spline estimate of the frontier $\varphi(x)$ is then defined in the same way as the envelopment quadratic spline $\tilde{\varphi}_n(x)$ with the same knot selection process, that is, $t_j = x_{[jn/k_n]}$ is the j/k_nth quantile of the distinct values of x_i for $j = 1, \ldots, k_n - 1$. The number of inter-knot segments k_n is obtained by calling the function `cub_spline_kn` (option `method = "u"`), which consists in minimizing the information criterion $A\tilde{I}C(k)$ (option `type = "AIC"`) or $B\tilde{I}C(k)$ (option `type = "BIC"`).

Regarding the monotonicity constraint, it cannot be formulated into linear constraints at the knots since, as opposed to quadratic splines, the first derivative of cubic splines is a quadratic spline. Daouia *et al.* (2016) have been able to come up with an alternative formulation of monotonicity in terms of standard second-order cone constraints, but in our R package for computational convenience we use the following sufficient condition to ensure monotonicity:

$$\alpha_0 \leq \alpha_1 \leq \ldots \leq \alpha_{N-1}.$$

This condition was previously used in Lu, Zhang, and Huang (2007) and Pya and Wood (2014). Note that since the condition corresponds to linear constraints on α, the estimator satisfying the monotonicity constraint can be obtained via linear programming.

When the estimate is required to be both monotone and concave, we use the function `cub_spline_est` with the option `method = "mc"`. The estimate is obtained as the cubic spline function which minimizes the same linear objective function as the unconstrained estimate subject to the same linear envelopment constraints, the monotonicity constraint above and the additional linear concavity constraints $\pi''(t_j)^\top \alpha \leq 0$, $j = 0, 1, \ldots, k_n$, where the second derivative π'' is a linear spline. Regarding the choice of knots, we just apply the same scheme as for the unconstrained cubic spline estimate.

By way of example we consider again the `air` and `green` data. We first calculate the optimal numbers k_n via the BIC criterion:

```
R> (kn.bic.air.u <- cub_spline_kn(air$xtab, air$ytab, method = "u",
+   type = "BIC"))

[1] 1

R> (kn.bic.green.u <- cub_spline_kn(log(green$COST), log(green$OUTPUT),
+   method = "u", type = "BIC"))

[1] 8
```

```
R> (kn.bic.air.m <- cub_spline_kn(air$xtab, air$ytab, method = "m",
+   type = "BIC"))

[1] 7

R> (kn.bic.green.m <- cub_spline_kn(log(green$COST), log(green$OUTPUT),
+   method = "m", type = "BIC"))

[1] 12

R> (kn.bic.air.mc <- cub_spline_kn(air$xtab, air$ytab,
+   method = "mc", type = "BIC"))

[1] 3

R> (kn.bic.green.mc <- cub_spline_kn(log(green$COST), log(green$OUTPUT),
+   method = "mc", type = "BIC"))

[1] 5
```

Note that we find the same values by applying the AIC criterion. To compute the corresponding (un)constrained cubic spline frontiers, we employ the following commands

```
R> y.cub.air.u <- cub_spline_est(air$xtab, air$ytab, x.air,
+    kn = kn.bic.air.u, method = "u")
R> y.cub.green.u <- cub_spline_est(log(green$COST), log(green$OUTPUT),
+   x.green, kn = kn.bic.green.u, method = "u")
R> y.cub.air.m <- cub_spline_est(air$xtab, air$ytab, x.air,
+   kn = kn.bic.air.m, method = "m")
R> y.cub.green.m <- cub_spline_est(log(green$COST), log(green$OUTPUT),
+   x.green, kn = kn.bic.green.m, method = "m")
R> y.cub.air.mc <- cub_spline_est(air$xtab, air$ytab, x.air,
+   kn = kn.bic.air.mc, method = "mc")
R> y.cub.green.mc <- cub_spline_est(log(green$COST), log(green$OUTPUT),
+   x.green, kn = kn.bic.green.mc, method = "mc")
```

The resulting unconstrained and concave frontier estimates are graphed in Figure 5 for each dataset. The following code will generate Figure 5.

```
R> plot(y.cub.air.u ~ x.air, type = "l", col = "green",
+   xlab = "input", ylab = "output")
R> lines(x.air, y.cub.air.m, lty = 2, col = "cyan")
R> lines(x.air, y.cub.air.mc, lty = 3, col = "magenta")
R> points(ytab ~ xtab, data = air)
R> legend("topleft", col = c("green", "cyan", "magenta"), lty = c(1, 2, 3),
+    bty = "n", legend=c("unconstrained", "monotone", "monotone+concave"))
R> plot(y.cub.green.u ~ x.green, type = "l", col = "green",
```

```
+   xlab = "log(COST)", ylab = "log(OUTPUT)")
R> lines(x.green, y.cub.green.m, lty = 2,  col = "cyan")
R> lines(x.green, y.cub.green.mc, lty = 3, col = "magenta")
R> points(log(OUTPUT) ~ log(COST), data = green)
R> legend("topleft", col = c("green", "cyan", "magenta"), lty = c(1, 2, 3),
+    bty = "n", legend = c("unconstrained", "monotone", "monotone+concave"))
```

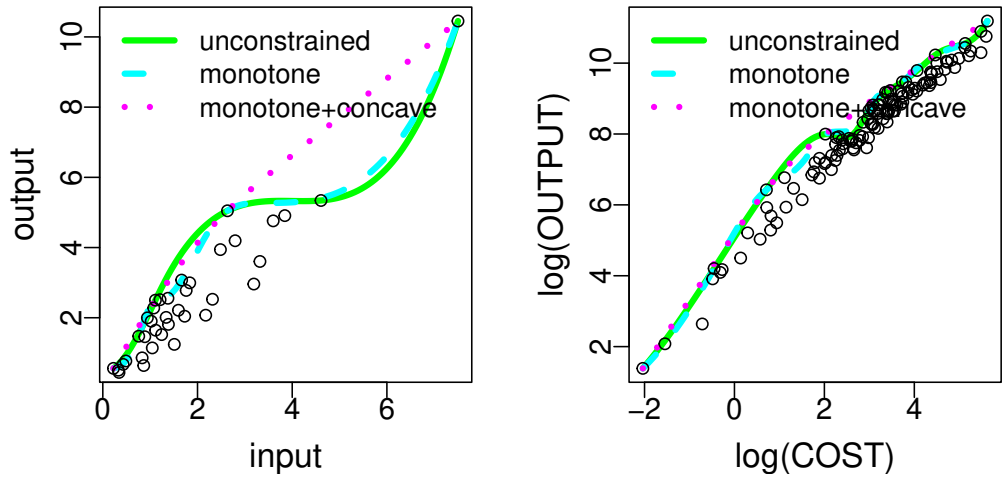

Figure 5: The unconstrained and concave cubic spline frontiers for the 37 European air controllers (left) and the 123 American electric utility companies (right).

3.2. Localized boundary regression

This section is concerned with localizing the frontier estimation and considers local linear fitting (Hall *et al.* 1998; Hall and Park 2004), local maximum and extreme-value smoothing (Gijbels and Peng 2000).

Local linear fitting

The function `loc_est` computes the local linear smoothing frontier estimators of Hall, Park, and Stern (1998) and Hall and Park (2004). In the unconstrained case (option `method = "u"`), the implemented estimator of $\varphi(x)$ is defined by

$$\begin{aligned}\hat{\varphi}_{n,LL}(x) \quad = \quad & \min\Big\{z : \text{there exists } \theta \text{ such that } y_i \leq z + \theta(x_i - x) \\ & \text{for all } i \text{ such that } x_i \in (x-h, x+h)\Big\},\end{aligned}$$

where the bandwidth h has to be fixed by the user in the 4th argument of the function. This estimator may lack of smoothness in case of small samples and has no guarantee of being monotone even if the true frontier is so. Following the curvature of the monotone frontier φ, the unconstrained estimator $\hat{\varphi}_{n,LL}$ is likely to exhibit substantial bias, especially at the sample

boundaries. A simple way to remedy to this drawback is by imposing the extra condition $\theta \geq 0$ in the definition of $\hat{\varphi}_{n,LL}(x)$ to get

$$\begin{aligned}\tilde{\varphi}_{n,LL}(x) &= \min\Big\{z : \text{there exists } \theta \geq 0 \text{ such that } y_i \leq z + \theta_1(x_i - x) \\ &\qquad \text{for all } i \text{ such that } x_i \in (x-h, x+h)\Big\}.\end{aligned}$$

As shown in Daouia *et al.* (2016), this version only reduces the vexing bias and border defects of the original estimator when the true frontier is monotone. The option `method = "m"` indicates that the improved fit $\tilde{\varphi}_{n,LL}$ should be utilized in place of $\hat{\varphi}_{n,LL}$.

Optimal bandwidth choice

Hall and Park (2004) proposed a bootstrap procedure for selecting the optimal bandwidth h in $\hat{\varphi}_{n,LL}$ and $\tilde{\varphi}_{n,LL}$. The function `loc_est_bw` computes this optimal bootstrap bandwidth. To initiate Hall and Park's bootstrap device, one needs to set a pilot bandwidth, which seems to be quite critical to the quality of $\hat{\varphi}_{n,LL}$ and $\tilde{\varphi}_{n,LL}$.

Practical guidelines

To see how the local linear unconstrained estimate $\hat{\varphi}_{n,LL}$ and its improved version $\tilde{\varphi}_{n,LL}$ perform in the case of `records`, `air` and `nuclear` data. We first compute the optimal bandwidths over 100 bootstrap replications by using, for instance, the values 2, 2 and 40 as pilot bandwidths.

```
R> h.records.u <- loc_est_bw(records$year, 1/records$result, x.records,
+   hini = 2, B = 100, method = "u")

[1] 22.5

R> h.air.u <- loc_est_bw(air$xtab, air$ytab, x.air,
+   hini = 2, B = 100, method = "u")

[1] 2.89278

R> h.air.m <- loc_est_bw(air$xtab, air$ytab, x.air,
+   hini = 2, B = 100, method = "m")

[1] 3.586696

R> h.nucl.u <- loc_est_bw(nuclear$xtab, nuclear$ytab, x.nucl,
+   hini = 40, B = 100, method = "u")

[1] 82.32759

R> h.nucl.m <- loc_est_bw(nuclear$xtab, nuclear$ytab, x.nucl,
+   hini = 40, B = 100, method = "m")
```

```
[1] 82.32759
```

Note that the computational burden here is very demanding, so be forewarned. Now to evaluate $\hat{\varphi}_{n,LL}$ and/or $\tilde{\varphi}_{n,LL}$, we employ the commands

```
R> y.records.u <- loc_est(records$year, 1/records$result, x.records,
+   h = h.records.u, method = "u")
R> y.air.u <- loc_est(air$xtab, air$ytab, x.air, h = h.air.u, method = "u")
R> y.air.m <- loc_est(air$xtab, air$ytab, x.air, h = h.air.m, method = "m")
R> y.nucl.u <- loc_est(nuclear$xtab, nuclear$ytab, x.nucl,
+   h = h.nucl.u, method = "u")
R> y.nucl.m <- loc_est(nuclear$xtab, nuclear$ytab, x.nucl,
+   h = h.nucl.m, method = "m")
```

Figure 6 superimposes the obtained estimates for each dataset. For the particular datasets `air` and `nuclear`, the resulting unconstrained and improved estimates are very similar. The following code will generate Figure 6.

```
R> plot(x.records, 1/y.records.u, type = "l", col = "magenta")
R> points(result ~ year, data = records)
R> legend("topright", legend = "unconstrained",  bty = "n",
+   col = "magenta", lty = 1)
R> plot(y.air.u ~ x.air, type = "l", col = "magenta")
R> lines(x.air, y.air.m, lty = 2, col = "cyan")
R> points(ytab ~ xtab, data = air)
R> legend("topleft", legend = c("unconstrained", "improved"), bty = "n",
+   col = c("magenta", "cyan"), lty = c(1, 2))
R> plot(y.nucl.u ~ x.nucl, type = "l", col = "magenta")
R> lines(x.nucl, y.nucl.m, lty = 2, col = "cyan")
R> points(ytab ~ xtab, data = nuclear)
R> legend("topleft", legend = c("unconstrained", "improved"),  bty = "n",
+   col = c("magenta", "cyan"), lty = c(1, 2))
```

Local maximum estimation

The function `loc_max` implements the local maximum estimates of $\varphi(x)$ proposed by Gijbels and Peng (2000): a local constant estimator at first (option `type = "one-stage"`) and subsequently a local DEA estimator (option `type = "two-stage"`).

The methodology of Gijbels and Peng consists of considering a strip around x of width $2h$, where $h = h_n \to 0$ with $nh_n \to \infty$ as $n \to \infty$, and focusing then on the y_i values of observations falling into this strip. More precisely, they consider the transformend variables $z_i^{xh} = y_i \mathbb{1}_{\{|x_i - x| \leq h\}}$, $i = 1, \ldots, n$, and the corresponding order statistics $z_{(1)}^{xh} \leq \ldots \leq z_{(n)}^{xh}$. The simple maximum $z_{(n)}^{xh} = \max_{i=1,\ldots,n} z_i^{xh}$ defines then the local constant estimator (option `type = "one-stage"`) of the frontier point $\varphi(x)$. This opens a way to a two-stage estimation procedure as follows. In a first stage, Gijbels and Peng calculate the maximum $z_{(n)}^{xh}$. Then,

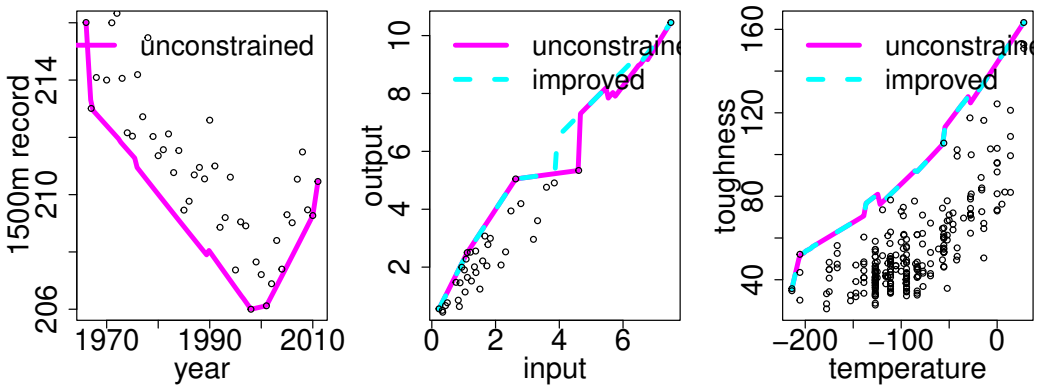

Figure 6: Local linear frontier estimates $\hat{\varphi}_{n,LL}$ and $\tilde{\varphi}_{n,LL}$ for the 46 annual sport records (left), the 37 European air controllers (middle) and the 254 nuclear reactors (right).

they suggest to replace each observation y_i in the strip of width $2h$ around x by this maximum, leaving all observations outside the strip unchanged. More specifically, they define

$$\tilde{y}_i = \begin{cases} y_i & \text{if} \quad |x_i - x| > h \\ z_{(n)}^{xh} & \text{if} \quad |x_i - x| \leq h. \end{cases}$$

Then, they apply the DEA estimator (see the function `dea_est`) to these transformed data $(x_i, \tilde{y}_i)$, giving the local DEA estimator (option `type = "two-stage"`).

The bandwidth h has to be fixed by the user in the 4th argument of the function. By way of example, in the case of the `green` data, the value $h = 0.5$ leads to reproduce in Figure 7 (left) the estimates obtained by Gijbels and Peng (2000).

```
R> loc_max_1stage <- loc_max(log(green$COST), log(green$OUTPUT), x.green,
+   h = 0.5, type = "one-stage")
R> loc_max_2stage <- loc_max(log(green$COST), log(green$OUTPUT), x.green,
+   h = 0.5, type = "two-stage")
```

A data-driven rule for selecting h

Note that the frontier point $\varphi(x)$ is identical to the right-endpoint of the cumulative distribution function $F(\cdot|x)$ of Y given $X = x$, and that the local constant estimate $z_{(n)}^{xh}$ coincides with the right-endpoint of the kernel estimator

$$F_n(y|x) = \sum_{i=1}^{n} K(\frac{x - x_i}{h}) \mathbb{1}_{(y_i \leq y)} / \sum_{i=1}^{n} K(\frac{x - x_i}{h}),$$

with $K(\cdot)$ being the uniform kernel. When the interest is in the estimation of the conditional distribution function, one way to select the bandwidth h is by making use of the following commands

```
R> require("np")
R> bw <- npcdistbw(log(OUTPUT) ~ log(COST), data = green,
+   cykertype = "uniform", bwtype = "fixed")$xbw
R> (h.opt <- max(bw, max(diff(sort(log(green$COST))))/2))

[1] 0.4152283
```

The first command returns the bandwidth *bw* computed via the least squares cross-validation method (see Li, Lin, and Racine 2013, for details). As the resulting bandwidth can be smaller than half the maximum spacing due to sparsity in data, the second command selects the maximum value. On may then use this value to compute the estimates of the conditional endpoint $\varphi(x)$ itself. This is an *ad hoc* choice, but it works quite well. It might be viewed as an exploratory tool, rather than as a method for final analysis. The corresponding local maximum frontier estimates are graphed in Figure 7 (right).

```
R> loc_max_1stage.opt <- loc_max(log(green$COST), log(green$OUTPUT), x.green,
+   h = h.opt, type = "one-stage")
R> loc_max_2stage.opt <- loc_max(log(green$COST), log(green$OUTPUT), x.green,
+   h = h.opt, type = "two-stage")
```

The following code will generate Figure 7.

```
R> plot(log(OUTPUT) ~ log(COST), data = green)
R> lines(x.green, loc_max_1stage, lty = 1, col = "magenta")
R> lines(x.green, loc_max_2stage, lty = 2, col = "cyan")
R> legend("topleft", legend = c("one-stage", "two-stage"), bty = "n",
+   col = c("magenta", "cyan"), lty = c(1, 2))
R> plot(log(OUTPUT) ~ log(COST), data = green)
R> lines(x.green, loc_max_1stage.opt, lty = 1, col = "magenta")
R> lines(x.green, loc_max_2stage.opt, lty = 2, col = "cyan")
R> legend("topleft",legend = c("one-stage", "two-stage"),  bty = "n",
+   col = c("magenta", "cyan"), lty = c(1, 2))
```

Local extreme-value estimation

The function `pick_est` computes the local Pickands type of estimator introduced by Gijbels and Peng (2000). The implemented estimator of $\varphi(x)$, obtained by applying the well-known extreme value approach of Dekkers *et al.* (1989) in conjunction with the transformed sample $(z_1^{xh}, \ldots, z_n^{xh})$ described above in Section 3.2.2, is defined as:

$$\tilde{\varphi}_{pick}(x) := z^{xh}_{(n-k)} + \left(z^{xh}_{(n-k)} - z^{xh}_{(n-2k)}\right) \left\{ 2^{-\log \frac{z^{xh}_{(n-k)} - z^{xh}_{(n-2k)}}{z^{xh}_{(n-2k)} - z^{xh}_{(n-4k)}} / \log 2} - 1 \right\}^{-1}.$$

It is based on three upper order statistics $z^{xh}_{(n-k)}$, $z^{xh}_{(n-2k)}$, $z^{xh}_{(n-4k)}$, and depends on the bandwidth h as well as an intermediate sequence $k = k(n) \to \infty$ with $k/n \to 0$ as $n \to \infty$. The

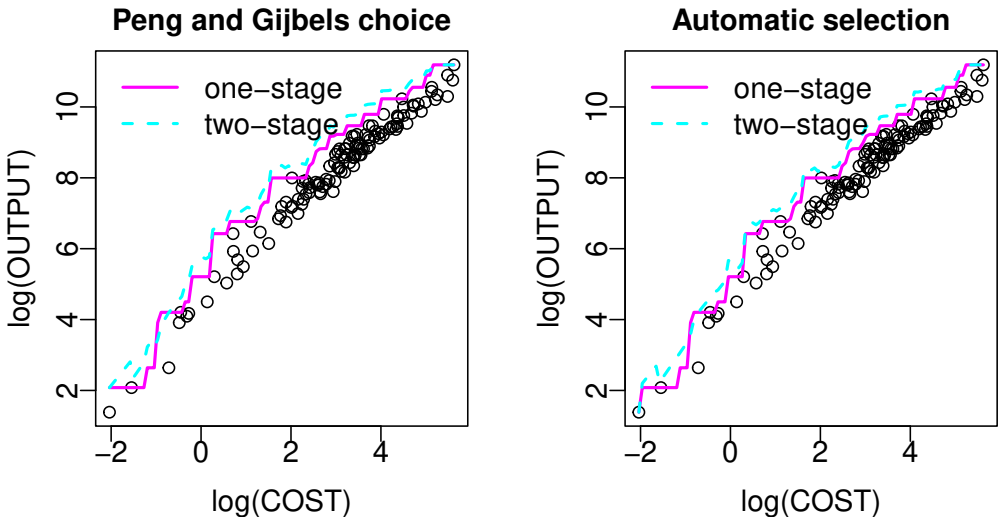

Figure 7: Local maximum frontier estimates for the 123 American electric utility companies with $h = 0.5$ (left) and $h.opt = 0.4152283$ (right).

two smoothing parameters h and k have to be fixed by the user in the 4th and 5th arguments of the function. Also, as for the two-stage local frontier estimator presented above, writing

$$\tilde{y}_i = \begin{cases} y_i & \text{if} \quad |x_i - x| > h \\ \tilde{\varphi}_{pick}(x) & \text{if} \quad |x_i - x| \leq h, \end{cases}$$

one can then apply the DEA estimator to these transformed data $(x_i, \tilde{y}_i)$, giving thus the local DEA estimator (option `type = "two-stage"`).

Regarding the choice of the smoothing parameters, it should be clear that any automatic data-driven method has to pick up h and k simultaneously, which is a daunting problem. Doubtlessly, further work to define a concept of selecting appropriate values for h and k will yield new refinements.

3.3. Kernel smoothing

Recently, kernel smoothing methods have been developed for estimating smooth frontier functions. The function `kern_smooth` implements two up-to-date approaches in such direction.

Parmeter and Racine's estimator

The function `kern_smooth` computes Parmeter and Racine (2013)'s estimator (option `technique = "pr"`) without constraints (option `method = "u"`), and with the monotonicity constraint (option `method = "m"`) as well as the monotone concavity constraint (option `method = "mc"`).

Definition of the estimator

To estimate the frontier function, Parameter and Racine (2013) considered the following

generalization of linear regression smoothers $\hat{\varphi}(x|p) = \sum_{i=1}^{n} p_i A_i(x) y_i$, where $A_i(x)$ is the kernel weight function of x for the i-th data depending on x_i's and the sort of linear smoothers. For example, the Nadaraya-Watson kernel weights are $A_i(x) = K_i(x)/(\sum_{j=1}^{n} K_j(x))$, where $K_i(x) = h^{-1}K\{(x - x_i)/h\}$, with the kernel function K being a bounded and symmetric probability density, and h is a bandwidth. Then, the weight vector $p = (p_1, \ldots, p_n)^\top$ is chosen to minimize the distance $D(p) = (p - p_u)^\top (p - p_u)$ subject to the envelopment constraints and the choice of the shape constraints, where p_u is an n-dimensional vector with all elements being one. The envelopement and shape constraints are

$$\begin{aligned}
\hat{\varphi}(x_i|p) - y_i &= \sum_{i=1}^{n} p_i A_i(x_i) y_i - y_i \geq 0, \ i = 1, \ldots, n; \quad \text{(envelopment constraints)} \\
\hat{\varphi}^{(1)}(x|p) &= \sum_{i=1}^{n} p_i A_i^{(1)}(x) y_i \geq 0, \ x \in \mathcal{M}; \quad \text{(monotonocity constraints)} \\
\hat{\varphi}^{(2)}(x|p) &= \sum_{i=1}^{n} p_i A_i^{(2)}(x) y_i \leq 0, \ x \in \mathcal{C}, \quad \text{(concavity constraints)}
\end{aligned}$$

where $\hat{\varphi}^{(s)}(x|p) = \sum_{i=1}^{n} p_i A_i^{(s)}(x) y_i$ is the s-th derivative of $\hat{\varphi}(x|p)$, with $\mathcal{M}$ and $\mathcal{C}$ being the collections of points where monotonicity and concavity are imposed, respectively. In our implementation of the estimator, we simply take the entire dataset $\{(x_i, y_i), \ i = 1, \ldots, n\}$ to be $\mathcal{M}$ and $\mathcal{C}$ and, in case of small samples, we augment the sample points by an equispaced grid of length 201 over the observed support $[\min_i x_i, \max_i x_i]$ of X. For the weight $A_i(x)$, we use the Nadaraya-Watson weights.

Optimal bandwidth

Bandwidth selection is crucial to good performance of the frontier estimator as with other kernel smoothing estimators. Parmeter and Racine (2013)'s recommendation is to adapt the optimal bandwidth for mean regression curve estimation chosen by least squares cross-validation to the boundary regression context. This is implemented with `bw_method = "cv"` in the function `kern_smooth_bw`. We also refer to existing functions from the **np** (Hayfield and Racine 2008) and **quadprog** (Turlach and Weingessel 2013) packages that can be found at `http://socserv.mcmaster.ca/racinej/Gallery/Home.html`.

Noh's estimator

Noh (2014) considered the same generalization of linear smoothers $\hat{\varphi}(x|p)$ for frontier estimation, but with a different method for choosing the weight p. This is implemented in the function `kern_smooth` with option `technique = "noh"`.

Definition of the estimator

In contrast with Parmeter and Racine (2013), along with the same envelopment and shape constraints, the weight vector p is chosen to minimize the area under the estimator $\hat{\varphi}(x|p)$, that is $A(p) = \int_a^b \hat{\varphi}(x|p)dx = \sum_{i=1}^{n} p_i y_i \left(\int_a^b A_i(x)dx \right)$, where $[a, b]$ is the true support of X. In practice, we integrate over the observed support $[\min_i x_i, \max_i x_i]$ since the theoretic one

is unknown. In what concerns the kernel weights $A_i(x)$, we use the Priestley-Chao weights

$$A_i(x) = \begin{cases} 0 & , \ i = 1 \\ (x_i - x_{i-1})K_i(x) & , \ i \neq 1 \end{cases},$$

where it is assumed that the pairs (x_i, y_i) have been ordered so that $x_1 \leq \ldots \leq x_n$. The choice of such weights is motivated by their convenience for the evaluation of the integral $\int A_i(x)dx$.

Optimal bandwidth

Following Parmeter and Racine (2013)'s recommendation, we may use the resulting bandwidth from cross-validation for Noh (2014)'s estimator. Another option proposed by Noh (2014) is to select the bandwidth which minimizes a BIC-type criterion developed for frontier estimation. The criterion is the following:

$$BIC(h) = \log\left(\sum_{i=1}^{n}(\hat{\varphi}(x_i|\hat{p}(h)) - y_i)\right) + \frac{\log n \cdot tr(S(h))}{2n},$$

where $\hat{p}(h)$ is the chosen weight vector given the bandwidth h, and $tr(S(h))$ is the trace of the smoothing matrix

$$S(h) = \begin{pmatrix} A_1(x_1) & \ldots & A_n(x_1) \\ \vdots & \ddots & \vdots \\ A_1(x_n) & \ldots & A_n(x_n) \end{pmatrix}.$$

We refer to Noh (2014) for a thorough discussion of the rationale for this BIC-type criterion. The function `kern_smooth_bw` computes the optimal bandwidth from this criterion with option `bw_method = "bic"`.

Comparison between the two estimators

To illustrate the use of `kern_smooth` and compare the two estimators, we consider the `green` data and compute each estimator under the monotonicity constraint (option `method = "m"`). First, using the function `kern_smooth_bw` we compute the optimal bandwidth for each estimator.

```
R> require("np")
R> (h.pr.green.m <- kern_smooth_bw(log(green$COST), log(green$OUTPUT),
+   method = "m", technique = "pr", bw_method = "cv"))

[1] 0.8304566

R> (h.noh.green.m <- kern_smooth_bw(log(green$COST), log(green$OUTPUT),
+   method = "m", technique = "noh", bw_method = "bic"))

[1] 2.695624
```

To compute the estimators for the chosen bandwidths obeying the constraint , we employ the following commands:

```
R> y.pr.green.m <- kern_smooth(log(green$COST), log(green$OUTPUT), x.green,
+   h = h.pr.green.m, method = "m", technique = "pr")
R> y.noh.green.m <- kern_smooth(log(green$COST), log(green$OUTPUT), x.green,
+   h = h.noh.green.m, method = "m", technique = "noh")
```

The resulting two constrained estimates are graphed in Figure 8 from the following commands:

```
R> plot(log(OUTPUT) ~ log(COST), data = green, xlab = "log(COST)",
+   ylab = "log(OUTPUT)")
R> lines(x.green, y.pr.green.m, lty = 2, col = "blue")
R> lines(x.green, y.noh.green.m, lty = 3, col = "red")
R> legend("topleft",  bty = "n", legend = c("noh", "pr"),
+   col = c("red", "blue"), lty = c(3,2))
```

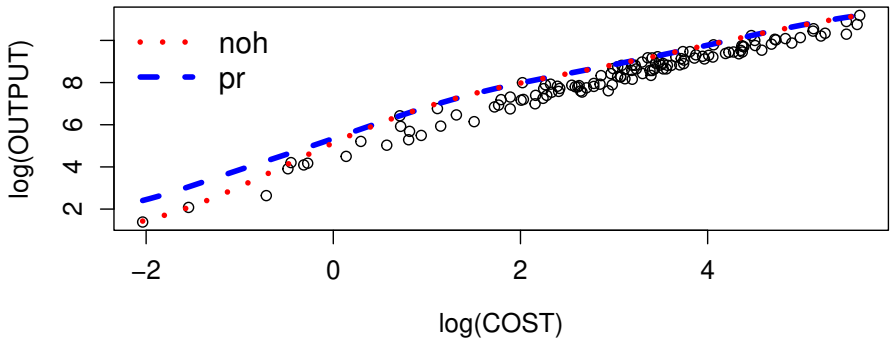

Figure 8: The two kernel smoothing frontier estimators for 123 American electric utility companies.

3.4. Robust regularization approaches

In applied settings where outlying observations are omnipresent, as is the case for instance in production data, it is prudent to seek a "robustification" strategy. To achieve this objective, we propose in this section three regularization extreme-value based methods (Daouia, Florens, and Simar 2010, 2012). All of these methods are based on the assumption that the frontier function φ is monotone nondecreasing.

Moment frontier estimator

The function `dfs_momt` is an implementation of the moment-type estimator and the corresponding confidence interval developed by Daouia *et al.* (2010) under the monotonicity constraint. Combining the ideas from Dekkers, Einmahl, and de Haan (1989) with the dimensionless transformation $\{z_i^x := y_i \mathbb{1}_{\{x_i \leq x\}}, i = 1, \ldots, n\}$ of the observed sample $\{(x_i, y_i), i =$

$1, \ldots, n\}$, they estimate the conditional endpoint $\varphi(x)$ by

$$\tilde{\varphi}_{momt}(x) = z^x_{(n-k)} + z^x_{(n-k)} M^{(1)}_n \{1 + \rho_x\}$$

where $M^{(1)}_n = (1/k) \sum_{i=0}^{k-1} \left(\log z^x_{(n-i)} - \log z^x_{(n-k)}\right)$, $z^x_{(1)} \leq \ldots \leq z^x_{(n)}$ are the ascending order statistics of the transformed sample $\{z^x_i,\, i = 1, \ldots, n\}$, and $\rho_x > 0$ is referred to as the extreme-value index and has the following interpretation: When $\rho_x > 2$, the joint density of data decays smoothly to zero at a speed of power $\rho_x - 2$ of the distance from the frontier; when $\rho_x = 2$, the density has sudden jumps at the frontier; when $\rho_x < 2$, the density increases toward infinity at a speed of power $\rho_x - 2$ of the distance from the frontier. As a matter of fact, we have $\rho_x = \beta_x + 2$, where β_x is the shape parameter of the joint density introduced in Section 1. Most of the contributions to the econometric literature on frontier analysis assume that the joint density is strictly positive at its support boundary, or equivalently, $\rho_x = 2$ for all x.

Estimation strategy when ρ_x is unknown

In this case, Daouia *et al.* (2010) suggest to use the following two-step estimator: First, estimate ρ_x by the moment estimator $\tilde{\rho}_x$ implemented in the function `rho_momt_pick` by utilizing the option `method = "moment"`, or by the Pickands estimator $\hat{\rho}_x$ by using the option `method = "pickands"` (see the paragraph **Moment and Pickands estimates of the tail-index** ρ_x below for a detailed description of the function `rho_momt_pick`). Second, use the estimator $\tilde{\varphi}_{momt}(x)$, as if ρ_x were known, by substituting the estimated value $\tilde{\rho}_x$ or $\hat{\rho}_x$ in place of ρ_x.

Confidence interval

The 95% confidence interval of $\varphi(x)$ derived from the asymptotic normality of $\tilde{\varphi}_{momt}(x)$ is given by

$$[\tilde{\varphi}_{momt}(x) \pm 1.96\sqrt{V(\rho_x)/k} z^x_{(n-k)} M^{(1)}_n (1 + 1/\rho_x)],$$

where $V(\rho_x) = \rho_x^2 (1 + 2/\rho_x)^{-1}$.

Selection of the sequence k

The number $k = k_n(x)$ plays here the role of the smoothing parameter and varies between 1 and $N_x - 1$, with $N_x = \sum_{i=1}^n \mathbb{1}_{\{x_i \leq x\}}$ being the number of observations (x_i, y_i) such that $x_i \leq x$. The question of selecting the optimal value of $k_n(x)$ is still an open issue and is not addressed yet. Daouia *et al.* (2010) have only suggested an empirical rule implemented in the function `kopt_momt_pick` (option `method = "moment"`) that turns out to give reasonable values of the sequence $k_n(x)$ for estimating the frontier $\varphi(x)$ [see the paragraph **Threshold selection for moment and Pickands frontiers** below for a detailed description of the function `kopt_momt_pick`]. However, as it is common in extreme-value theory, good results require a large sample size N_x of the order of several hundreds. If the resulting pointwise frontier estimates and confidence intervals exhibit severe instabilities, the user should call the function `kopt_momt_pick` by tuning the parameter `wind.coef` in the interval $(0, 1]$ until obtaining more stable curves (default option `wind.coef = 0.1`). See `help(kopt_momt_pick)` for further details.

Practical guidelines

For our illustration purposes using the large dataset `post`, we consider the following three possible scenarios: either ρ_x is known (typically equal to 2 if the assumption of a jump at the frontier is reasonable), or ρ_x is unknown and estimated by the moment estimator $\tilde{\rho}_x$, or ρ_x is unknown independent of x and estimated by the (trimmed) mean of $\tilde{\rho}_x$. First, we select the points at which we want to evaluate the frontier estimator.

```
R> x.post <- seq(post$xinput[100], max(post$xinput), length.out = 100)
```

In the case where the extreme-value index ρ_x is known and equal to 2, we set

```
R> rho <- 2
```

Then, we determine the sequence $k = k_n(x)$ in $\tilde{\varphi}_{momt}(x)$.

```
R> best_kn.1 <- kopt_momt_pick(post$xinput, post$yprod, x.post, rho = rho)
```

When ρ_x is unknown and dependent of x, its estimate $\tilde{\rho}_x$ is computed via the command

```
R> rho_momt <- rho_momt_pick(post$xinput, post$yprod, x.post,
+   method = "moment")
```

To determine the number k in the two-stage estimator $\tilde{\varphi}_{momt}(x)$, we use

```
R> best_kn.2 <- kopt_momt_pick(post$xinput, post$yprod, x.post,
+   rho = rho_momt)
```

Here, for the `post` data, we used the default value `wind.coef = 0.1` in the function `kopt_momt_pick` to avoid numerical instabilities. When employing another large dataset, the user should tune this coefficient until the resulting pointwise frontier estimates and confidence intervals exhibit stable curves (see the function `kopt_momt_pick` for details).

When ρ_x is unknown but independent of x, which is a more realistic setting in practice, a robust estimation strategy is obtained by using the (trimmed) mean over the moment estimates $\tilde{\rho}_x$.

```
R> rho_trimmean <- mean(rho_momt, trim = 0.05)
R> best_kn.3 <- kopt_momt_pick(post$xinput, post$yprod, x.post,
+   rho = rho_trimmean)
```

Finally, we compute the frontier estimates and confidence intervals as follows:

```
R> res.momt.1 = dfs_momt(post$xinput, post$yprod, x.post,
+   rho = rho, k = best_kn.1)
R> res.momt.2 = dfs_momt(post$xinput, post$yprod, x.post,
+   rho = rho_momt, k = best_kn.2)
R> res.momt.3 = dfs_momt(post$xinput, post$yprod, x.post,
+   rho = rho_trimmean, k = best_kn.3)
```

The following code can be used to construct the resulting moment frontier plots graphed in Figure 9.

```
R> my_samp <- post[sample(1:nrow(post), 1000), ]
R> plot(yprod ~ xinput, data = my_samp,
+       xlab = "Quantity of labor",
+   ylab = "Volume of delivered mail")
R> lines(x.post, res.momt.1[,1], lty = 1, col = "cyan")
R> lines(x.post, res.momt.1[,2], lty = 3, col = "magenta")
R> lines(x.post, res.momt.1[,3], lty = 3, col = "magenta")
R> plot(yprod ~ xinput, data = my_samp, xlab = "Quantity of labor",
+   ylab = "Volume of delivered mail")
R> lines(x.post, res.momt.2[,1], lty = 1, col = "cyan")
R> lines(x.post, res.momt.2[,2], lty = 3, col = "magenta")
R> lines(x.post, res.momt.2[,3], lty = 3, col = "magenta")
R> plot(yprod ~ xinput, data = my_samp, xlab = "Quantity of labor",
+   ylab = "Volume of delivered mail")
R> lines(x.post, res.momt.3[,1], lty = 1, col = "cyan")
R> lines(x.post, res.momt.3[,2], lty = 3, col = "magenta")
R> lines(x.post, res.momt.3[,3], lty = 3, col = "magenta")
```

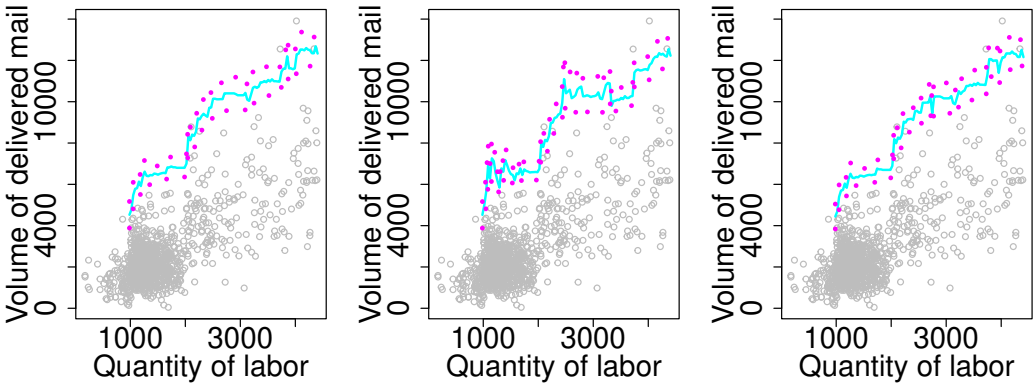

Figure 9: Resulting moment estimator $\tilde{\varphi}_{momt}$ and 95% confidence bands of φ for the 4000 European post offices. From left to right, we have the case $\rho_x = 2$, plugging $\tilde{\rho}_x$ and plugging the mean of $\tilde{\rho}_x$.

Pickands frontier estimator

The function `dfs_pick` computes the Pickands type of estimator and its associated confidence interval introduced by Daouia *et al.* (2010) under the monotonicity constraint.

Built on the ideas of Dekkers and de Haan (1989), Daouia *et al.* (2010) proposed to estimate the frontier point $\varphi(x)$ by

$$\hat{\varphi}_{pick}(x) = \frac{z^x_{(n-k+1)} - z^x_{(n-2k+1)}}{2^{1/\rho_x} - 1} + z^x_{(n-k+1)}$$

from the transformed data $\{z_i^x := y_i \mathbb{1}_{\{x_i \leq x\}}, i = 1, \ldots, n\}$, where $\rho_x > 0$ is the same tail-index as in `dfs_momt`.

If ρ_x is known (typically equal to 2 if the joint density of data is believed to have sudden jumps at the frontier), then one can use the estimator $\hat{\varphi}_{pick}(x)$ in conjunction with the data driven method for selecting the threshold k as described below.

In contrast, if ρ_x is unknown, one could consider using the following two-step estimator: First, estimate ρ_x by the Pickands estimator $\hat{\rho}_x$ implemented in the function `rho_momt_pick` by using the option `method = "pickands"`, or by the moment estimator $\tilde{\rho}_x$ by utilizing the option `method = "moment"` [a detailed description of the function `rho_momt_pick` is provided below in a separate paragraph]. Second, use the estimator $\hat{\varphi}_{pick}(x)$, as if ρ_x were known, by substituting the estimated value $\hat{\rho}_x$ or $\tilde{\rho}_x$ in place of ρ_x.

The pointwise 95% confidence interval of the frontier function obtained from the asymptotic normality of $\hat{\varphi}_{pick}(x)$ is given by

$$[\hat{\varphi}_{pick}(x) \pm 1.96\sqrt{v(\rho_x)/(2k)}(z^x_{(n-k+1)} - z^x_{(n-2k+1)})]$$

where $v(\rho_x) = \rho_x^{-2} 2^{-2/\rho_x}/(2^{-1/\rho_x} - 1)^4$.

Finally, to select the threshold $k = k_n(x)$, one could use the automatic data-driven method of Daouia *et al.* (2010) implemented in the function `kopt_momt_pick` (option `method = "pickands"`) as described below in the last paragraph.

Practical guidelines

For our illustration purposes, we used again the large dataset `post` and considered the following three scenarios: either ρ_x is known (typically equal to 2 if the joint density has sudden jumps at the frontier), ρ_x is unknown and estimated by the Pickands estimator $\hat{\rho}_x$, or ρ_x is unknown independent of x and estimated by the (trimmed) mean of $\hat{\rho}_x$. When ρ_x is known and equal to 2, we set

```
R> rho <- 2
```

Then, we determine the sequence $k = k_n(x)$ in $\hat{\varphi}_{pick}(x)$.

```
R> best_kn.1 <- kopt_momt_pick(post$xinput, post$yprod, x.post,
+   method = "pickands", rho = rho)
```

To estimate ρ_x by $\hat{\rho}_x$, we use the command

```
R> rho_pick <- rho_momt_pick(post$xinput, post$yprod, x.post,
+   method = "pickands")
```

Then, we compute the number $k = k_n(x)$ in the two-stage estimator $\hat{\varphi}_{pick}(x)$ as follows:

```
R> best_kn.2 <- kopt_momt_pick(post$xinput, post$yprod, x.post,
+   method = "pickands", rho = rho_pick)
```

When ρ_x is unknown but independent of x, a robust estimation strategy is by using the (trimmed) mean over the Pickands estimates $\hat{\rho}_x$.

```
R> rho_trimmean <- mean(rho_pick, trim = 0.05)
R> best_kn.3 <- kopt_momt_pick(post$xinput, post$yprod, x.post,
+   rho = rho_trimmean, method = "pickands")
```

Finally, the specifications to calculate the frontier estimates and confidence intervals are given by

```
R> res.pick.1 <- dfs_pick(post$xinput, post$yprod, x.post,
+   rho = rho, k = best_kn.1)
R> res.pick.2 <- dfs_pick(post$xinput, post$yprod, x.post,
+   rho = rho_pick, k = best_kn.2)
R> res.pick.3 <- dfs_pick(post$xinput, post$yprod, x.post,
+   rho = rho_trimmean, k = best_kn.3)
```

The obtained pickands frontiers are graphed in Figure 10. The following code will generate Figure 10.

```
R> plot(yprod ~ xinput, data = my_samp, xlab = "Quantity of labor",
+   ylab = "Volume of delivered mail")
R> lines(x.post, res.pick.1[,1], lty = 1, col = "cyan")
R> lines(x.post, res.pick.1[,2], lty = 3, col = "magenta")
R> lines(x.post, res.pick.1[,3], lty = 3, col = "magenta")
R> plot(yprod ~ xinput, data = my_samp, xlab = "Quantity of labor",
+   ylab = "Volume of delivered mail")
R> lines(x.post, res.pick.2[,1], lty = 1, col = "cyan")
R> lines(x.post, res.pick.2[,2], lty = 3, col = "magenta")
R> lines(x.post, res.pick.2[,3], lty = 3, col = "magenta")
R> plot(yprod ~ xinput, data = my_samp, xlab = "Quantity of labor",
+   ylab = "Volume of delivered mail")
R> lines(x.post, res.pick.3[,1], lty = 1, col = "cyan")
R> lines(x.post, res.pick.3[,2], lty = 3, col = "magenta")
R> lines(x.post, res.pick.3[,3], lty = 3, col = "magenta")
```

Moment and Pickands estimates of the tail-index ρ_x

The function `rho_momt_pick` computes the moment and Pickands estimates of the extreme-value index ρ_x involved in the frontier estimators $\tilde{\varphi}_{momt}(x)$ [see `dfs_momt`] and $\hat{\varphi}_{pick}(x)$ [see `dfs_pick`].

For the case where `method = "moment"`, the estimator of ρ_x defined as

$$\tilde{\rho}_x = -\left(M_n^{(1)} + 1 - \frac{1}{2}\left[1 - (M_n^{(1)})^2/M_n^{(2)}\right]^{-1}\right)^{-1}$$

is based on the moments $M_n^{(j)} = (1/k)\sum_{i=0}^{k-1}\left(\log z^x_{(n-i)} - \log z^x_{(n-k)}\right)^j$ for $j = 1, 2$, with $z^x_{(1)} \leq \ldots \leq z^x_{(n)}$ being the ascending order statistics which correspond to the transformed

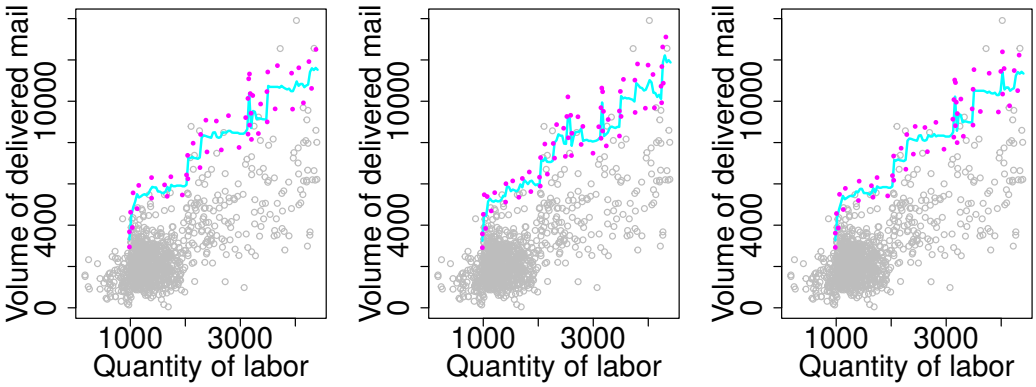

Figure 10: Resulting Pickands estimator $\hat{\varphi}_{pick}$ and 95% confidence interval of φ for the 4000 European post offices. From left to right, we have the case $\rho_x = 2$, plugging $\hat{\rho}_x$, and plugging the mean of $\hat{\rho}_x$.

sample $\{z_i^x := y_i \mathbb{1}_{\{x_i \leq x\}}, i = 1, \ldots, n\}$. See the note in `help(rho_momt_pick)` for further details.

In the case where `method = "pickands"`, the estimator of ρ_x is given by

$$\hat{\rho}_x = -\log 2 / \log\{(z^x_{(n-k+1)} - z^x_{(n-2k+1)})/(z^x_{(n-2k+1)} - z^x_{(n-4k+1)})\}.$$

To select the threshold $k = k_n(x)$ in $\tilde{\rho}_x$ and $\hat{\rho}_x$, Daouia *et al.* (2010) have suggested to use the following data driven method for each x: They first select a grid of values for $k_n(x)$. For the Pickands estimator $\hat{\rho}_x$, they choose $k_n(x) = [N_x/4] - k + 1$, where k is an integer varying between 1 and the integer part $[N_x/4]$ of $N_x/4$, with $N_x = \sum_{i=1}^n \mathbb{1}_{\{x_i \leq x\}}$. For the moment estimator $\tilde{\rho}_x$, they choose $k_n(x) = N_x - k$, where k is an integer varying between 1 and $N_x - 1$. Then, they evaluate the estimator $\hat{\rho}_x(k)$ (respectively, $\tilde{\rho}_x(k)$) and select the k where the variation of the results is the smallest. They achieve this by computing the standard deviation of $\hat{\rho}_x(k)$ (respectively, $\tilde{\rho}_x(k)$) over a "window" of $\max([\sqrt{N_x/4}], 3)$ (respectively, $\max([\sqrt{N_x - 1}], 3)$) successive values of k. The value of k where this standard deviation is minimal defines the value of $k_n(x)$.

The user can also appreciably improve the estimation of ρ_x and $\varphi(x)$ itself by tuning the choice of the lower limit (default option `lrho = 1`) and upper limit (default option `urho = Inf`).

Threshold selection for moment and Pickands frontiers

The function `kopt_momt_pick` is an implementation of an experimental method by Daouia *et al.* (2010) for the automated threshold selection (choice of $k = k_n(x)$) for the moment frontier estimator $\tilde{\varphi}_{momt}(x)$ [see `dfs_momt`] in the case where `method = "moment"` and for the Pickands frontier estimator $\hat{\varphi}_{pick}(x)$ [see `dfs_pick`] in the case where `method = "pickands"`. The idea is to select first (for each x) a grid of values for the number $k_n(x)$ given by $k = 1, \ldots, [\sqrt{N_x}]$, where $[\sqrt{N_x}]$ stands for the integer part of $\sqrt{N_x}$ with $N_x = \sum_{i=1}^n \mathbb{1}_{\{x_i \leq x\}}$, and then select the k where the variation of the results is the smallest. To achieve this here, Daouia

et al. (2010) compute the standard deviations of $\tilde{\varphi}_{momt}(x)$ [option `method = "moment"`] or $\hat{\varphi}_{pick}(x)$ [option `method = "pickands"`] over a "window" of size $\max(3, [\texttt{wind.coef} \times \sqrt{N_x}/2])$, where the coefficient `wind.coef` should be selected in the interval $(0, 1]$ in such a way to avoid numerical instabilities. The default option `wind.coef = 0.1` corresponds to having a window large enough to cover around 10% of the possible values of k in the selected range of values for $k_n(x)$. The value of k where the standard deviation is minimal defines the desired number $k_n(x)$. See the note in `help(kopt_momt_pick)` for further details.

Probability-weighted moment frontier estimator

The function `dfs_pwm` computes the regularized frontier estimator introduced by Daouia *et al.* (2012). It is based on the unregularized probability-weighted moment (PWM) estimator

$$\hat{\varphi}_m(x) = \varphi_{fdh}(x) - \int_0^{\varphi_{fdh}(x)} \hat{F}^m(y|x)dy$$

where the trimming order $m \geq 1$ is an integer such that $m = m_n \to \infty$ as $n \to \infty$, and $\hat{F}(y|x) = \sum_{i=1}^n \mathbb{1}_{(x_i \leq x, y_i \leq y)} / \sum_{i=1}^n \mathbb{1}_{(x_i \leq x)}$. The implemented estimator of $\varphi(x)$ is then defined as

$$\tilde{\varphi}_{pwm}(x) = \hat{\varphi}_m(x) + \Gamma\left(1 + 1/\bar{\rho}_x\right) \left(1/m\, \hat{\ell}_x\right)^{1/\bar{\rho}_x}$$

where

$$\bar{\rho}_x = \log(a) \left\{ \log\left(\frac{\hat{\varphi}_m(x) - \hat{\varphi}_{am}(x)}{\hat{\varphi}_{am}(x) - \hat{\varphi}_{a^2m}(x)} \right) \right\}^{-1}, \quad \hat{\ell}_x = \frac{1}{m} \left[\frac{\Gamma(1 + 1/\bar{\rho}_x)(1 - a^{-1/\bar{\rho}_x})}{\hat{\varphi}_m(x) - \hat{\varphi}_{am}(x)} \right]^{\bar{\rho}_x},$$

with $a \geq 2$ being a fixed integer and $\bar{\rho}_x$ estimates the same tail-index $\rho_x = \beta_x + 2$ as in `dfs_momt` and `dfs_pick`. If the true value of ρ_x is known, we set $\bar{\rho}_x = \rho_x$ in the expressions above. In contrast, if ρ_x is unknown, its estimate $\bar{\rho}_x$ can be obtained separately in an optimal way by calling the function `rho_pwm` described below in the last paragraph. In both cases, we use the frontier estimator $\tilde{\varphi}_{pwm}(x)$ as if $\bar{\rho}_x$ were known by plugging in its value. As pointed out by Daouia *et al.* (2012), it is most efficient to conduct tail-index estimation and frontier estimation separately. Then, knowing the value $\bar{\rho}_x$, it remains to fix the two smoothing parameters a and m in order to calculate the frontier estimator $\tilde{\varphi}_{pwm}(x)$. A practical choice of these parameters that Daouia *et al.* (2012) have employed is the simple rule of thumb $a = 2$ [default option in the 5th argument of the function] and $m = \text{coefm} \times N_x^{1/3}$, where $N_x = \sum_{i=1}^n \mathbb{1}_{\{x_i \leq x\}}$ and the integer `coefm` is to be tuned by the user in the 4th argument of the function. Daouia *et al.* (2012) have suggested in their numerical illustrations to use, for instance, the value `coefm = 1`. An automatic data-driven rule for choosing the optimal tuning parameter `coefm` is implemented in the function `mopt_pwm` described below.

Confidence interval

The pointwise 95% confidence interval of $\varphi(x)$ derived from the asymptotic normality of $\tilde{\varphi}_{pwm}(x)$ is given by $[\tilde{\varphi}_{pwm}(x) \pm 1.96\, \hat{\sigma}(m, x)/\sqrt{n}]$ where

$$\hat{\sigma}^2(m, x) = \frac{2m^2}{\hat{F}_X(x)} \int_0^{\varphi_{fdh}(x)} \int_0^{\varphi_{fdh}(x)} \hat{F}^m(y|x) \hat{F}^{m-1}(u|x)(1 - \hat{F}(u|x))\, \mathbb{1}_{(y \leq u)}\, dy\, du,$$

with $\hat{F}_X(x) = (1/n)\sum_{i=1}^{n} \mathbb{1}_{(x_i \leq x)}$. Note that the standard deviation $\sigma(m,x)/\sqrt{n}$ of the bias-corrected estimator $\tilde{\varphi}_{pwm}(x)$ is adjusted by a bootstrap estimator in the numerical illustrations of Daouia *et al.* (2012), whereas the exact estimate $\hat{\sigma}(m,x)/\sqrt{n}$ is utilized in our implemented function.

Practical guidelines

By way of example, we used as before the large dataset `post` and considered the following three possible scenarios: either ρ_x is known (typically equal to 2 if the assumption of a jump at the frontier is valid), or ρ_x is unknown and estimated by the PWM estimator $\bar{\rho}_x$, or ρ_x is unknown independent of x and estimated by the (trimmed) mean of $\bar{\rho}_x$. When $\rho_x = 2$,

```
R> rho <- 2
```

we get `coefm` in $\tilde{\varphi}_{pwm}(x)$ and the frontier estimate $\tilde{\varphi}_{pwm}(x)$ itself via the commands

```
R> best_cm.1 <- mopt_pwm(post$xinput, post$yprod, x.post,
+   a = 2, rho = rho, wind.coef = 0.1)
R> res.pwm.1 <- dfs_pwm(post$xinput, post$yprod, x.post,
+   coefm = best_cm.1, a = 2, rho = rho)
```

To obtain the estimate $\bar{\rho}_x$ and its (trimmed) mean, we use the following specifications

```
R> rho_pwm <- rho_pwm(post$xinput, post$yprod, x.post, a = 2,
+   lrho = 1, urho = Inf)
R> rho_pwm_trim <- mean(rho_pwm, trim = 0.05)
```

The corresponding smoothing parameters `coefm` and frontier estimates are computed as follows:

```
R> best_cm.2 <- mopt_pwm(post$xinput, post$yprod, x.post,
+   a = 2, rho = rho_pwm)
R> best_cm.3 <- mopt_pwm(post$xinput, post$yprod, x.post,
+   a = 2, rho = rho_pwm_trim)
R> res.pwm.2 <- dfs_pwm(post$xinput, post$yprod, x.post,
+   coefm = best_cm.2, rho = rho_pwm)
R> res.pwm.3 <- dfs_pwm(post$xinput, post$yprod, x.post,
+   coefm = best_cm.3, rho = rho_pwm_trim)
```

The following code can be used to construct the resulting PWM frontier plots graphed in Figure 11.

```
R> plot(yprod ~ xinput, data = my_samp, xlab = "Quantity of labor",
+   ylab = "Volume of delivered mail")
R> lines(x.post, res.pwm.1[,1], lty = 1, col = "cyan")
R> lines(x.post, res.pwm.1[,2], lty = 3, col = "magenta")
R> lines(x.post, res.pwm.1[,3], lty = 3, col = "magenta")
R> plot(yprod ~ xinput, data = my_samp, xlab = "Quantity of labor",
```

```
+   ylab = "Volume of delivered mail")
R> lines(x.post, res.pwm.2[,1], lty = 1, col = "cyan")
R> lines(x.post, res.pwm.2[,2], lty = 3, col = "magenta")
R> lines(x.post, res.pwm.2[,3], lty = 3, col = "magenta")
R> plot(yprod ~ xinput, data = my_samp, xlab = "Quantity of labor",
+   ylab = "Volume of delivered mail")
R> lines(x.post, res.pwm.3[,1], lty = 1, col = "cyan")
R> lines(x.post, res.pwm.3[,2], lty = 3, col = "magenta")
R> lines(x.post, res.pwm.3[,3], lty = 3, col = "magenta")
```

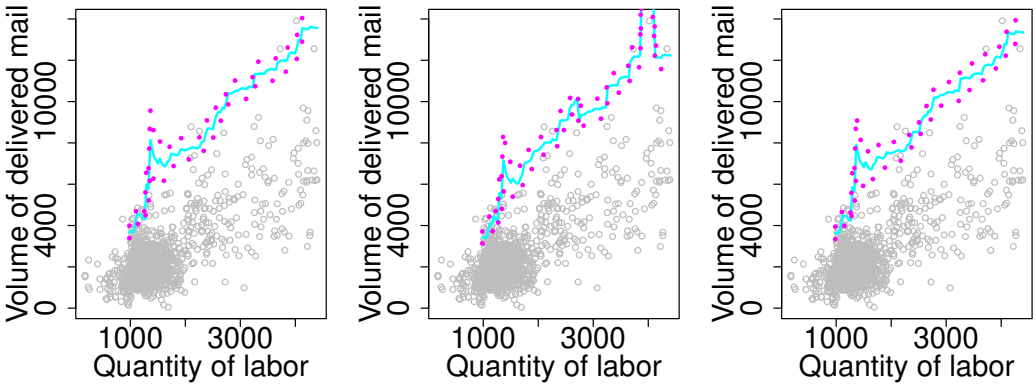

Figure 11: Resulting regularized PWM estimator $\tilde{\varphi}_{pwm}$ and 95% confidence interval of φ for the 4000 European post offices. From left to right, we have the case $\rho_x = 2$, plugging $\bar{\rho}_x$ and plugging the mean of $\bar{\rho}_x$.

Threshold selection for the PWM frontier estimator

The function `mopt_pwm` implements an automated selection of the parameter `coefm` involved in the probability-weighted moment (PWM) estimator $\tilde{\varphi}_{pwm}(x)$ [see `dfs_pwm`]. It is an adaptation of the experimental method `kopt_momt_pick` by Daouia *et al.* (2010). The idea is to select first (for each x) a grid of values for the parameter `coefm` given by $c = 1, \ldots, \min(10, [\sqrt{N_x}])$, where $N_x = \sum_{i=1}^{n} \mathbb{1}_{\{x_i \leq x\}}$, and then select the c where the variation of the results is the smallest. To achieve this, we compute the standard deviations of $\tilde{\varphi}_{pwm}(x)$ over a "window" of size $wind.coef \times \min(10, [\sqrt{N_x}])$, where the coefficient `wind.coef` should be selected in the interval $(0, 1]$ in such a way to avoid numerical instabilities. The default option `wind.coef = 0.1` corresponds to having a window large enough to cover around 10% of the possible values of c in the selected range of values for `coefm`. The value of c where the standard deviation is minimal defines the desired `coefm`.

PWM estimate of the tail-index ρ_x

The function `rho_pwm` computes the probability-weighted moment (PWM) estimator $\bar{\rho}_x$ utilized in the frontier estimate $\tilde{\varphi}_{pwm}(x)$ [see `dfs_pwm`]. This estimator depends on the smoothing parameters a and m. A simple selection rule of thumb that Daouia *et al.* (2012) have em-

ployed is $a = 2$ [default option in the 4th argument of the function] and $m = \text{coefm} \times N_x^{1/3}$, where $N_x = \sum_{i=1}^{n} \mathbb{1}_{\{x_i \leq x\}}$ and the integer `coefm` is to be tuned by the user. To choose this parameter in an optimal way for each x, we adapt the automated threshold selection method of Daouia *et al.* (2010) as follows: We first evaluate the estimator $\bar{\rho}_x$ over a grid of values of `coefm` given by $c = 1, \ldots, 150$. Then, we select the c where the variation of the results is the smallest. This is achieved by computing the standard deviation of the estimates $\bar{\rho}_x$ over a "window" of $\max([\sqrt{150}], 3)$ successive values of c. The value of c where this standard deviation is minimal defines the value of `coefm`.

The user can also appreciably improve the estimation of ρ_x and $\varphi(x)$ itself by tuning the choice of the lower limit (default option `lrho = 1`) and upper limit (default option `urho = Inf`).

4. Numerical illustrations

Comparisons among most of the selected estimation methods described above have been undertaken by Daouia *et al.* (2016) and more recently by Noh (2014) via simulation experiments. To encourage others to explore these methods and easily compare the quality of any new proposal with the competitive existing methods, we provide some guidelines that facilitate comparision based on Monte-Carlo simulations in a similar way to the devices of Daouia *et al.* (2016) and Noh (2014).

4.1. Comparison criteria

After estimating the true frontier function $\varphi(x)$ from N independent samples of size n, Daouia *et al.* (2016) and Noh (2014) considered the empirical mean integrated squared error (MISE), the empirical integrated squared bias (IBIAS2) and the empirical integrated variance (IVAR), which are given by

$$\begin{aligned}
\text{MISE} &= \frac{1}{N}\sum_{j=1}^{N} \text{ISE}(\hat{\varphi}^{(j)}) := \frac{1}{N}\sum_{j=1}^{N}\left[\frac{1}{I}\sum_{i=0}^{I}\left(\hat{\varphi}^{(j)}(z_i) - \varphi(z_i)\right)^2\right] \\
&= \frac{1}{I}\sum_{i=0}^{I}\left(\varphi(z_i) - \bar{\hat{\varphi}}(z_i)\right)^2 + \frac{1}{I}\sum_{i=0}^{I}\left[\frac{1}{N}\sum_{j=1}^{N}(\hat{\varphi}^{(j)}(z_i) - \bar{\hat{\varphi}}(z_i))^2\right] \\
&\equiv \text{IBIAS2} + \text{IVAR},
\end{aligned}$$

where $\{z_i,\ i = 0, \ldots, I\}$ is an equispaced grid having width $1/I$ over $[a, b]$ (the true support of the input variable), with $I = 1000$, $\hat{\varphi}^{(j)}(\cdot)$ is the estimated frontier function from the j-th data sample and $\bar{\hat{\varphi}}(z_i) = N^{-1}\sum_{j=1}^{N}\hat{\varphi}^{(j)}(z_i)$. Although the definition of these comparison criteria is quite straightforward, some caution should be taken when calculating them. The reason is that the estimation of $\hat{\varphi}^{(j)}(z_i)$ is possible only when z_i lies between the minimum and maximum of the inputs of the jth sample $x_1^{(j)}, \ldots, x_n^{(j)}$. In our package, when storing the estimates $\hat{\varphi}^{(j)}(z_i)$, $i = 1, \ldots, n$, we let the value $\hat{\varphi}^{(j)}(z_i)$ assigned to zero for distinction when the estimation is not possible. The function `evaluation` automatically computes the comparison criteria using only nonzero estimates at every grid point z_i. The first argument

of this function is the matrix where the estimation results are stored, the second argument is the evaluation grid vector, and the third argument is the vector of values of the true frontier function at the grid points.

4.2. Some Monte Carlo evidence

By way of example, to evaluate finite-sample performance of the empirical LFDH and DEA frontier estimators in comparison with the polynomial, spline and kernel smoothed estimators, we have undertaken some simulation experiments following Daouia *et al.* (2016)'s study. The experiments all employ the model $y_i = \varphi(x_i)\, v_i$, where x_i is uniform on $[0, 1]$ and v_i, independent of x_i, is Beta(β, β) with values of $\beta = 0.5, 1$ and 3 [corresponding, respectively, to a joint density of the (x_i, y_i)'s increasing toward infinity, having a jump or decreasing to zero as it approaches the support boundary]. Tables 3 and 4 report the obtained Monte Carlo estimates when $\varphi(x) = x^{1/2}$ and $\varphi(x) = \exp(-5 + 10x)/(1 + \exp(-5 + 10x))$, respectively. All the experiments were performed over $N = 5000$ independent samples of size $n = 25$, 50, 100 and 200.

The code which generates the results in Tables 3 and 4 is given in the supplementary file. Note that the computational burden here is demanding, so be forewarned. Note also that only $N = 200$ replications were considered in Daouia *et al.* (2016).

		dea_est (type="dea")	cub_spline_est (type="mc") (all.dea=T)	quad_spline_est (type="mc") (all.dea=T)	kern_smooth (type="mc") (technique="pr") ("cv")	kern_smooth (type="mc") (technique="noh") ("bic")	poly_est ("BIC")
$\beta = 0.5$							
$n = 25$	IBIAS2	0.002655	0.001525	0.001759	0.018617	0.001208	0.011507
	IVAR	0.001942	0.001955	0.002045	0.007450	0.001935	0.031622
	IMSE	0.004597	0.003480	0.003803	0.026067	0.003143	0.043130
$n = 50$	IBIAS2	0.000793	0.000412	0.000481	0.009313	0.000347	0.001429
	IVAR	0.000615	0.000594	0.000621	0.003511	0.000584	0.007217
	IMSE	0.001408	0.001006	0.001102	0.012824	0.000931	0.008646
$n = 100$	IBIAS2	0.000226	0.000105	0.000127	0.005078	0.000152	0.000350
	IVAR	0.000183	0.000168	0.000174	0.001336	0.000168	0.001007
	MISE	0.000409	0.000274	0.000300	0.006414	0.000320	0.001358
$n = 200$	IBIAS2	0.000061	0.000025	0.000032	0.003399	0.000105	0.000198
	IVAR	0.000048	0.000044	0.000045	0.000539	0.000049	0.000167
	MISE	0.000109	0.000069	0.000077	0.003938	0.000154	0.000365
$\beta = 1$					(N=4799)		
$n = 25$	IBIAS2	0.008049	0.005598	0.006092	0.014150	0.005202	0.024311
	IVAR	0.002856	0.003188	0.003282	0.006447	0.003160	0.027117
	MISE	0.010905	0.008786	0.009374	0.020597	0.008362	0.051428
$n = 50$	IBIAS2	0.003401	0.002223	0.002447	0.007114	0.002065	0.007184
	IVAR	0.001288	0.001390	0.001438	0.003040	0.001400	0.010049
	MISE	0.004688	0.003613	0.003885	0.010154	0.003465	0.017233
$n = 100$	IBIAS2	0.001305	0.000784	0.000878	0.003904	0.000747	0.001928
	IVAR	0.000497	0.000538	0.000537	0.001320	0.000539	0.003196
	MISE	0.001802	0.001322	0.001415	0.005224	0.001286	0.005124
					(N=4999)		
$n = 200$	IBIAS2	0.000525	0.000298	0.000342	0.002540	0.000310	0.000589
	IVAR	0.000201	0.000219	0.000212	0.000551	0.000216	0.001054
	MISE	0.000727	0.000517	0.000555	0.003091	0.000526	0.001643
$\beta = 3$					(N=4773)		
$n = 25$	IBIAS2	0.029439	0.024860	0.025751	0.021526	0.024553	0.050245
	IVAR	0.002940	0.003485	0.003555	0.004882	0.003441	0.014190
	MISE	0.032379	0.028345	0.029306	0.026407	0.027994	0.064435
$n = 50$	IBIAS2	0.018980	0.015737	0.016307	0.014489	0.015749	0.030942
	IVAR	0.001857	0.002204	0.002258	0.002895	0.002196	0.007812
	MISE	0.020837	0.017941	0.018565	0.017384	0.017944	0.038755
$n = 100$	IBIAS2	0.012697	0.010435	0.010824	0.010368	0.010586	0.019784
	IVAR	0.001177	0.001411	0.001447	0.001708	0.001366	0.004460
	MISE	0.013874	0.011846	0.012271	0.012076	0.011952	0.024244
					(N=4995)		
$n = 200$	IBIAS2	0.008182	0.006616	0.006885	0.007150	0.006820	0.012588
	IVAR	0.000735	0.000901	0.000903	0.000976	0.000843	0.002722
	MISE	0.008917	0.007518	0.007788	0.008126	0.007673	0.015310
					(N=4682)		

Table 3: Monte-Carlo comparison when the true frontier is monotone and concave ($\varphi(x) = \sqrt{x}$), with $N = 5000$ replications. Colors code : 1st rank, 2nd rank. When $N < 5000$, this means that `solve.QP` was unable to find a solution (Hayfield and Racine 2008, suggest then to adjust constraints and restart).

		dea_est (type="lfdh")	cub_spline_est (type="m") (all.dea=F)	quad_spline_est (type="m") (all.dea=F)	kern_smooth (type="m") (technique="pr") ("cv")	kern_smooth (type="m") (technique="noh") ("bic")	poly_est ("BIC")
$\beta = 0.5$							
$n = 25$	IBIAS2	0.007032	0.001601	0.001580	0.002627	0.002071	0.015539
	IVAR	0.005284	0.004923	0.005130	0.006121	0.004485	0.030231
	MISE	0.012316	0.006524	0.006710	0.008748	0.006557	0.045770
$n = 50$	IBIAS2	0.002492	0.000200	0.000294	0.000570	0.000554	0.003047
	IVAR	0.002073	0.000988	0.001595	0.002283	0.001264	0.009148
	MISE	0.004565	0.001188	0.001889	0.002854	0.001818	0.012195
$n = 100$	IBIAS2	0.000916	0.000023	0.000122	0.000099	0.000126	0.000489
	IVAR	0.000829	0.000180	0.000549	0.000818	0.000431	0.002151
	MISE	0.001745	0.000203	0.000672	0.000918	0.000557	0.002640
					(N=4997)		
$n = 200$	IBIAS2	0.000347	0.000009	0.000044	0.000014	0.000017	0.000080
	IVAR	0.000335	0.000039	0.000171	0.000149	0.000102	0.000409
	MISE	0.000682	0.000048	0.000215	0.000163	0.000119	0.000489
$\beta = 1$					(N=4769)		
$n = 25$	IBIAS2	0.016099	0.006079	0.006077	0.005694	0.007217	0.025964
	IVAR	0.005824	0.006517	0.006624	0.006149	0.005723	0.023157
	MISE	0.021923	0.012597	0.012700	0.011843	0.012941	0.049121
$n = 50$	IBIAS2	0.007761	0.001880	0.001845	0.002213	0.003455	0.009294
	IVAR	0.003058	0.002341	0.002930	0.002528	0.002561	0.010293
	MISE	0.010819	0.004221	0.004775	0.004741	0.006016	0.019587
$n = 100$	IBIAS2	0.003700	0.000541	0.000525	0.000869	0.001590	0.002977
	IVAR	0.001511	0.000767	0.001274	0.001098	0.001252	0.003713
	MISE	0.005211	0.001308	0.001799	0.001967	0.002842	0.006690
					(N=4998)		
$n = 200$	IBIAS2	0.001757	0.000151	0.000176	0.000332	0.000708	0.000898
	IVAR	0.000745	0.000246	0.000609	0.000400	0.000602	0.001303
	MISE	0.002502	0.000397	0.000785	0.000732	0.001310	0.002202
$\beta = 3$					(N=4706)		
$n = 25$	IBIAS2	0.038773	0.024179	0.024325	0.021889	0.025572	0.044459
	IVAR	0.004420	0.005280	0.005490	0.004586	0.004817	0.011995
	MISE	0.043193	0.029459	0.029815	0.026475	0.030389	0.056454
$n = 50$	IBIAS2	0.026249	0.014575	0.014621	0.015485	0.018127	0.027638
	IVAR	0.002996	0.003205	0.003465	0.002635	0.002996	0.006719
	MISE	0.029245	0.017779	0.018087	0.018121	0.021123	0.034357
$n = 100$	IBIAS2	0.018164	0.009415	0.009543	0.011111	0.013310	0.017469
	IVAR	0.001981	0.001902	0.002250	0.001677	0.002002	0.003854
	MISE	0.020144	0.011317	0.011794	0.012788	0.015312	0.021323
					(N=4996)		
$n = 200$	IBIAS2	0.012546	0.006264	0.006477	0.007611	0.009773	0.011220
	IVAR	0.001328	0.001196	0.001488	0.001081	0.001351	0.002338
	MISE	0.013874	0.007460	0.007966	0.008692	0.011123	0.013558
					(N=4631)		

Table 4: Monte-Carlo comparison when the true frontier is only monotone ($\varphi(x) = \frac{\exp(-5+10\times x)}{(1+\exp(-5+10\times x))}$), with $N = 5000$ replications. Colors code : 1st rank, 2nd rank. When $N < 5000$, this means that `solve.QP` was unable to find a solution (Hayfield and Racine 2008, suggest then to adjust constraints and restart).

Acknowledgments

The first author acknowledges financial support by the Toulouse School of Economics Individual Research Fund (IRF/Daouia-20125) and by the Seventh Framework Programme of the European Union (IEF/273584/EMBAF-project). The research of the third author was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2014R1A1A2059875).

References

Akaike H (1973). "Information Theory and an Extension of the Maximum Likelihood Principle." In BN Petrov, F Csaki (eds.), *2nd International Symposium on Information Theory*, pp. 267–281.

Aragon Y, Daouia A, Thomas-Agnan C (2005). "Nonparametric Frontier Estimation: A Conditional Quantile-Based Approach." *Econometric Theory*, **21**, 358–389.

Bogetoft P, Otto L (2011). *Benchmarking with DEA, SFA and R*. Springer-Verlag.

Cazals C, Florens JP, Simar L (2002). "Nonparametric Frontier Estimation: A Robust Approach." *Journal of Econometrics*, **106**, 1–25.

Chernozhukov V (2005). "Extremal Quantile Regression." *The Annals of Statistics*, **33**, 806–839.

Christensen LR, Greene WH (1976). "Economies of Scale in U.S. Electric Power Generation." *Journal of Political Economy*, **84**, 655–676.

Daouia A, Florens JP, Simar L (2008). "Functional Convergence of Quantile-Type Frontiers with Application to Parametric Approximations." *Journal of Statistical Planning and Inference*, **138**, 708–725.

Daouia A, Florens JP, Simar L (2010). "Frontier Estimation and Extreme Value Theory." *Bernoulli*, **16**, 1039–1063.

Daouia A, Florens JP, Simar L (2012). "Regularization of Nonparametric Frontier Estimators." *Journal of Econometrics*, **168**, 285–299.

Daouia A, Gardes L, Girard S (2013). "On Kernel Smoothing for Extremal Quantile Regression." *Bernoulli*, **19**, 2557–2589.

Daouia A, Gijbels I (2011). "Robustness and Inference in Nonparametric Partial Frontier Modeling." *Journal of Econometrics*, **161**, 174–165.

Daouia A, Girard S, Guillou A (2014). "A Γ-Moment Approach to Monotonic Boundary Estimation." *Journal of Econometrics*, **78**, 727–740.

Daouia A, Laurent T, Noh H (2017). ***npbr**: Nonparametric Boundary Regression*. R package version 1.5, URL http://CRAN.R-project.org/package=npbr.

Daouia A, Noh H, Park BU (2016). "Data Envelope Fitting with Constrained Polynomial Splines." *Journal of the Royal Statistical Society B*, **78**, 3–30.

Daouia A, Ruiz-Gazen A (2006). "Robust Nonparametric Frontier Estimators: Qualitative Robustness and Influence Function." *Statistica Sinica*, **16**, 1233–1253.

Daouia A, Simar L (2005). "Robust Nonparametric Estimators of Monotone Boundaries." *Journal of Multivariate Analysis*, **96**, 311–331.

de Haan L, Resnick S (1994). "Estimating the Home Range." *Journal of Applied Probability*, **31**, 700–720.

Dekkers ALM, de Haan L (1989). "On the Estimation of Extreme-Value Index and Large Quantiles Estimation." *The Annals of Statistics*, **17**, 1795–1832.

Dekkers ALM, Einmahl JHJ, de Haan L (1989). "A Moment Estimator for the Index of an Extreme-Value Distribution." *The Annals of Statistics*, **17**, 1833–1855.

Deprins D, Simar L, Tulkens H (1984). "Measuring Labor Efficiency in Post Offices." In M Marchand, P Pestieau, H Tulkens (eds.), *The Performance of Public Enterprises: Concepts and Measurements*, pp. 243–267.

Farrell MJ (1957). "The Measurement of Productive Efficiency." *Journal of the Royal Statistical Society A*, **120**, 253–281.

Gijbels I, Mammen E, Park BU, Simar L (1999). "On Estimation of Monotone and Concave Frontier Functions." *Journal of American Statistical Association*, **94**, 220–228.

Gijbels I, Peng L (2000). "Estimation of a Support Curve via Order Statistics." *Extremes*, **3**, 251–277.

Girard S, Jacob P (2003). "Extreme Values and Haar Series Estimates of Point Process Boundaries." *Scandinavian Journal of Statistics*, **30**, 369–384.

Girard S, Jacob P (2004). "Extreme Values and Kernel Estimates of Point Processes Boundaries." *ESAIM: Probability and Statistics*, **8**, 150–168.

Hall P, Huang H (2001). "Nonparametric Kernel Regression Subject to Monotonicity Constraints." *The Annals of Statistics*, **29**, 624–647.

Hall P, Nussbaum M, Stern SE (1997). "On the Estimation of a Support Curve of Indeterminate Sharpness." *Journal of Multivariate Analysis*, **62**, 204–232.

Hall P, Park BU (2002). "New Methods for Bias Correction at Endpoints and Boundaries." *The Annals of Statistics*, **30**, 1460–1479.

Hall P, Park BU (2004). "Bandwidth Choice for Local Polynomial Estimation of Smooth Boundaries." *Journal of Multivariate Analysis*, **91**, 240–261.

Hall P, Park BU, Stern SE (1998). "On Polynomial Estimators of Frontiers and Boundaries." *Journal of Multivariate Analysis*, **66**, 71–98.

Härdle W, Park BU, Tsybakov AB (1995). "Estimation of Non-Sharp Support Boundaries." *Journal of Multivariate Analysis*, **43**, 205–218.

Hayfield T, Racine JS (2008). "Nonparametric Econometrics: the **np** Package." *Journal of Statistical Software*, **27**, 1–32.

Hendricks W, Koenker K (1992). "Hierarchical Spline Models for Conditional Quantiles and the Demand for Electricity." *Journal of the American Statistical Association*, **87**, 58–68.

Jacob P, Suquet P (1995). "Estimating the Edge of a Poisson Process by Orthogonal Series." *Journal of Statistical Planning and Inference*, **46**, 215–234.

Jeong SO, Simar L (2006). "Linearly Interpolated FDH Efficiency Score for Nonconvex Frontiers." *Journal of Multivariate Analysis*, **97**, 2141–2161.

Jirak M, Meister A, Reiss M (2014). "Optimal Adaptive Estimation in Nonparametric Regression with One-Sided Errors." *The Annals of Statistics*, **42**, 1970–2002.

Knight K (2001). "Limiting Distributions of Linear Programming Estimators." *Extremes*, **4**, 87–103.

Koenker R (2017). ***quantreg**: Quantile Regression*. R package version 5.33, URL http://CRAN.R-project.org/package=quantreg.

Korostelev A, Tsybakov AB (1993). *Minimax Theory of Image Reconstruction*. Volume 82 of Lecture Notes in Statistics, Springer-Verlag, New-York.

Li Q, Lin J, Racine JS (2013). "Optimal Bandwidth Selection for Nonparametric Conditional Distribution and Quantile Functions." *Journal of Business and Economic Statistics*, **31**, 57–65.

Lu M, Zhang Y, Huang J (2007). "Estimation of the Mean Function with Panel Count Data Using Monotone Polynomial Splines." *Biometrika*, **94**, 705–718.

Makhorin A (2017). *GNU Linear Programming Kit*. URL http://www.gnu.org/software/glpk/glpk.html.

Mouchart M, Simar L (2002). "Efficiency Analysis of Air Controllers: First Insights." *Technical report*, Institut de Statistique, Université Catholique de Louvain, Belgium.

Ng P, Maechler M (2007). "A Fast and Efficient Implementation of Qualitatively Constrained Quantile Smoothing Splines." *Statistical Modelling*, **7**, 315–328.

Noh H (2014). "Frontier Estimation Using Kernel Smoothing with Data Transformation." *Journal of the Korean Statistical Society*, **43**, 503–512.

Park BU (2001). "On Nonparametric Estimation of Data Edges." *Journal of the Korean Statistical Society*, **30**, 265–280.

Parmeter C, Racine JS (2013). "Smooth Constrained Frontier Analysis." In X Chen, N Swanson (eds.), *Recent Advances and Future Directions in Causality, Prediction, and Specification Analysis: Essays in Honor of Halbert L. White, Jr.*, pp. 463–488. Springer-Verlag.

Pya N, Wood S (2014). "Shape Constrained Additive Models." *Statistics and Computing*, pp. 1–17. doi:10.1007/s11222-013-9448-7.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org.

Schumaker LL (2007). *Spline Functions: Basic Theory*. Cambridge University Press.

Schwartz G (1978). "Estimating the Dimension of a Model." *The Annals of Statistics*, **6**, 461–464.

Simar L, Wilson PW (2008). "Statistical Inference in Nonparametric Frontier Models: Recent Developments and Perspectives." In H Fried, C Lovell, S Schmidt (eds.), *The Measurement of Productive Efficiency*, pp. 421–521. Oxford University Press.

Theussl S, Hornik K (2017). ***Rglpk**: R/GNU Linear Programming Kit Interface*. R package version 0.6-3, URL https://CRAN.R-project.org/package=Rglpk.

Turlach BA, Weingessel A (2013). ***quadprog**: Functions to Solve Quadratic Programming Problems*. R package version 1.5-5, S original by Berwin A. Turlach, R port by Andreas Weingessel, URL http://CRAN.R-project.org/package=quadprog.

Affiliation:

Thibault Laurent
Toulouse School of Economics (CNRS)
Université Toulouse 1 Capitole
21, allée de Brienne
31042 Toulouse, FRANCE
E-mail: Thibault.Laurent@univ-tlse1.fr

Journal of Statistical Software https://www.jstatsoft.org/
published by the Foundation for Open Access Statistics https://www.foastat.org/
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd