Journal of Statistical Software

October 2015, Volume 67, Issue 3. doi: 10.18637/jss.v067.i03

Threshold Value Estimation Using Adaptive Two-Stage Plans in R

Shawn Mankad
Cornell University

George Michailidis
University of Michigan

Moulinath Banerjee
University of Michigan

Abstract

This paper introduces the R package **twostageTE** for estimation of an inverse regression function at a given point when one can sample an explanatory covariate at different values and measure the corresponding responses. The package implements a number of nonparametric methods for budget constrained threshold value estimation. Specifically, it contains methods for classical one-stage designs and also adaptive two-stage designs, which have been shown to yield more efficient and accurate results. A major advantage of the methods in package **twostageTE** is that threshold value estimation is performed without penalization or kernel smoothing, and hence, avoids the well-known problems of choosing the corresponding tuning parameter (regularization, bandwidth). The user can easily perform a two-stage analysis with **twostageTE** by (i) identifying the second stage sampling region from an initial sample, and (ii) computing various types of confidence intervals to ensure a robust analysis. The package **twostageTE** is illustrated through simulated examples.

Keywords: threshold estimation, two-stage estimation, R.

1. Introduction

The problem of estimating an inverse regression function has a long history in statistics, and has impacted a diverse set of areas including toxicology (Rosenberger and Haines 2002), statistical calibration studies (Osborne 1991), and engineering (Tang, Banerjee, and Michailidis 2011).

The canonical formulation posits

$$Y = f(X) + \epsilon, \tag{1}$$

where f is a function establishing the relationship between the design variable X and the response Y, and ϵ is an error term with zero mean and finite variance. It is further assumed

that f is monotone. The main problem is to estimate $d_0 = f^{-1}(\theta_0)$ for some θ_0 in the range of f.

In practice, there are situations where the total budget of measurements to be obtained is fixed a priori. Given the limited budget of points that can be sampled and lack of a priori knowledge about the location of d_0, the adaptive two-stage methods of Tang *et al.* (2011) and Tang, Banerjee, Michailidis, and Mankad (2015) have been shown to provide more accurate estimates compared to one-stage procedures.

In particular with two-stage procedures, given the monotonicity assumption on f, isotonic regression is used at stage one to obtain an initial estimate. Subsequently, the remaining portion of the available points is sampled from a neighborhood around this initial estimate to yield a new estimate of d_0. The more intensive sampling around the initial estimate produces a more accurate estimate than the one that would have been obtained by utilizing the entire budget in a uniform fashion. Tang *et al.* (2011) and Tang *et al.* (2015) show two-stage methods accelerate the convergence rate of one-stage procedures and achieve the parametric $n^{1/2}$ rate under certain regularity conditions.

This paper introduces the R (R Core Team 2015) package **twostageTE** (Mankad, Michailidis, and Banerjee 2015), which implements a number of nonparametric methods for budget constrained threshold value estimation using one- and two-stage designs and is available from the Comprehensive R Archive Network (CRAN) at `http://CRAN.R-project.org/package=twostageTE`. **twostageTE** is created to facilitate such analysis by providing functions that perform classical one-stage analysis, identify the second stage sampling region from an initial sample, and compute various types of one- and two-stage confidence intervals to ensure a robust analysis.

A notable feature about the inference methods in package **twostageTE** is that they do not utilize penalization or kernel smoothing and, thus, are easier to implement. Moreover, the case for smoothing-based methods is dubious, as extensive investigations (see Section 5 and Tang *et al.* 2015, for further details) have found that kernel based two-stage methods do not perform well for more "ill-behaved" functions, nor do they yield significant gains over simpler methods, as in Tang *et al.* (2011), for well-behaved functions.

The remainder of this article is organized as follows: In the next section, we review estimation methods and the underlying nonparametric procedures based on isotonic regression. Section 3 discusses the structure of **twostageTE**, and Section 4 illustrates the methodology on simulated examples. The article concludes with a brief discussion (Section 5). Appendix A contains a brief overview of the main theoretical results for one- and two-stage procedures.

2. A general adaptive two-stage estimation framework

2.1. Background on isotonic regression

We provide a brief description of the main nonparametric procedure underlying both one- and two-stage approaches, named the isotonic regression procedure. Specifically, given n fixed or random design points $\{X_i\}_{i=1}^n$ in $[a, b]$ and the corresponding responses $\{Y_i\}_{i=1}^n$, the isotonic

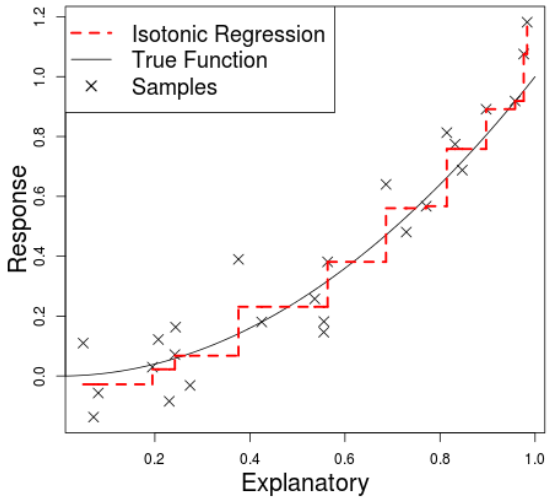

Figure 1: Isotonic regression estimates for a quadratic function $f(x) = x^2$ over the $[0, 1]$ interval.

regression estimate of $f(\cdot)$ is given by

$$f_I(x) = f_1^\star 1\{x \in [a, X_1]\} + \sum_{i=1}^{n-1} f_i^\star 1\{x \in [X_i, X_{i+1})\} + f_n^\star 1\{x \in [X_n, b]\} \tag{2}$$

where $\{f_i^\star\}_{i=1}^n = \underset{f_1 \le f_2 \le \ldots \le f_n}{\operatorname{argmin}} \sum_{i=1}^n (Y_i - f_i)^2$. This minimizer exists uniquely, has a nice geometric characterization as the slope of the greatest convex minorant of a stochastic process and is readily computable using the pool adjacent violators algorithm (PAVA, see De Leeuw, Hornik, and Mair 2009; Robertson, Wright, and Dykstra 1988, for further discussion). Then, for a prespecified value $\theta_0 \in (f(a), f(b))$, the one-stage isotonic regression estimator of d_0 is defined by

$$d_I = f_I^{-1}(\theta_0) = \inf\{x \in [a, b] : f_I(x) \ge \theta_0\}, \tag{3}$$

where $\inf\{\emptyset\} = b$.

To illustrate, Figure 1 shows isotonic regression (IR) estimates, where the design space is the $[0, 1]$ interval for a quadratic function $f(x) = x^2$; the random error follows a $N(0, \sigma^2)$ distribution with $\sigma = 0.1$. We can see the IR procedure provides an accurate estimate with relatively few samples. The estimated curve is computed by calling the function `pava` in package **twostageTE** on the sampled data. Details on this and other functions are given in Section 3.

An alternative nonparametric estimate of $f(\cdot)$ is given by smoothing the isotonic regression estimate. However, as noted in the previous section, smoothed two-stage procedures do not bring significant gains. Thus, in this article and in package **twostageTE**, we discuss inference procedures that do not utilize penalization or kernel smoothing.

2.2. Two-stage frameworks

As noted in Section 1, adaptive two-stage procedures can lead to accelerated convergence rates and hence to sharper confidence intervals for d_0. The main steps of such an adaptive two-stage fully nonparametric procedure are outlined next:

Type	Function
(i) External (Wrapper)	`stageOneAnalysis`, `stageTwoAnalysis`
(ii) Internal	`likelihoodConfidenceInterval`, `linearBootstrapConfidenceInterval_stageTwo`, `waldConfidenceInterval_ir_stageOne`, `waldConfidenceInterval_ir_stageTwo`, `threshold_estimate_ir`, `threshold_estimate_locLinear`
(iii) Internal Helper	`estimateDeriv`, `estimateSigmaSq`, `pava`

Table 1: Organizational hierarchy for the R package **twostageTE**.

1. Allocate n_1 samples to the first stage and n_2 samples to the second stage, where $n_1+n_2 = n$, the total budget.

2. Generate the first stage data $\{(X_{1,i}, Y_{1,i})\}_{i=1}^{n_1}$ with a design density g_1 on $[a, b]$. Then, compute a first stage monotone nonparametric estimator $\hat{f}_1$ of f and obtain the corresponding first stage estimator $\hat{d}_1 = \hat{f}_1^{-1}(\theta_0)$ of d_0 for a prespecified value θ_0.

3. Specify the second stage sampling interval $[L_1, U_1]$ based on a high probability confidence interval from the first stage data.

4. Obtain the second stage data $\{(X_{2,i}, Y_{2,i})\}_{i=1}^{n_2}$ with a design density g_2 on $[L_1, U_1]$. Employ these data and a nonparametric procedure (which could be different from the one used previously) to compute a monotone second stage estimator $\hat{f}_2$ and, as in the first stage, the corresponding $\hat{d}_2$.

5. Construct confidence intervals for d_0 using the (asymptotic) distribution of $\hat{d}_2$.

There are a number of different approaches one could pursue in the first and second stages. For example, Tang *et al.* (2011) utilize a local linear approximation at the second stage to accelerate the convergence rate from $n^{1/3}$ to $n^{1/2}$ under certain regularity conditions. However, as shown in Tang *et al.* (2015), the local linear approximation and other nonparametric procedures that require an estimate of the derivative $f'(d_0)$ struggle when faced with underlying regression functions that exhibit strong nonlinearity at d_0. To overcome these difficulties, one can pursue likelihood ratio type confidence intervals, which are preferred due to their simplicity and robustness. A number of these different procedures are implemented in package **twostageTE**, which is discussed next.

3. Package description and functional structure

The **twostageTE** package contains three main components: (i) wrapper functions that a user would need to call in the course of a typical analysis; (ii) internal functions that compute first and second stage point estimates and confidence intervals; and (iii) auxiliary functions that support functions in (ii). This organizational structure is summarized in Table 1 and Figure 2.

In the following subsections, we highlight various aspects of the package and present simple examples based on simulated data.

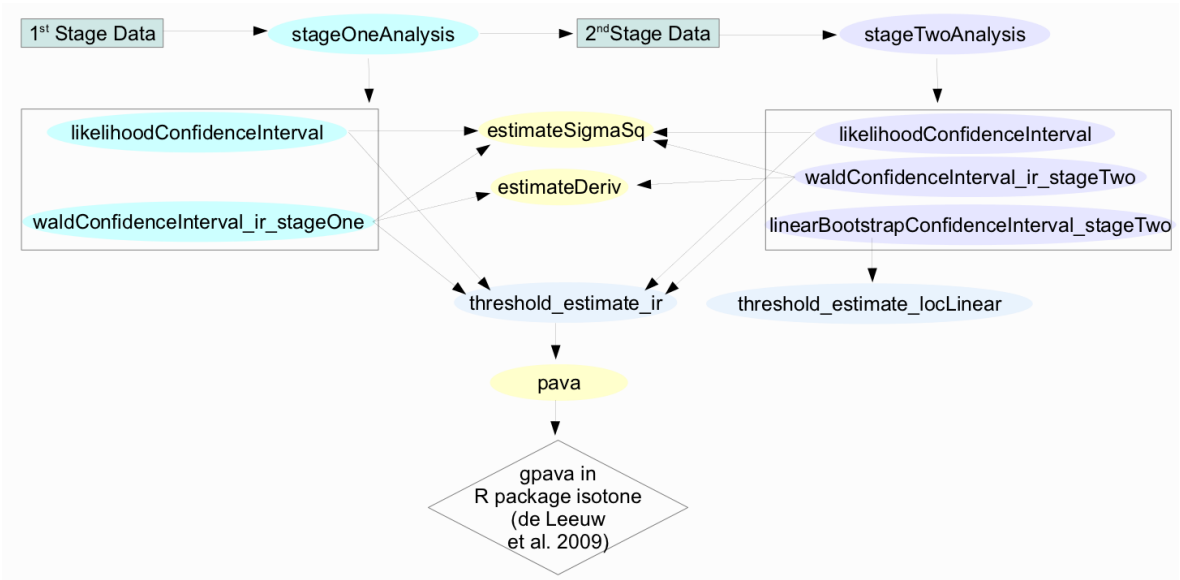

Figure 2: Flowchart of the R package **twostageTE**.

3.1. Creating a 'twostageTE' object

The wrapper function `StageOneAnalysis` requires inputs of the samples and their corresponding responses, the threshold of interest $f(d_0)$, a confidence level, and a string that specifies what type of confidence interval to compute (Wald or likelihood ratio). Thus, this function is appropriate after obtaining a first stage sample, or if one were to proceed in a classical fashion by using the entire sampling budget uniformly. `StageTwoAnalysis` additionally requires as input the object returned from `StageOneAnalysis`. Both functions return a '`twostageTE`' object.

We illustrate below use of these functions for the following procedures.

1. One-stage procedure based on isotonic regression with Wald confidence intervals (IR).
2. One-stage procedure based on isotonic regression with likelihood ratio (LR) confidence intervals.
3. Two-stage procedure based on isotonic regression for both stages and using Wald confidence intervals both for selecting (L_1, U_1) and constructing the final confidence interval (IR + IR).
4. Two-stage procedure similar to (IR + IR), but employing LR confidence intervals in both stages (LR + LR).
5. Two-stage procedure from Tang *et al.* (2011) that uses isotonic regression followed by a local linear approximation and bootstrapping for constructing confidence intervals for d_0 (IR + locLinear).

For the first stage, we generate synthetic data shown in Figure 1 with the following R code.

```
R> X <- runif(25, 0, 1)
R> Y <- X^2 + rnorm(n = length(X), sd = 0.1)
```

One stage: IR

The one-stage IR-Wald procedure uses isotonic regression to estimate d_0 and quantiles of the standard Chernoff distribution to form confidence intervals. Additional discussion can be found in Appendix A.1.

To obtain a 99% Wald-type confidence interval using IR for the threshold $\theta = 0.25$ ($d_0 = f^{-1}(\theta) = 0.5$), we call `stageOneAnalysis`.

```
R> library("twostageTE")
R> oneStage_IR <- stageOneAnalysis(X, Y, 0.25, type = "IR-wald", 0.99)
R> class(oneStage_IR)

[1] "twostageTE"

R> oneStage_IR

Call:
stageOneAnalysis(X, response = Y, threshold = 0.25, type = "IR-wald",
    level = 0.99)

99.0% Confidence Interval
n   Lower   d0_hat   Upper
25   0.119   0.507   0.923
```

One stage: LR

The one-stage LR procedure uses isotonic regression to estimate d_0, and inverts the likelihood ratio test to form confidence intervals. Previous investigations (Tang *et al.* 2015) have demonstrated the superiority of likelihood ratio type confidence intervals. Additional discussion can be found in Appendix A.1.

To obtain a likelihood ratio-based 99% confidence interval for $\theta = 0.25$, we call

```
R> oneStage_LR <- stageOneAnalysis(X, Y, 0.25, type = "IR-likelihood", 0.99)
R> oneStage_LR

Call:
stageOneAnalysis(X, response = Y, threshold = 0.25, type = "IR-likelihood",
    level = 0.99)

99.0% Confidence Interval
n   Lower   d0_hat   Upper
25   0.209   0.507   0.696
```

3.2. Implementing the second stage

Two-stage analysis: LR + LR

The two-stage LR procedure is similar to its single stage counterpart. Both procedures

use isotonic regression to estimate d_0, and invert the likelihood ratio test statistic to form confidence intervals. Additional discussion can be found in Appendix A.2.

We simulate the second stage by generating 75 second stage samples using the 99% LR-type confidence interval computed above from `oneStage_LR` as the second stage design space.

```
R> X2 <- seq(oneStage_LR$L1, oneStage_LR$U1, length.out = 75)
R> Y2 <- X2^2 + rnorm(n = length(X2), sd = 0.1)
```

To obtain a two-stage 95% confidence interval based on the likelihood ratio for both stages, we call `stageTwoAnalysis`.

```
R> twoStage_LR_LR <- stageTwoAnalysis(oneStage_LR, X2, Y2,
+    type = "IR-likelihood", 0.95)
R> twoStage_LR_LR

Call:
stageTwoAnalysis(oneStage_LR, explanatory = X2, response = Y2,
    type = "IR-likelihood", level = 0.95)

95.0% Confidence Interval
n1   n2   Lower   d0_hat   Upper
25   75   0.433   0.544   0.604
```

Two-stage analysis: IR + LocLinear

The main idea for the procedure in Tang *et al.* (2011) is to utilize a local linear approximation in the vicinity of the first stage estimate, and to bootstrap this local approximation to obtain confidence intervals. Additional details are provided in Appendix A.2.

To perform the local linear procedure, we first simulate the second stage by repeatedly sampling from the 99% IR Wald-type confidence interval end points computed above in `oneStage_IR`.

```
R> X2 <- c(rep(oneStage_IR$L1, 37), rep(oneStage_IR$U1, 38))
R> Y2 <- X2^2 + rnorm(n = length(X2), sd = 0.1)
```

Then we obtain a 95% confidence interval based on the two-stage local linear approximation by calling

```
R> twoStage_IR_linear <- stageTwoAnalysis(oneStage_IR, explanatory = X2,
+    response = Y2, type = "locLinear", level = 0.95)
R> twoStage_IR_linear

Call:
stageTwoAnalysis(oneStage_LR, explanatory = X2, response = Y2,
    type = "logLinear", level = 0.95)

95.0% Confidence Interval
n1   n2   Lower   d0_hat   Upper
25   75   0.323   0.350   0.377
```

The local linear approximation in this particular example is downward biased over the interval $[0, 1]$, because the true function is quadratic. As the budget and/or noise-level is increased, the local linear approximation improves and the bias decreases. When the true function is well approximated locally with a linear function, then this procedure should work well. Thus, we recommend the local linear approximation is used only when the user is comfortable making the linearity assumption. Otherwise, the nonparametric techniques should be preferred.

Two-stage analysis: IR + IR

As with one-stage IR, the two-stage IR-Wald procedure uses isotonic regression to estimate d_0 from second stage samples, and quantiles of the standard Chernoff distribution to form confidence intervals. Additional discussion can be found in Appendix A.2.

We again simulate the second stage by generating 75 second stage samples using the 99% IR Wald-type confidence interval computed above in `oneStage_IR` as the second stage design space.

```
R> X2 <- seq(oneStage_IR$L1, oneStage_IR$U1, length.out = 75)
R> Y2 <- X2^2 + rnorm(n = length(X2), sd = 0.1)
```

To obtain a 95% IR + IR confidence interval for $\theta = 0.25$ ($d_0 = 0.5$), we call `stageTwoAnalysis`.

```
R> twoStage_IR_IR <- stageTwoAnalysis(oneStage_IR, X2, Y2, type = "IR-wald",
+    0.95)
R> twoStage_IR_IR

Call:
stageTwoAnalysis(oneStage_IR, explanatory = X2, response = Y2,
    type = "IR-wald", level = 0.95)

95.0% Confidence Interval
n1   n2   Lower   d0_hat   Upper
25   75   0.127   0.532   0.923
```

Combining first and second stage data

When calling `stageTwoAnalysis`, there is an optional Boolean input called `combineData` that, if set to `TRUE`, will utilize all available data (from both stages) that are contained in the second stage design space for estimation of auxiliary parameters and inversion of the likelihood ratio statistic. Combining data can help overcome difficulties of working with modest budgets. By default, however, `stageTwoAnalysis` does not combine data from both stages, since this practice is outside the strict purview of most theoretical results. In our toy example, we find that combining data from both stages results in a slightly more precise confidence interval.

```
R> stageTwoAnalysis(oneStage_IR, X2, Y2, type = "IR-wald", 0.95,
+    combineData = TRUE)

Call:
stageTwoAnalysis(oneStage_IR, explanatory = X2, response = Y2,
```

```
    type = "IR-wald", level = 0.95, combineData = TRUE)

95.0% Confidence Interval
n1   n2   Lower   d0_hat   Upper
25   98   0.123   0.499    0.876
```

Internal and helper functions

The internal and helper functions that support `StageOneAnalysis` and `StageTwoAnalysis` are fairly straightforward. The function `pava` in particular is critical to the package. It computes IR estimates with the pool adjacent violators algorithm (PAVA), and depends on the function `gpava` in the package **isotone** (De Leeuw *et al.* 2009).

PAVA is an iterative algorithm for solving monotonic regression problems. It starts with measurement pairs (X_i, Y_i), ordered with respect to the responses Y_i. The initialization of PAVA sets the estimated regression function $\hat{f}(X_i) = Y_i$. When violations of monotonicity are discovered, e.g., $\hat{f}(X_{i+1}) < \hat{f}(X_i)$, then $\hat{f}(X_{i+1})$ and $\hat{f}(X_i)$ are replaced by their average. This iterative process stops when the estimated regression function $\hat{f}(X_i)$ is non-decreasing. The function `gpava` by default utilizes the sample mean at repeated X_i values. An extensive description of the PAVA algorithm and its implementation are provided in De Leeuw *et al.* (2009, Section 3.1) and references therein.

For the estimation of σ^2, `estimateSigmaSq` computes the nonparametric estimator proposed by Gasser, Sroka, and Jennen-Steinmetz (1986). This estimator takes ordered triples of design points X_{i-1}, X_i, X_{i+1}, joins the two outer observations by a straight line and then computes the difference between this straight line and the middle observation. Specifically,

$$\tilde{\epsilon}_i = \frac{X_{i+1} - X_i}{X_{i+1} - X_{i-1}} Y_{i-1} + \frac{X_i - X_{i-1}}{X_{i+1} - X_{i-1}} Y_{i+1} - Y_i \tag{4}$$

$$= a_i Y_{i-1} + b_i Y_{i+1} - Y_i. \tag{5}$$

The estimate of σ^2 is then

$$\hat{\sigma}^2 = \frac{1}{n-2} \sum_{i=1}^{n-1} c_i^2 \tilde{\epsilon}_i^2, \tag{6}$$

where $c_i = 1/(a_i + b_i + 1)$ for $i = 2, \ldots, n-1$. As in `gpava`, `estimateSigmaSq` utilizes the sample mean when given replicate X_i values.

To estimate $f'(d_0)$, a local quadratic regression procedure is implemented in `estimateDeriv`. Specifically, let $K(\cdot)$ denote the Epanechnikov kernel function and $h > 0$ the bandwidth, so that $K_h(\cdot) = (1/h)K(\cdot/h)$.

The bandwidth h is chosen based on Equation 3.20 of Fan and Gijbels (1996, p. 67). In particular,

$$\hat{h}_{opt}(\hat{d}_0) = C_{1,2}(K) \left[\frac{\hat{\sigma}^2}{(\hat{f}^{(3)}(\hat{d}_0))^2} \right]^{1/7} n^{-1/7}, \tag{7}$$

where $C_{1,2}(K) = 2.275$ and $\hat{f}^{(3)}(\hat{d}_0)$ is the third order derivative of f at $\hat{d}_0$. First, a fifth order polynomial function is fit to the data, where $f(x) = \sum_{j=0}^{5} \alpha_j x^j$. Then, an estimate of the third order derivative at $\hat{d}_0$ is obtained by $f^{(3)}(\hat{d}_0) = 6\alpha_3 + 24\alpha_4 \hat{d}_0 + 60\alpha_5 \hat{d}_0^2$.

Once the bandwidth is set, the estimate of $f'(d_0)$ is given by $\hat{\beta}_1$ from

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^3}{\operatorname{argmin}} \sum_{i=1}^{n} \left[Y_i - \sum_{j=0}^{2} \beta_j (X_i - \hat{d}_0)^j \right]^2 K_{h_{opt}}(X_i - \hat{d}_0), \tag{8}$$

where $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2)$.

3.3. Using a 'twostageTE' object

Upon building a 'twostageTE' object, one can explore the model characteristics by using the `summary` and `plot` functions.

The `summary` *function*

To summarize a 'twostageTE' object, the summary function returns the confidence interval, and estimates of d_0, σ^2 and $f'(d_0)$.

The following code runs `summary` on one- and two-stage results.

```
R> summary(oneStage_IR)


Call:
stageOneAnalysis(X, response = Y, threshold = 0.25, type = "IR-wald",
    level = 0.99)

99.0% Confidence Interval
n   Lower   d0_hat   Upper
25   0.119   0.507   0.923

Auxiliary Estimates
f'(d_0)   sigma^2
0.749     0.007


R> summary(twoStage_IR_IR)


Call:
stageTwoAnalysis(oneStage_IR, explanatory = X2, response = Y2,
    type = "IR-wald", level = 0.95)

95.0% Confidence Interval
n1   n2   Lower   d0_hat   Upper
25   75   0.127   0.532   0.923

Auxiliary Estimates
f'(d_0)   sigma^2
0.746     0.011
```

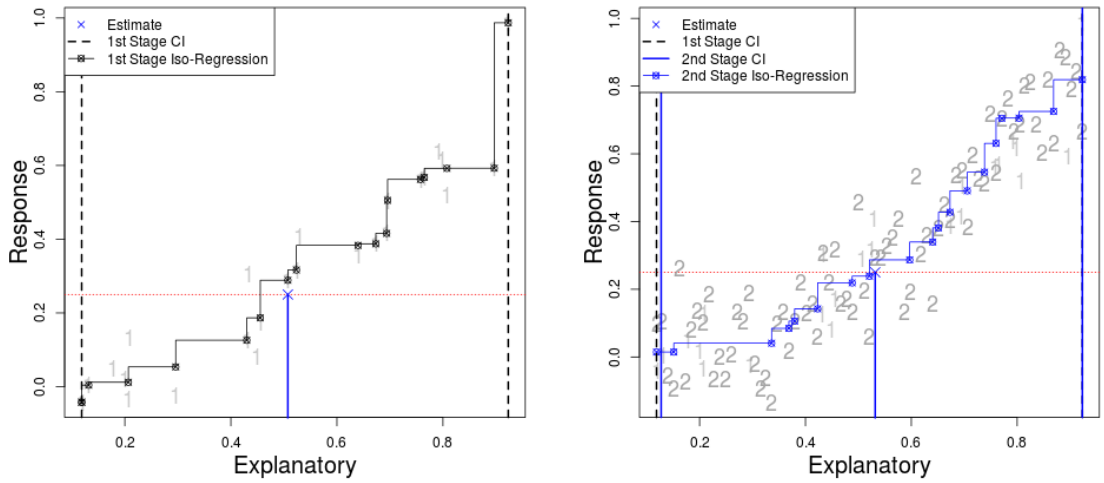

Figure 3: Visualizing samples and results. The dashed vertical lines denote the first stage confidence interval (second stage design space), full vertical lines denote the second stage confidence interval, and "X" marks the threshold and most current point estimate.

The `plot` *method*

The `plot` method for '`twostageTE` objects overrides the default `plot` function in R by plotting the full data (from both stages, if available), first and second stage confidence intervals, the final point estimate, and the isotonic regression estimate using the most current stage data.

The output of calling `plot(oneStage_IR)` and `plot(twoStage_IR_IR)` is shown in the left and right panels, respectively, of Figure 3.

4. A synthetic example based on the isotonic sine function

In this section we test the different methods on a synthetic example that allows us to compare and validate the methods' abilities to estimate the inverse regression function at a given point when the underlying function is "ill-behaved." In particular, we investigate the isotonic sine function, which is of special interest as it features severe departures from linearity that can adversely affect estimation of auxiliary parameters and the local linear-based approximation. As noted above, Tang *et al.* (2015) show that the likelihood ratio based procedures are most robust with respect to this highly nonlinear function.

We first create a sampling function (`sampleData`) that is repeatedly invoked to simulate a true adaptive two-stage sampling and compare it against classical one-stage procedures. In the following code, we also define the total budget, true value of d_0, as well as the first and second stage sample allocations. The true, underlying function is set to be an isotonic sine shown in the left panel of Figure 4. Specifically, the function is

$$f(x) = (1/40)\sin(6\pi x) + 1/4 + (1/2)x + (1/4)x^2 x. \tag{9}$$

The key aspect of the isotonic sine is that its derivative is not well-behaved, as shown in Figure 4, which degrades the performance of less flexible techniques. Thus, this function is useful for illustration, since it provides key insights about different techniques when the true

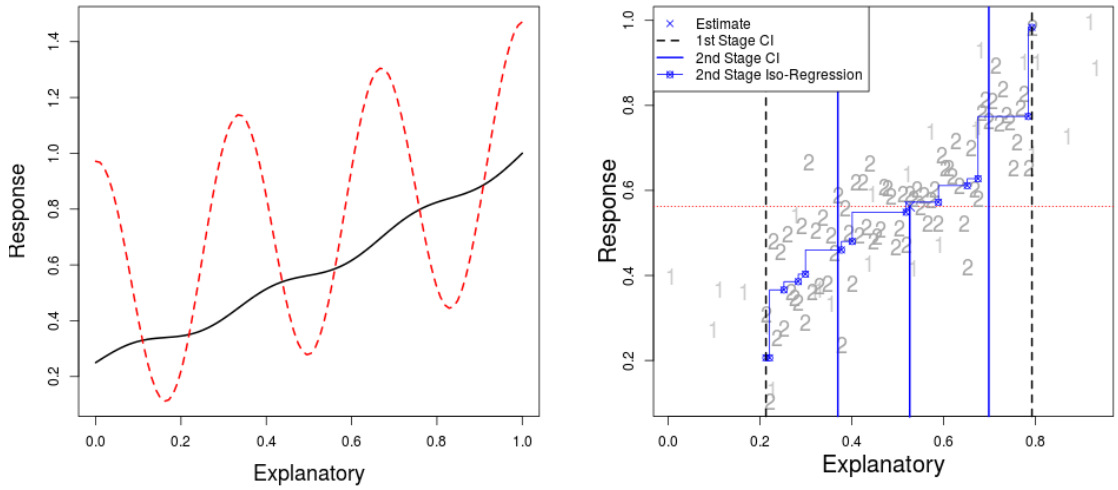

Figure 4: As shown in the left panel the isotonic sine function (solid) is challenging due to its derivative (dashed). The right hand panel shows the output of `plot` on the two-stage likelihood ratio approach.

underlying function is far from linear. In practice, deviation from linearity can be visually assessed after the first stage or a priori information may be useful. The target point is again set to be $d_0 = 0.5$, so $f(d_0) \approx 0.56$ is known.

```
R> sampleData <- function(n, lower, upper, equal = FALSE)
+  {
+    if (equal) x <- seq(lower, upper, length.out = n)
+    else x <- runif(n, lower, upper)
+    y <- (1/40) * sin(6 * pi * x) + 1/4 + x/2 + (1/4) * x^2 +
+    rnorm(n = length(x), sd = 0.1)
+    return(list(X = x, Y = y))
+  }
R> Budget <- 100
R> d0 <- 0.5
R> threshold <- (1/40) * sin(6 * pi * d0) + 1/4 + d0/2 + (1/4) * d0^2
```

Next we simulate two-stage plans using the first and second stage allocations set to the (asymptotically) optimal ratio of 25% of points in the first stage (Tang *et al.* 2015).

The following generates first stage data.

```
R> n1 <- floor(Budget * 0.25)
R> n2 <- Budget - n1
R> samp <- sampleData(n1, lower = 0, upper = 1)
R> X <- samp$X
R> Y <- samp$Y
```

The following simulates two-stage analysis setting `type = "IR-Wald"`.

Procedure	n_1	n_2	Confidence interval	$\hat{d}_0$
`twoStageLR`	25	75	$[0.370, 0.699]^*$	0.526
`twoStageWald`	25	75	$[0.146, 0.867]$	0.507
`twoStageLinear`	25	75	$[0.274, 0.443]$	0.359
`oneStageLR`	100	0	$[0.385, 0.626]$	0.469
`oneStageWald`	100	0	$[0.051, 0.887]$	0.469

Table 2: Summary of output for the isotonic sine simulation. `twoStageLR` features the shortest confidence interval that contains $d_0 = 0.5$.

```
R> stageOne_IR <- stageOneAnalysis(X, Y, threshold, type = "IR-wald", 0.99)
R> samp2 <- sampleData(n2, lower = stageOne_IR$L1, upper = stageOne_IR$U1,
+    equal = TRUE)
R> X2 <- samp2$X
R> Y2 <- samp2$Y
R> twoStageIR <- stageTwoAnalysis(stageOne_IR, X2, Y2, type = "IR-wald",
+    0.95)
```

The following simulates two-stage analysis setting `type = "IR-likelihood"`.

```
R> stageOne_LR <- stageOneAnalysis(X, Y, threshold,
+    type = "IR-likelihood", 0.99)
R> samp2 <- sampleData(n2, lower = stageOne_LR$L1, upper = stageOne_LR$U1,
+    equal = TRUE)
R> X2 <- samp2$X
R> Y2 <- samp2$Y
R> twoStageLR <- stageTwoAnalysis(stageOne_LR, X2, Y2,
+    type = "IR-likelihood", 0.95)
```

The following simulates two-stage analysis setting `type = "locLinear"`.

```
R> X2 <- c(rep(stageOne_IR$L1, floor(n2/2)),
+    rep(stageOne_IR$U1, floor(n2/2)))
R> Y2 <- (1/40) * sin(6 * pi * X2) + 1/4 + d0/2 + (1/4) * X2^2 +
+    rnorm(n = length(X2), sd = 0.1)
R> twoStageLinear <- stageTwoAnalysis(stageOne_IR, X2, Y2,
+    type = "locLinear", level = 0.95)
```

Lastly, the following code computes the classical one-stage estimates using the entire budget uniformly.

```
R> samp <- sampleData(Budget, lower = 0, upper = 1)
R> X <- samp$X
R> Y <- samp$Y
R> oneStageIR <- stageOneAnalysis(X, Y, threshold, type = "IR-wald", 0.95)
R> oneStageLR <- stageOneAnalysis(X, Y, threshold, type = "IR-likelihood",
+    0.95)
```

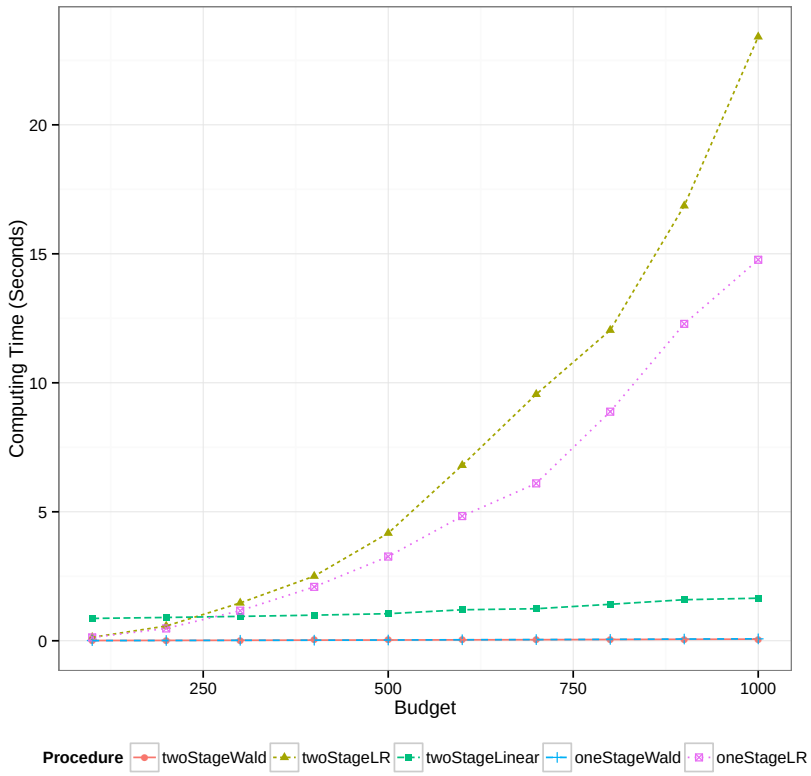

Figure 5: Average computing times with different budgets for the isotonic sine example. Likelihood ratio based procedures are most robust, but require inverting the likelihood ratio, which adds computing cost.

The results are summarized in Table 2. In this particular instance, all procedures except `twoStageLinear` cover the true value. As expected, the two-stage procedures feature narrower confidence intervals, with `twoStageLR` providing the best results. The samples and results from `twoStage_LR` are shown in the right panel of Figure 4.

These results are consistent with Tang *et al.* (2015), who found in an extensive simulation that the two-stage likelihood ratio type confidence intervals perform well even with "ill-behaved" functions like the isotonic sine. The local linear approximation and IR-Wald procedures that require an estimate of the derivative $f'(d_0)$ struggle when faced with the isotonic sine function, since it exhibits strong nonlinearity at $d_0 = 0.5$.

Altogether, our and previous numerical studies have shown that likelihood ratio based procedures are robust to small budgets and "ill-behaved" functions. However, inversion of the likelihood ratio statistic can be computationally costly. The Wald and local linear procedures can be performed faster, though their efficacy is dubious with smaller budgets and with "ill-behaved" functions, due to auxiliary parameters estimation.

5. Discussion

The simulation results above support the conclusions in Tang *et al.* (2015) that the local linear approximation is useful with an approximately linear underlying function near the threshold.

In practice, linearity can be assessed visually after obtaining the first stage results. We recommend the local linear approximation is used only when the user is comfortable making the linearity assumption. Otherwise, the non-parametric techniques should be preferred.

The R package **twostageTE** implements a broad range of nonparametric methods for threshold value estimation. A notable and novel feature of **twostageTE** is that it accommodates and facilitates two-stage designs, which have been shown to yield more efficient and accurate results without penalization or kernel smoothing, and hence, avoids the well-known problems of choosing a penalization or smoothing parameter. The user can easily perform a full two-stage analysis by (i) identifying the second stage sampling region from an initial sample, and (ii) computing a battery of confidence intervals based on local linear approximation and other non-parametric methods to ensure a robust analysis.

References

Banerjee M (2000). *Likelihood Ratio Inference in Regular and Non-Regular Problems*. Ph.D. thesis, University of Washington.

Banerjee M (2009). "Inference in Exponential Family Regression Models under Certain Shape Constraints." In *Advances in Multivariate Statistical Methods, Statistical Science and Interdisciplinary Research*, volume 4, pp. 249–272. World Scientific.

Banerjee M, Wellner J (2001). "Likelihood Ratio Tests for Monotone Functions." *The Annals of Statistics*, **29**(6), 1699–1731. doi:10.1214/aos/1015345959.

De Leeuw J, Hornik K, Mair P (2009). "Isotone Optimization in R: Pool-Adjacent-Violators Algorithm (PAVA) and Active Set Methods." *Journal of Statistical Software*, **32**(5), 1–24. doi:10.18637/jss.v032.i05.

Fan J, Gijbels I (1996). *Local Polynomial Modelling and Its Applications*, volume 66 of *Monographs on Statistics and Applied Probability*. Chapman & Hall, London.

Gasser T, Sroka L, Jennen-Steinmetz C (1986). "Residual Variance and Residual Pattern in Nonlinear Regression." *Biometrika*, **73**(3), 625–633. doi:10.1093/biomet/73.3.625.

Mankad S, Michailidis G, Banerjee M (2015). ***twostageTE**: Two-Stage Threshold Estimation*. R package version 1.3, URL http://CRAN.R-project.org/package=twostageTE.

Osborne C (1991). "Statistical Calibration: A Review." *International Statistical Review*, **59**(3), 309–336. doi:10.2307/1403690.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Robertson T, Wright F, Dykstra R (1988). *Order Restricted Statistical Inference*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, Chichester.

Rosenberger W, Haines L (2002). "Competing Designs for Phase I Clinical Trials: A Review." *Statistics in Medicine*, **21**(18), 2757–2770. doi:10.1002/sim.1229.

Tang R, Banerjee M, Michailidis G (2011). "A Two-Stage Hybrid Procedure for Estimating an Inverse Regression Function." *The Annals of Statistics*, **39**(2), 956–989. doi:10.1214/10-aos820.

Tang R, Banerjee M, Michailidis G, Mankad S (2015). "Two-Stage Plans for Estimating a Threshold Value of a Regression Function." *Technometrics*, **57**(3), 395–407. doi:10.1080/00401706.2014.940773.

A. Synthesis of theoretical results

A.1. One-stage confidence intervals

One-stage IR

Define $C_{d_I} = \left(4\sigma^2/f'(d_0)^2\right)^{1/3}$ and let $\mathcal{Z}$ follow the standard Chernoff distribution.
Then under mild conditions on the regression function and on the design density g, one can construct a $1-\alpha$ Wald-type confidence interval for d_0:

$$\left[d_I \pm n^{-1/3}\, \widehat{C_{d_I}}\, \widehat{g(d_0)}^{-1/3}\, q(\mathcal{Z}, 1-\alpha/2)\right], \tag{10}$$

where the hats denote consistent estimates and $q(\xi, \tau)$ is the lower τth quantile of a random variable ξ. Quantiles of ξ are provided for the user by specifying `data("chernoff_realizations", package = "twostageTE")` (see Tang *et al.* 2011, for more details on this result).

One-stage likelihood ratio

An alternative is to construct confidence intervals through likelihood ratio (LR) testing. Specifically, the hypotheses of interest are

$$H_0 : f^{-1}(\theta_0) = d_0 \leftrightarrow H_a : f^{-1}(\theta_0) \neq d_0. \tag{11}$$

Then, the LR test statistic is given by

$$2\log\lambda_I = 2\log\lambda_I(x_0) = 2\left[l_n(f_I, \hat{\sigma}) - l_n(f_{Ic}, \hat{\sigma})\right], \tag{12}$$

where $l_n(f, \sigma) = -(2\,\sigma^2)^{-1}\sum_{i=1}^{n}(Y_i - f(X_i))^2$, f_I is as before, f_{Ic} is the constrained isotonic regression of m under H_0 and $\hat{\sigma}$ a consistent estimate of σ. It is known that f_{Ic} uniquely exists (Banerjee 2000). The asymptotic distribution of $2\log\lambda_I$ under H_0 is given by: $2\log\lambda_I \xrightarrow{d} \mathbb{D}$, where $\mathbb{D}$ is a "universal" random variable not depending on the parameters of the model (see Banerjee and Wellner 2001; Banerjee 2009, for more details on this result). Realizations of $\mathbb{D}$ are provided for the user in a data frame called `RVforLR_realizations`, which can be loaded with `data("RVforLR_realizations", package = "twostageTE")`.

The result in Equation 12 allows us to construct a $1-\alpha$ LR-type confidence region for d_0:

$$\{x \in [a, b] : 2\log\lambda_I(x) \leq q(\mathbb{D}, 1-\alpha)\}. \tag{13}$$

The LR-type confidence region can be shown to be an interval and is typically *asymmetric* around d_I, unlike the Wald-type ones. Its main advantage is that only σ needs to be estimated for its construction, whereas for the other confidence intervals, estimation of $f'(d_0)$ is also needed, a significantly more involved task.

A.2. Two-stage confidence intervals

IR + locLinear

After using a high probability confidence interval for d_0 from stage one data, Tang *et al.* (2011) advocate the use of a bootstrapped second stage estimate. The main steps are

1. Allocate the second stage design points equally to the confidence interval end points L_1 and U_1, and obtain responses $Y_i' = f(L_1) + \epsilon'$ and $Y_i'' = f(U_1) + \epsilon''$.

2. Fit the second stage data $\{(L, Y_i'), (L, Y_i'')\}$ with the linear model $y = \beta_0 + \beta_1 x$. The second stage estimator of d_0 is given by $\hat{d}_0 = (\theta_0 - \hat{\beta}_0)/\hat{\beta}_1$.

3. Sample with replacement, responses $Y'*_i$ and $Y''*_i$ from Y_i' and Y_i''. Construct the bootstrapped estimator $\hat{d}*_0$, and calculate $R_n = n^{1/2}(\hat{d}\,*_0 - \hat{d}_0)$.

4. After repeating many times, the $1-\alpha$ Wald-type confidence interval for d_0 is given by

$$[\hat{d}_0 - n^{-1/2} q*_u, \hat{d}_0 - n^{-1/2} q*_l], \tag{14}$$

   where $q*_l$ and $q*_u$ are the lower and upper $\alpha/2$ quantiles for R_n.

Two-stage likelihood ratio

Similar to its stage one counterpart, the two-stage LR procedure uses second stage data to compute and invert the likelihood ratio statistic. The hypotheses of interest and LR test statistic remain as in Equations 11 and 12, and as before, we use the quantiles of the random variable $\mathbb{D}$ to form confidence intervals. Again, the two-stage LR confidence intervals can be asymmetric around the estimate of d_0, which is found by applying isotonic regression on second stage data. The two-stage likelihood ratio procedure shares the same advantages of the one-stage likelihood ratio procedure, namely, enhanced precision and avoidance of $f'(d_0)$ estimation.

IR + IR

From Step 3 of the two-stage procedure discussed in Section 2.2, the determination of $[L_1, U_1]$ is achieved through a high probability confidence interval for d_0 from stage one data. So, let $[L_1, U_1]$ be the following $1-\beta$ Wald-type confidence interval

$$[d_{1,I} \pm n_1^{-1/3} \hat{C}_{d_I} g_1(d_{1,I})^{-1/3} q(\mathcal{Z}, 1-\beta/2)] \cap [a, b], \tag{15}$$

where the computation of $\hat{C}_{d_I}$ involves estimating both σ^2 and $f'(d_0)$ and where β is a small positive number such as 0.01.

Then a Wald-type $1-\alpha$ asymptotic confidence interval for d_0 using second stage data is given by

$$[d_{2,I} \pm n^{-(1+\gamma_1)/3} \widehat{C_{d_{2,I}}} q(\mathcal{Z}, 1-\alpha/2)], \tag{16}$$

where $\widehat{C_{d_{2,I}}}$ is a function of the $\sigma, f'(d_0)$, the second stage budget allocation and sampling density at d_0.

Additional details and discussion can be found in Tang *et al.* (2015).

Affiliation:

Shawn Mankad
Department of Operations, Technology and Information Management
Cornell University
401H Sage Hall
Ithaca, NY 14850, United States of America
E-mail: smankad@cornell.edu

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
October 2015, Volume 67, Issue 3 *Submitted:* 2013-04-24
doi:10.18637/jss.v067.i03 *Accepted:* 2014-11-13