Next: Machine Learning for Regression Discontinuity

(This Draft: October, 2020)

Mark C. Long
(Corresponding Author)
University of Washington
Evans School of Public Policy and Governance
1100 NE Campus Parkway, Seattle, WA 98105
marklong@uw.edu
https://orcid.org/0000-0001-5500-9967

Jordan Rooklyn
Portland Water Bureau

Abstract

This paper develops a data-driven algorithm that simultaneously selects the polynomial specification and bandwidth combination that minimizes the predicted mean squared error at the threshold of a discontinuity. It achieves this selection by evaluating the combinations of specification and bandwidth that perform best in estimating the next point in the observed sequence on each side of the discontinuity. Our method learns the optimal specification. We illustrate this method by applying it to data with a simulated treatment effect to show its efficacy. We contrast the performance of this algorithm with those commonly used in the literature, notably Imbens and Kalyanaraman (2012) and Calonico, Cattaneo, and Titiunik (2014). Using simulated treatment effects applied to real data, we find that our method reduces squared prediction errors by over 20% and substantially reduces the rate of false negative results for modest-sized treatment effects. Finally, we utilize our method to reexamine the results of notable papers using RDD methods. We illustrate how and why use of this method produces different results from these published studies.

Keywords

Regression Discontinuity, Machine Learning, Program Evaluation

JEL Codes

C01, C1

Acknowledgments

Support for this research came from the U.S. Department of Education's Institute of Education Sciences (R305A140380) and a Eunice Kennedy Shriver National Institute of Child Health and Human Development research infrastructure grant (R24 HD042828) to the Center for Studies in Demography & Ecology at the University of Washington. Helpful comments were provided by Brian Dillon, Dan Goldhaber, Jon Smith, Aureo de Paula, Jake Vigdor, Ted Westling, and University of Washington seminar audience members. Excellent research assistance was provided by Ben Glasner and Tom Lindman.

Next: Machine Learning for Regression Discontinuity

Introduction

The fundamental question that must be answered by a researcher using the regression discontinuity design is: what is the most likely value of the dependent variable, y, as the policy assignment variable, x, reaches the threshold, T, that determines program participation? The researcher must estimate these values of y coming from each side of the threshold and, for the purpose of inference, construct a confidence interval around the discontinuity in these estimates. These are prediction problems.

The current wisdom in this field is that one should use low-order polynomials, most commonly linear regressions of y on x, within narrow ranges near the threshold. There are strong arguments for this approach. Higher-order polynomials may have a stronger fit to the whole of the data on a given side of the threshold, but their out-of-sample predictive power may be weak and standard errors on their predictions at the threshold, $\hat{y}_{x=T}$, may be quite large. Gelman and Imbens (2019) note that "we do not have good methods for choosing that order in a way that is optimal for the objective of a good estimator for the causal effect of interest" and "optimizing some global goodness-of-fit measure … is not closely related to the research objective of causal inference" (p. 447). Given these concerns, Calonico, Cattaneo, and Titiunik (2014) note, "the local linear RD estimator … is perhaps the preferred and most common choice in practice" (p. 912). Further, small bandwidths are warranted as the fundamental relationship between y and x may be different close to the threshold than it is far away. In such circumstances, values of y and x far away from the threshold may yield estimated polynomials with poor estimation of y for $x = T$.

The two most commonly used approaches in applied research are based on the methods articulated in Imbens and Kalyanaraman (2012) and Calonico, Cattaneo, and Titiunik (2014), which are henceforth labeled IK and CCT.[1] IK present an algorithm that selects the optimal bandwidth under asymptotic mean squared error loss for regression discontinuity analyses using a local linear regression. CCT develops an alternative bandwidth selection procedure and robust confidence intervals that correct for bias

[1] These papers have 2,262 and 1,833 citations, respectively, per Google Scholar as of October 2020.

introduced by the bandwidth selection process.[2] While these approaches are attractive and data-driven, they do not solve the problem of identifying the optimal polynomial order and they leave it to user discretion as to what kernel to use in weighting the regression. The default settings on software developed to implement these popular methods assume that the user wants to implement a local linear regression with a triangular weight. Guidance is often provided to the user to try a quadratic specification, different kernels, and smaller and larger bandwidths. Yet, this suggestion yields uncertainty as to what is the preferred specification and in practice users treat the default settings as optimal and other choices, if considered at all, as robustness checks.

A local linear or quadratic specification may yield a poor approximation if the underlying relationship between y and x is of higher order, and higher-order specifications are likely to require larger bandwidths for reliable parameter estimation. Finally, these standard approaches do not consider whether a simple or weighted average of prior observation near the threshold (i.e., a zero-order polynomial) might produce a more precise and accurate prediction.

This paper introduces a fully data-driven procedure for selecting the optimal polynomial specification, bandwidth, and kernel. Our algorithm "learns" the optimal bandwidth by evaluating the performance of wider and narrower widths in making out-of-sample predictions of y and adjusts the preferred bandwidth accordingly. It selects the optimal specification and kernel by choosing the combination that is predicted to have the lowest squared prediction error at the threshold based on learned experience.

Through a series of simulations, we show that our algorithm yields estimates of the local average treatment effects (LATEs) that have smaller mean squared errors than the IK and CCT methods using default settings of popular software using their methods. We then apply

[2] Suppose that one uses a local linear regression applied to data generated by a process that is not globally linear. The broader the bandwidth, the more bias is introduced as the linear regression captures more of the nonlinear global process. However, a broader bandwidth reduces variance in the estimates of the parameters as it captures more data. Algorithms, such as IK, that minimize mean squared error, which equals bias-squared plus variance, will trade bias that comes from a larger bandwidth with variance that is reduced by a larger bandwidth. The CCT approach addresses the bias that results from such bandwidth selection procedures.

our method to set of recent papers in the economics and public policy literatures to show how and whether using this method alters the findings.

The "Next" Algorithm

The ideas that motivates our algorithm are that (a) our goal is to predict y when x reaches threshold T (i.e., $\hat{y}_{x=T}$) and (b) our data provide us experience with how well the next value of y can be predicted as a function of the prior series of observations of (x, y). We let this experience teach us the best method to use the prior series of observations, i.e., the best bandwidth, kernel weights, and polynomial order.

The following is an outline of the steps that are used in our algorithm, and this outline is explained in detail below. Step 1: Identify the best bandwidth, b_x, for predicting y_x for a particular value of x on the left side of the threshold, using a selected polynomial order and kernel. Repeat this procedure for all but the first several x values on the left side of the threshold. Step 2: Remove noise by smoothing this series of bandwidths, $\breve{b}_x$, and predict the best bandwidth at the threshold, $\hat{\breve{b}}_{x=T}$. Step 3: Using the series of smoothed-bandwidth values, predict $\hat{y}_x$ at various levels of x, and compute squared prediction errors, $s_x = (y_x - \hat{y}_x)^2$. Step 4: Smooth this series of squared prediction errors, $\breve{s}_x$, and predict the squared prediction error at the threshold, $\hat{\breve{s}}_{x=T}$. Step 5: Repeat Steps 1-4 for different polynomial orders and kernels and identify the polynomial order and kernel that produces the minimum predicted squared prediction error at the threshold. Step 6: Repeat Steps 1-5 for the right side of the threshold. Step 7: Use the optimal polynomial, kernel, and bandwidth for each side of the threshold and standard ordinary least squares methods for identifying the magnitude of the discontinuity in y at the threshold.

Step 1

We begin by sorting the data by x from smallest to largest. If multiple observations have the same value of x, we collapse the data by x, creating the average value of y for each x, and creating a frequency weight that is equal to the number of observations sharing that x value. All subsequent regressions used in finding the optimal bandwidth and resulting prediction errors are frequency weighted. In the applications discussed below, we evaluate the performance of polynomial orders from 1 to 3 and we evaluate the performance of the uniform, triangular, and Epanechnikov kernels.[3] Let p denote the maximum polynomial

[3] In the *Stata* code that we developed for users to implement this algorithm (Long and Rooklyn, 2020), we allow the user to vary the desired minimum and maximum orders that

order that will be considered. We first seek to predict y for the p+4th distinct value of x. For illustration purposes, suppose p=1, and thus we would begin with a prediction of y for the 5th observation of x. Suppose that the first five observations, (x, y), are as follows: (12, 7), (15, 11), (21,10), (23,15), and (27,9). We measure bandwidths in terms of the percentage of prior observations that are included in the prediction of the next observation. We consider three possible bandwidths to predict the 5th value of y: 100% (i.e., using the first four observations), 75% (i.e., using just the 2nd, 3rd, and 4th observations), and 50% (i.e., using just the 4th and 5th observations). Using a uniform kernel and a linear regression applied to these observations, these three bandwidths would yield the following predictions of the 5th value of y: 16.8, 13.5, and 19.[4] The best bandwidth for the 5th observation is the one that produces a prediction of y that is closest to the 5th observed value of y. In this case, since the 5th value of y is 9, the best bandwidth, $b_{x=27}$, would be 75%.

Step 2:

We assume that the observed series of optimal bandwidths, b_x, is equal to a latent "true" bandwidth (that evolves as x increases) plus noise (produced by noise in y_x). We attempt to remove the noise and thereby recover the latent bandwidth series, $\breve{b}_x$, by using an exponential-weighted average.[5] We define $\breve{b}_x = \alpha_{BW} b_x + (1 - \alpha_{BW})\breve{b}_{x-}$, where x- denotes

---

are considered and allow the user to set the kernel type rather than have it chosen by the data.

[4] For the triangular and Epanechnikov weights, we slightly extend the range so that each observation gets positive weight. So, for example, when computing the triangular weighted average of the first four observations, we extend the range by five-fourths. More generally, we extend the range by $e = (c + 1)/c$, where c is number of distinct observations that we wish to include. The triangular weights are given by $(1 - (x_{max} - x) \;/\; (e \times (x_{max} - x_{min})))$, where x_{min} and x_{max} are the minimum and maximum values of x amongst the observations being given weight. Note that this formulation accounts for irregular spacing of the values of x. For our four observations, whose values of x are 12, 15, 21, and 23, the corresponding triangular weights would be 0.20, 0.42, 0.85, and 1. The Epanechnikov weights are given by $0.75 \times \left(1 - \left((x_{max} - x) \;/\; (e \times (x_{max} - x_{min}))\right)^2\right)$. For our four observations, the corresponding Epanechnikov weights would be 0.27, 0.50, 0.73, and 0.75. With this extender, as the number of observations that are included in the weighted average goes to infinity, the weight on the first observation in the included series goes to 0.

[5] If we were to assume that the observed optimal bandwidth series is generated by a random walk plus noise, then exponential smoothing generates optimal forecasts (Muth, 1960). However, note that it is not appropriate to assume a pure random walk as the

the value of x of the prior observation and α is the smoothing parameter.[6] Higher values of α_{BW} would indicate that the movement of $\breve{b}_x$ is highly responsive to new information contained within b_x. We let the data reveal the optimal value of the smoothing parameter. To do this, we evaluate the performance of a variety of candidate α_{BW}'s and choose the α_{BW} with the best performance.[7] Performance is measured by the extent to which the resulting forecast of the next level of the latent bandwidth predicts the observed next bandwidth. Here we are assuming that a deviation would reflect noise. The α_{BW} with the best performance is the one that minimizes mean squared forecast error. Once α_{BW} has been selected, we can produce a weighted average of all of the observed optimal values of b_x on the left side of the threshold to predict the best bandwidth for predicting y at the threshold, $\hat{\breve{b}}_{x=T}$. It is this bandwidth that will ultimately be used in Step 7 to identify the causal effect of the discontinuity on outcomes.

Step 3:

In this step, we begin the process of generating results to aid in selecting between possible candidate polynomial orders and kernels. Using the series of smoothed-bandwidth values, we predict y at various levels of x, $\hat{y}_x$, and we compute squared prediction errors, $s_x = (y_x - \hat{y}_x)^2$. If this particular polynomial order and kernel is a good performer, then it will have generated good predictions of the observed series of y values. Note that this process produces an out-of-sample forecast of y. For example, $\hat{y}_{x=32}$ is generated only from observations of (x, y) for $x < 32$. As a consequence, high-order polynomials will tend to perform poorly in generating out-of-sample predictions for precisely the reasons articulated

---

bandwidth is bounded (e.g., we cannot use more than 100% of the prior observations). Nonetheless, we assume that, as a practical matter, using this exponential smoothing is sufficient for our purposes.

[6] Our algorithm adds an additional element to capture non-uniform spacing between the observed values of x. To do this, we construct weights for each observation that are equal to $\alpha^{(1-(x-\underline{x})/(\overline{x}-\underline{x}))}$, where $\underline{x}$ and $\overline{x}$ respectively denote the lowest and highest observed values of x in the series of optimal bandwidths, b_x, on the left (or right) side of the threshold. We then take a weighted average of prior values of b_x to yield $\breve{b}_x$.

[7] We evaluate the following 44 candidate levels of α_{BW}: 1 (which produces no smoothing), 0.96, 0.92, …, 0.12, 0.10, …, 0.02, 0.01, 0.005, 0.0025, 0.001, 0.0004, 10^{-04}, 10^{-05}, 10^{-06}, 10^{-7}, 10^{-9}, 10^{-11}, 10^{-14}, 10^{-17}, 10^{-20}, and 10^{-23}. Appendix Figure 1 shows the resulting possible weighting schemes that could be applied to b_x in the range x=1 to x=16 so as to generate $\breve{b}_{x=16}$. This figure reveals that this set of candidate weighting schemes produces a fairly comprehensive coverage of the possible weighting schemes.

by Gelman and Imbens (2019). Yet, if higher-orders are indeed helpful in making accurate out-of-sample predictions, then they will be selected by the algorithm.

Step 4:
Since we want to select the polynomial order and kernel that performs best at the threshold, we again proceed to eliminate idiosyncratic noise in the series of s_x so as to recover the latent value, $\breve{s}_x$, which is projected, out-of-sample, to the threshold, $\hat{\breve{s}}_{x=T}$. We again use exponential smoothing, but this time we set the smoothing parameter, α_{SPE}, to 0.02 as a default setting. As shown by the dashed line in Appendix Figure 1, this level of smoothing will place one-third of the total weight on the rightmost 10% of observations (assuming observations are uniformly distributed) and will place two-thirds of the total weight on the rightmost quarter of observations. That is, the bulk of the selection of the best order/kernel comes from performance making out-of-sample predictions of observations near to the threshold.

In the language of machine learning, α_{SPE} is a "hyperparameter" as it is not learned by the data. We cannot allow this parameter to be selected by the data and vary across polynomial orders and kernels because it would create an unfair competition between these specifications. If we were to do so, then a particular order/kernel might end up being selected simply because it put less weight on a particular observation that is an outlier and hard to predict.[8] Our choice of $\alpha_{SPE} = 0.02$ is arbitrary and the user might want to select different values of α_{SPE} to yield robustness checks.

Step 5:
Next, we repeat Steps 1-4 for different polynomial orders and kernels. For each, we compute $\hat{\breve{s}}_{x=T}$. We identify the polynomial order and kernel that produces the smallest $\hat{\breve{s}}_{x=T}$ and this combination of order and kernel is then used for the left side of the threshold in generating the causal impact estimates.

Step 6:
We then start over and repeat Steps 1-5 for the right side of the threshold. In this case, we are moving in the opposite direction, right to left, i.e., moving from the higher values of x leftward towards the threshold. This procedure generates an optimal bandwidth, polynomial

[8] If we allow α_{SPE} to be learned and separately estimated for each kernel/polynomial order, it is possible for a particular kernel / polynomial order to have a lower value of $\hat{\breve{s}}_{x=T}$ despite have higher values of s_x than an alternative specification at every distinct value of x.

order, and kernel for the right side, and the algorithm allows these choices to be different in all dimensions from the choices made on the left side. While this might sound unconventional, it is reasonable as the treatment may have affected the shape of the relationship between x and y, which would yield different optimal choices for the right and left sides.

Step 7:
The final step is to run an ordinary least squares regression to generate the LATE estimate. This regression is applied to the original dataset (i.e., not collapsed by distinct values of x) using the span of data within the selected bandwidths and with the data-chosen kernel weights and polynomial order. We center the threshold at 0 by generating $\tilde{x} = x - t$. The resulting regression would look like the following assuming that the selected optimal polynomial orders are 2 for the left and 3 for the right sides:

$$y = \beta_{L0}L + \beta_{L1}L\tilde{x} + \beta_{L2}L\tilde{x}^2 + \beta_{R0}R + \beta_{R1}R\tilde{x} + \beta_{R2}R\tilde{x}^2 + \beta_{R3}R\tilde{x}^3 + \varepsilon$$

L is an indicator for $\tilde{x} < 0$ and R is an indicator for $\tilde{x} \geq 0$ (assuming that those with $\tilde{x} = 0$ have the same treatment status as those on the right side of the threshold). The estimate effect of the treatment is given by the difference between the left- and right-side intercepts, i.e., $\beta_{R0} - \beta_{L0}$. Robust standard errors are used for inference.

Illustration of the Next Algorithm

Table 1 illustrates the Next algorithm. In this illustration, we show hypothetical data for 17 observations on the left side of a threshold, which is at x=75. We illustrate the results assuming that we are currently considering a first-order polynomial (i.e., a linear regression) with a uniform kernel, and further suppose that first-order is the maximum polynomial order that we are willing to consider, thus p=1, and we would begin with a prediction of y for the 5$^{\text{th}}$ observation of x. As shown in the 5$^{\text{th}}$ row, and discussed above, the bandwidth that performs best in predicting the 5$^{\text{th}}$ observation is 75%.

We then assess the performance of this bandwidth in predicting the 6$^{\text{th}}$ observation. We run a linear regression using 75% of the first five observations – rounded, this results in using four observations, specifically the 2$^{\text{nd}}$, 3$^{\text{rd}}$, 4$^{\text{th}}$, and 5$^{\text{th}}$ observations. The resulting regression line is shown in blue in the bottom-left quadrant of Figure 1. Extrapolating this regression line gives our prediction of the next observation, and this prediction is shown by the left-most open circle on Figure 1. Returning to Table 1, we see that our out-of-sample prediction for the 6$^{\text{th}}$ observation of y is 11.0, whereas the actual value of y is 22, producing a squared prediction error, s, of 121.

We repeat this process for each of the subsequent observations on the left side of the threshold. Figure 2 illustrates the resulting optimal bandwidths, squared prediction errors, and the smoothing of these series. The final value of the smoothed squared prediction error series, 61.6, gives our prediction of the squared prediction error at the threshold $\hat{\tilde{s}}_{x=T}$ assuming we were to use a linear regression composed of 59% of the observations on the left side, which is the final value of the smoothed optimal bandwidth, and with uniform kernel weights. If this value, 61.6, is lower than all values of $\hat{\tilde{s}}_{x=T}$ using other polynomial orders or other kernels, then this is the optimal specification that would be chosen by this algorithm for use in producing causal estimates.

Comparison of the Next Algorithm with the approaches of Ludwig and Miller (2007) and Imbens and Lemieux (2008)

Our method bears similarity to the approaches in Ludwig and Miller (2007) and Imbens and Lemieux (2008), which are described as follows by Lee and Lemieux (2010, p. 321):

> … Jens Ludwig and Douglas Miller (2007) and Imbens and Lemieux (2008) have proposed a "leave one out" procedure aimed specifically at estimating the regression function at the boundary. The basic idea behind this procedure is the following. Consider an observation i. To see how well a linear regression with a bandwidth h fits the data, we run a regression with observation i left out and use the estimates to predict the value of Y at $X = X_i$. In order to mimic the fact that RD estimates are based on regression estimates at the boundary, the regression is estimated using only observations with values of X on the left of X_i $(X_i - h \leq X < X_i)$ for observations on the left of the cutoff point $(X_i < c)$. For observations on the right of the cutoff point $(X_i \geq c)$, the regression is estimated using only observations with values of X on the right of X_i $(X_i < X \leq X_i + h)$.
>
> Repeating the exercise for each and every observation, we get a whole set of predicted values of Y that can be compared to the actual values of Y. The optimal bandwidth can be picked by choosing the value of h that minimizes the mean square of the difference between the predicted and actual value of Y.

Our Next procedure contains several differences from these approaches. We allow the bandwidth to vary on the two sides of the discontinuity. We base bandwidth choice on a forecast of the mean squared error at the threshold, rather than selecting the bandwidth

that minimizes the mean squared error of the whole series of y values. Finally, we select polynomial order and kernel weights based on this forecast.

Simulation and Comparison of the Next Algorithm with IK and CCT

To contrast the performance of our Next algorithm with the IK and CCT approaches, we used data from Jacob et al. (2012). These data include 7th and 8th grade math assessments from a balanced panel of 2,767 students and Jacob et al. applied a simulated treatment effect to evaluate RD methods. The raw data are plotted in Appendix Figure 2. We evaluate simulated treatment effects at 54 values of the pretest score, ranging from 189.5 to 242.5 (the 2nd and 98th percentiles of the pretest score distribution), and these points are shown as tick marks on the x-axis in this figure. We add a simulated treatment to the posttest score for students above the threshold. We use the following 37 simulated treatment effects, where σ denotes one standard deviation of the posttest score distribution and θ denotes (student's pretest score minus the threshold) / standard deviation of the pretest scores:

	$LATE = 0$	$LATE = 0.2\sigma$	$LATE = 0.5\sigma$	$LATE = \sigma$	$LATE = -0.2\sigma$	$LATE = -0.5\sigma$	$LATE = -\sigma$
Flat:	0	0.2σ	0.5σ	σ	-0.2σ	-0.5σ	$-\sigma$
Growing Linearly:	$0.2\sigma\theta$, $0.5\sigma\theta$, $\sigma\theta$, $-0.2\sigma\theta$, $-0.5\sigma\theta$, $-\sigma\theta$	0.2σ $+0.2\sigma\theta$	0.5σ $+0.5\sigma\theta$	σ $+\sigma\theta$	-0.2σ $-0.2\sigma\theta$	-0.5σ $-0.5\sigma\theta$	$-\sigma$ $-\sigma\theta$
Growing Exponentially:	$0.2\sigma\theta^2$, $0.5\sigma\theta^2$, $\sigma\theta^2$, $-0.2\sigma\theta^2$, $-0.5\sigma\theta^2$, $-\sigma\theta^2$	0.2σ $+0.2\sigma\theta^2$	0.5σ $+0.5\sigma\theta^2$	σ $+\sigma\theta^2$	-0.2σ $-0.2\sigma\theta^2$	-0.5σ $-0.5\sigma\theta^2$	$-\sigma$ $-\sigma\theta^2$
Fading:		$0.2\sigma/2^\theta$	$0.5\sigma/2^\theta$	$\sigma/2^\theta$	$-0.2\sigma/2^\theta$	$-0.5\sigma/2^\theta$	$-\sigma/2^\theta$

In the case of "Flat" treatment effects, it is assumed that every observation to the right of the discontinuity gets the same treatment effect, which vary in size from -1 to 1 s.d. "Growing Linearly" are simulated effects that grow linearly as x moves away from T – in this case, the treatment effect not only has an intercept effect (assuming LATE≠0), but also changes the slope of the relationship between x and y. For "Growing Exponentially", we assume the effect grows by the square of the standard deviation increase of x above T, whereas for "Fading", we simulate an effect that is cut in half for every standard deviation increase of x above T. This set of possible effects fairly comprehensively captures the ways in which a treatment could affect the relationship between x and y.

We compare the performance of the Next algorithm to the IK and CCT approaches. For this analysis, we use the following software packages, which were run on Stata: *rd* written by Nichols (2011), which is used to estimate the IK algorithm, and *rdrobust* written by Calonico, Cattaneo, Farrell, and Titiunik (2018), which is used to estimate the CCT algorithm.[9] Using default settings, *rd* estimates the LATE using a triangular-kernel weighted linear regression for observations within the IK bandwidths. Using default settings, *rdrobust* also uses a triangular-kernel weighted linear regression with "MSE-optimal point estimation using a common bandwidth on both sides of the cutoff" (CCTF, p. 400) and uses a local quadratic regression to correct bias.

With 54 locations of the possible treatment, 37 different simulated effects, and 3 methods tested, we produce a total of 5,994 treatment estimates. Figure 3 graphically displays the estimates and confidence intervals for the first of these possible treatment effects, i.e., where the treatment effect is a flat 0.[10] For this case, the Next algorithm outperforms the IK and CCT approaches with root mean squared errors of 1.44, 1.80, and 2.11 respectively. This result is generally held as shown in panel A of Table 2. Across the 5,994 treatment estimates, we find significantly lower average root mean squared errors for our method (1.11), outperforming IK by 21% (1.35) and CCT by 41% (1.57). This better performance is achieved for all types of effects (i.e., Flat, Growing Steadily, Growing Linearly, Growing Exponentially, and Fading).

Panel B of Table 2 shows that each method, as expected, produces a false rejection of the true (simulated) LATE about 5% of the time using a 95% confidence interval. Yet, as shown in Panel C of Table 2, the Next algorithm has much lower rates of false negatives. When the absolute value of the simulated treatment effect is 0.2σ, our Next algorithm rejects this LATE in 30% of the cases, whereas the false rejection rates for IK and CCT are 57% and 61% respectively. This means that use of such standard methods is much more prone to fail to reject zero when the true LATE is modest. When the true (simulated) LATE is moderately large (0.5σ or -0.5σ), we find a false negative rate of 3% for our Next algorithm versus 6% for IK and 12% for CCT. When the true (simulated) LATE is large (σ or $-\sigma$), we find no false negatives using any of the three methods.

Having now (hopefully) established the utility of our method, in the next two sections we apply the method to two prominent papers in the RD literature.

---

[9] We use the versions of these packages that were available in May 2020.

[10] Appendix Figure 2 shows the full set of estimates.

Applying the Next Algorithm to Prior Published Studies

Next Algorithm Applied to Lee (2008)

Lee (2008) evaluates the impact of party incumbency on the probability that the incumbent party will retain the district's seat in the next election for the U.S. House of Representatives. In this analysis, x is defined as the Democratic vote share in year t minus the vote share of the "Democrats strongest opponent (virtually always a Republican)" (p. 686) and y is defined as the Democratic Party's vote share in year $t+1$.[11] The key identifying assumption is that there is a modest random component to the final vote share (e.g., rain on Election Day) that cannot be fully controlled by the candidates and that, effectively, "whether the Democrats win in a closely contested election is...determined as if by a flip of a coin" (p. 684). Lee used a 4^{th} order polynomial in x for each side of the discontinuity while restricting the data to an arbitrarily selected bandwidth of $-0.25 < x < 0.25$, i.e., excluding cases where the winning party won by more than 25 percentage points. Lee concluded that the impact of incumbency on vote share was 0.077 (s.e. = 0.011). That is, being the incumbent raised the expected vote share in the next election by 7.7 percentage points.

This study was reexamined by Lee and Lemieux (2010) and Imbens and Kalyanaraman (2011). Lee and Lemieux estimate the treatment effect by using polynomials ranging from order zero (i.e., the average of prior values) up to a 6^{th} order polynomial with the same order polynomial estimated for both sides of the discontinuity and with bandwidths ranging from 1% to 100% (i.e., using all of the data). For each bandwidth, they identify the "optimal order of the polynomial" by selecting the one with the lowest value of the Akaike Information Criterion (AIC). They identify an optimal bandwidth "by choosing the value of h that minimizes the mean square of the difference between the predicted and actual value of Y" (p. 321).[12] As shown in Table 2 of their paper, using the optimal bandwidth, which is roughly 5%, and the optimal order of the polynomial for this

---

[11] Lee's data comes from U.S. Congressional election returns from 1946 to 1998. We obtained these data on January 2, 2015 from http://economics.mit.edu/faculty/angrist/data1/mhe/lee.

[12] While this model selection procedure has the nice feature of selecting the specification and bandwidth "optimally", it has two limitations: (1) it suggests that a particular order of the polynomial and bandwidth be used on both sides of the discontinuity, and (2) the AIC evaluates the fit of the polynomial at all values of x, and doesn't attempt to evaluate the fit of the polynomial as x approaches the threshold, which is more appropriate for the RDD treatment effect estimation.

bandwidth (quadratic), the estimated effect of incumbency on the Democratic party's vote share in year $t+1$ is 0.100 (s.e. = 0.029). Imbens and Kalyanaraman found that the optimal bandwidth for a linear specification on both sides was 0.29 and using this bandwidth/specification produced an estimate of the treatment effect of 0.080 (s.e. = 0.008).

Lee's data poses a computational challenge for us as there are 2,740 (3,819) observations on the left (right) side of the discontinuity. Computing the best bandwidth and squared prediction errors for this number of outcomes takes a good deal of time – processing time increases by roughly the square of the number of observations. We hasten the process by coarsening the data into 200 bins, averaging x and y within each bin. Note that this binning is only used to identify the optimal bandwidths, kernels, and polynomial orders; once identified, we run the regression applying the chosen specification on the full, unbinned data. The user should be aware, however, that binning may yield bandwith, kernel, and polynomial order choices that are somewhat less than optimal, with greater coarsening leading to less optimal calculations.

The results of our Next method are shown in Figure 4. We find that the best specification uses a linear specification on both sides, applied to data in the range $-0.37 < x < 0.17$ and with an Epanechnikov (uniform) kernel weight applied to the left (right) side. Using this specification yields our estimate of treatment effect as 0.084, which is slightly larger than Lee's, and our estimate has a much smaller standard error (s.e. = 0.001).

We additionally re-examine the result shown in Figure 2a of Lee (2008), where y is an indicator variable that equals 1 if the Democratic Party won the election in that district in year $t+1$.[13] Lee applies "a logit with a 4^{th} order polynomial in the margin of victory, separately, for the winners and the losers" (Lee, 2001, p. 14) using all of the data on both sides of the discontinuity, and finds that incumbency raises the likelihood of the incumbent party maintaining the seat by 45.0 percentage points (s.e. = 3.1 pct. pt.).

Given the dichotomous nature of the dependent variable, we adapt the Next algorithm to use a logit specification with a polynomial in x and an indicator for being above the threshold as the independent variables. We again coarsen the data to 200 bins for identifying the (nearly) optimal specification. Given that binning results in fractional values of y that lie in the interval from 0% to 100%, we use a generalized linear model using a logit link

[13] The numerical estimates corresponding to this figure are included in Lee (2001).

function as recommended by Papke and Woolridge (1996) for modeling proportions. We test polynomial orders ranging from 1 to 4.

We find that the best logit specification is linear (quadratic) in x on the left (right) side, applied to data in the range $-0.19 < x < 0.37$, and with an Epanechnikov kernel weight applied to both sides. From this specification, we estimate that the Democratic Party has a 14.4% chance of winning the next election if they were barely below 50% on the prior election, and a 60.3% chance of winning the next election if they are just to the right of the discontinuity. Figure 5 shows the estimated curves. Our estimate of the treatment effect (i.e., barely winning the prior election) is 46.0 percentage points (p-value < 0.001) pt.), which is slightly larger than Lee's estimate.

Next Algorithm Applied to Brollo, Nannicini, Perotti, & Tabellini (2013)

Brollo et al. estimate the effect of additional government revenues on political corruption. A revenue windfall received by a local government provides a moral hazard that facilitates political corruption; "with a larger budget size, the incumbent has more room to grab political rents without disappointing rational but imperfectly informed voters" (p. 1760). They test this theory using data from Brazil where "transfers to municipalities…change exogenously and discontinuously at given population thresholds" (p. 1760). They evaluate the average change in frequency and severity of corruption at these thresholds for 1,202 municipalities utilizing four outcome measures:

— "Broad corruption" defined as the presence of irregularities, including illegal procurement practices, fraud, favoritism, over-invoicing, diversion of funds, and paid but not proven expenses.
— "Narrow corruption" defined as "severe irregularities that are also more likely to be visible to voters" (p. 1774), including severe illegal procurement practices, fraud, favoritism, and over-invoicing.
— "Broad corruption funds as a fraction of the total amount of the audited budget."
— "Narrow corruption funds as a fraction of the total amount of the audited budget."

Brollo et al. identify the effect of a revenue windfall on corruption using the discontinuity in a global third-order polynomial at the population threshold for receipt of additional revenues. In their online appendix, Brollo et al. runs a sensitivity analysis with respect to the functional form of the control function in population size. Specifically, they redefine the function "as a spline third-order polynomial, as a second-order polynomial (spline or not), and a fourth-order polynomial (spline or not)" and conclude that "(a)ll of these

robustness exercises support the validity of the results" (p. 1784). In this paper, we compare our method to their main findings.

Their reduced form (i.e., intent-to-treat) estimated effects are shown in their Figure 2 (which we reproduce below as Panel A of Figure 6). They find significant 16.3 and 17.0 percentage point increases in the likelihood of broad and narrow corruption respectively, and significant 2.9 and 1.9 percentage point increases in broad and narrow corruption funds as fractions of the total amount of the audited budget respectively.[14] For graphical display, they use wide population bins that include 250-person intervals. Despite these wide bins, these scatterplots show a lot of variability in the underlying relationships between population and corruption on either side of the revenue windfall thresholds, which makes this an interesting case study for using our method.

In using the Next algorithm, we coarsened the data to 100 bins data for the purposes of identifying the optimal specification and the graph of the results, while the regression is run on unbinned data. For the first two outcomes, which are dichotomous, we used a logit specification, while using an ordinary least squares specification for the latter outcomes. Panel B of Figure 6 shows our estimates of the treatment effects using our method. We found that a first-order polynomial (linear) was appropriate for both sides of the threshold for all four outcomes. Our estimates of the treatment effect are similar to Brollo et al.; we find 15.3 and 18.7 percentage point increases in the likelihood of broad and narrow corruption respectively, and significant 2.8 and 1.9 percentage point increases in broad and narrow corruption funds as fractions of the total amount of the audited budget respectively.[15]

Despite the similarity in impact estimates, there are clear differences in the selected bandwidths, with much smaller bandwidths selected by the Next algorithm on the left of the discontinuity. Such differences could be of more substantive importance in another case as we show in the next example.

Next Algorithm Applied to Chen, Ebenstein, Greenstone, and Li (2013)

In the appendix, we present the results of applying the Next algorithm to the data in Chen et a1. (2013). This paper has received much scrutiny (e.g. Gelman and Zelizer,

[14] These numerical estimates are not found in their paper, but rather were generated from the code and data they provide online with their article.

[15] The *p*-values for these four estimates are: 0.00, 0.04, 0.12, and 0.02.

2015) due to Chen et al.'s use of a global cubic polynomial specification with a treatment discontinuity. Our Next algorithm is not well suited to these data as there are only 12 observations on the left and 10 observations on the right side of the threshold. Consequently, there is not sufficient scope for Next to "learn" the optimal bandwidth. This suggests that Next is best suited when there are a larger number of distinct values of x on each side of the discontinuity.

Conclusion

This paper presents a new algorithm for identifying the optimal bandwidth, polynomial order, and kernel weight for use in estimating local average treatment effects in a regression discontinuity design. We name this algorithm "Next" to emphasize that RDD estimation is a forecasting problem where the goal is to predict y as x approaches the threshold from each side of the discontinuity. We show that the Next algorithm has lower mean squared prediction errors and lower rates of false negatives when confronted with a variety of types of simulated effects when compared with the IK and CCT procedures, which assume (as default settings in popular statistical software packages) a local linear specification with the same bandwidth and kernel weights used on both sides of the discontinuity. We illustrate the algorithms use when applied to notable papers in the RDD literature. We note that this algorithm is best suited when there are a large number of distinct values of x on each side of the threshold as such thick data gives the Next algorithm sufficient observations with which to "learn" the optimal specification.

References

Brollo, Fernanda, Tommaso Nannicini, Roberto Perotti, and Guido Tabellini. (2013). The Political Resource Curse. American Economic Review, 103(5): 1759-96.

Calonico, S., M. D. Cattaneo, M. H. Farrell, and R. Titiunik, 2018. RDROBUST: Stata Module to Provide Robust Data-Driven Inference in the Regression-Discontinuity Design. https://ideas.repec.org/c/boc/bocode/s458483.html.

Calonico, S., M. D. Cattaneo, and R. Titiunik. 2014. Robust Nonparametric Confidence Intervals for Regression-Discontinuity Designs. *Econometrica*, 82(6): 2295-2326.

Chen, Y., A. Ebenstein, M. Greenstone, and H. Li. (2013). Evidence on the Impact of Sustained Exposure to Air Pollution on Life Expectancy from China's Huai River Policy. *Proceedings of the National Academy of Sciences* 110, 12936–12941.

Gelman, A., and A. Zelizer. 2015. Evidence on the Deleterious Impact of Sustained Use of Polynomial Regression on Causal Inference. *Research & Politics*, 2(1), 1-7.

Gelman, A. and G. Imbens 2019. Why High-Order Polynomials Should Not Be Used in Regression Discontinuity Designs, *Journal of Business & Economic Statistics*, 37(3): 447-456.

Imbens, G., and K. Kalyanaraman. 2012. Optimal Bandwidth Choice for the Regression Discontinuity Estimator. *Review of Economic Studies*, 79(3): 933-959.

Imbens, G. W., and T. Lemieux. 2008. Regression Discontinuity Designs: A Guide to Practice. *Journal of Econometrics*, 142(2): 615–35.

Jacob, R., Zhu, P., Somers, M., and Bloom, H. 2012. A Practical Guide to Regression Discontinuity. *MDRC*.
http://www.mdrc.org/sites/default/files/regression_discontinuity_full.pdf.

Lee, D.S. 2001. The Electoral Advantage to Incumbency and Voters' Valuation of Politicians' Experience: A Regression Discontinuity Analysis of Elections to the U.S. *National Bureau of Economics Research*, Working Paper 8441.

Lee, D.S. 2008. Randomized Experiments from Non-Random Selection in U.S. House Elections. *Journal of Econometrics*, 142, 675-697.

Lee, D. S., and Lemieux, T. 2010. Regression Discontinuity Designs in Economics. *Journal of Economic Literature*, 48, 281-355.

Long, M.C., and J. Rooklyn. 2020. NEXT: Stata module to perform regression discontinuity. https://ideas.repec.org/c/boc/bocode/s458238.html

Ludwig, J., and D. L. Miller. 2007. Does Head Start Improve Children's Life Chances? Evidence from a Regression Discontinuity Design." *Quarterly Journal of Economics*, 122(1): 159–208.

Muth, J. F. 1960. Optimal Properties of Exponentially Weighted Forecasts. *Journal of the American Statistical Association*, 55, 299– 306.

Nichols, A. 2011. rd 2.0: Revised Stata Module for Regression Discontinuity Estimation. http://ideas.repec.org/c/boc/bocode/s456888.html

Papke, L.E., and J. Wooldridge. 1996. Econometric Methods for Fractional Response Variables with an Application to 401(k) Plan Participation Rates. *Journal of Applied Econometrics* 11: 619–632.

Table 1:
Illustration of the Next Algorithm for the Left-Side of a Threshold at x=75

Observation	x	y	Optimal bandwidth to predict this observation b_x	Smoothed optimal bandwidth $\check{b}_x$	Out-of-sample prediction of y using smoothed optimal bandwidth $\hat{y}_x$	Squared prediction error s_x	Smoothed squared prediction error $\check{s}_x$
1	12	7					
2	15	11					
3	21	12					
4	23	15					
5	27	9	0.75	0.75			
6	32	22	1.00	0.88	11.0	121	121
7	33	27	0.33	0.69	19.7	73	96
8	38	19	1.00	0.78	28.8	96	96
9	43	31	0.50	0.71	26.7	18	63
10	44	29	0.89	0.75	29.8	1	43
11	48	37	0.30	0.66	32.9	17	35
12	51	42	0.36	0.61	38.6	12	28
13	56	61	0.25	0.55	43.5	307	119
14	60	54	0.85	0.60	57.0	21	87
15	64	64	0.21	0.54	62.2	3	60
16	68	60	1.00	0.61	68.8	159	92
17	73	74	0.50	0.59	72.2	3	62

Notes: The "optimal bandwidth" denotes the share of the prior span of x (between the value of x for the first observation and the value of x of the nearest observation) that produces the minimum squared error in predicting y for this outcome using a linear regression (i.e., first-order polynomial) with a uniform kernel weight applied to the prior observations in this span. Smoothing is done using an exponential weighted average of the series to that point.

Table 2:
Comparison of the Performance of the Methods

Panel A: Root Mean Squared Error

	Coef.	(s.e.)	p-value	Coef.	(s.e.)	p-value	Coef.	(s.e.)	p-value	Coef.	(s.e.)	p-value	Coef.	(s.e.)	p-value
Constant (i.e., mean for Long and Rooklyn)	1.112	(0.028)	0.000	1.045	(0.062)	0.000	1.041	(0.048)	0.000	1.221	(0.050)	0.000	1.114	(0.069)	0.000
IK (i.e., difference between IK and LR)	0.237	(0.039)	0.000	0.299	(0.088)	0.001	0.301	(0.067)	0.000	0.140	(0.071)	0.050	0.228	(0.097)	0.015
CCTF (i.e., difference between CCTF and LR)	0.458	(0.039)	0.000	0.521	(0.088)	0.000	0.525	(0.067)	0.000	0.356	(0.071)	0.000	0.454	(0.097)	0.000
Number of Estimates	5,994			1,134			1,944			1,944			972		
Type of Simulated Effect	All			Flat			Growing Linearly			Growing Expontially			Fading		

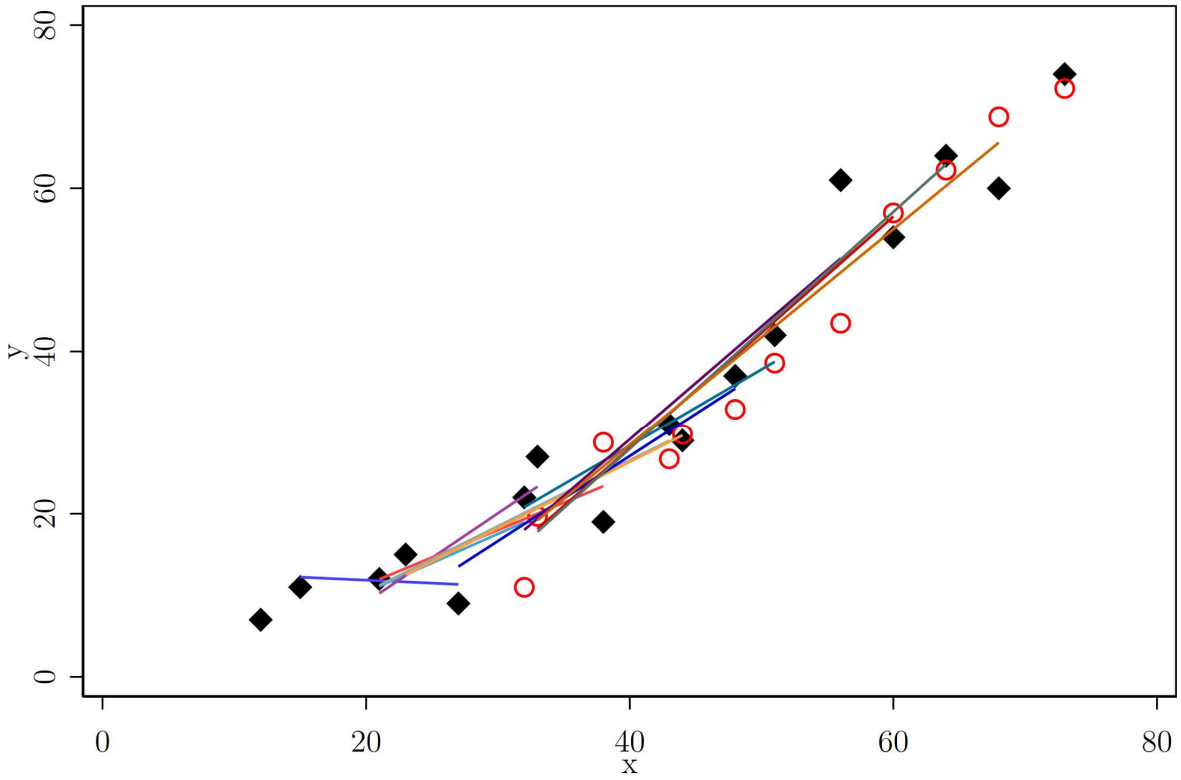
Panel B: Falsely Reject LATE

	Coef.	(s.e.)	p-value	Coef.	(s.e.)	p-value	Coef.	(s.e.)	p-value	Coef.	(s.e.)	p-value	Coef.	(s.e.)	p-value
Constant (i.e., mean for Long and Rooklyn)	0.043	(0.004)	0.000	0.043	(0.007)	0.000	0.035	(0.009)	0.000	0.044	(0.009)	0.000	0.049	(0.010)	0.000
IK (i.e., difference between IK and LR)	-0.001	(0.006)	0.871	0.000	(0.010)	1.000	0.005	(0.013)	0.719	-0.005	(0.013)	0.727	-0.005	(0.014)	0.736
CCTF (i.e., difference between CCTF and LR)	-0.008	(0.006)	0.224	-0.009	(0.010)	0.414	0.002	(0.013)	0.857	-0.009	(0.013)	0.485	-0.014	(0.014)	0.312
Number of Estimates	5,994			2,106			1,296			1,296			1,296		
Size of Simulated Effect	All			LATE=0			|LATE|=0.2sd			|LATE|=0.5sd			|LATE|=1.0sd		

Panel C: Fail to Reject Null when LATE≠0

	Coef.	(s.e.)	p-value	Coef.	(s.e.)	p-value	Coef.	(s.e.)	p-value	Coef.	(s.e.)	p-value
Constant (i.e., mean for Long and Rooklyn)	0.109	(0.011)	0.000	0.296	(0.023)	0.000	0.030	(0.012)	0.061	0.000		
IK (i.e., difference between IK and LR)	0.102	(0.015)	0.000	0.273	(0.033)	0.000	0.032	(0.017)	0.000	0.000		
CCTF (i.e., difference between CCTF and LR)	0.136	(0.015)	0.000	0.317	(0.033)	0.000	0.090	(0.017)	0.014	0.000		
Number of Estimates	3,888			1,296			1,296			1,296		
Size of Simulated Effect	LATE≠0			|LATE|=0.2sd			|LATE|=0.5sd			|LATE|=1.0sd		

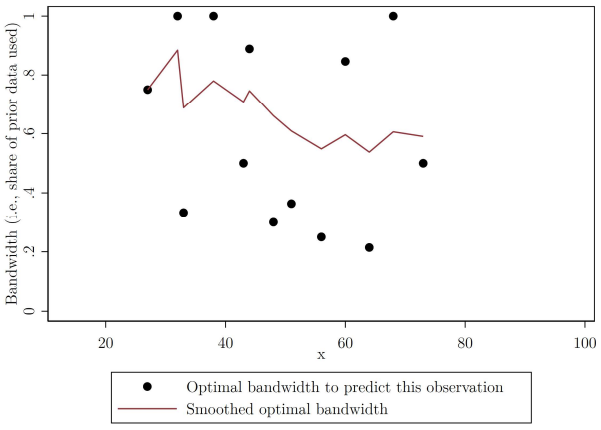
Figure 1:
Illustration of the Next Algorithm for the Left Side of a Threshold at x=75



Black diamonds = data
Red circles = out-of-sample predictions of the next value (from extension of the regression line)

Figure 2:
Illustration of the Smoothing of the Bandwidth and the Squared Prediction Error

Panel A: Bandwidth Smoothing



Panel B: Squared Prediction Error Smoothing

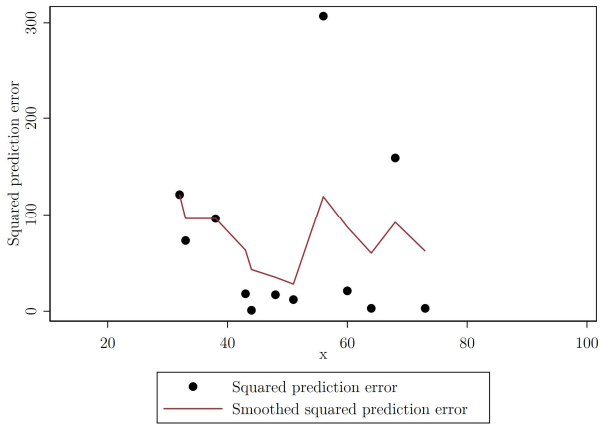
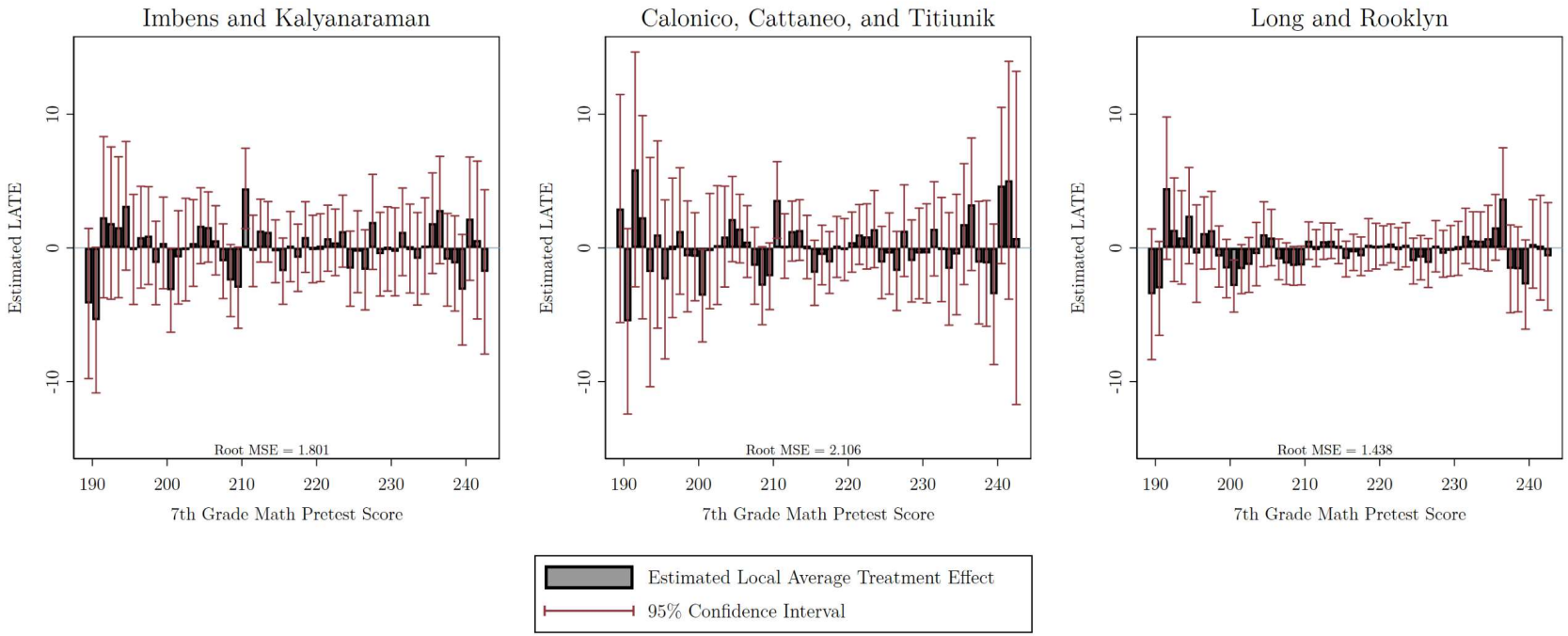


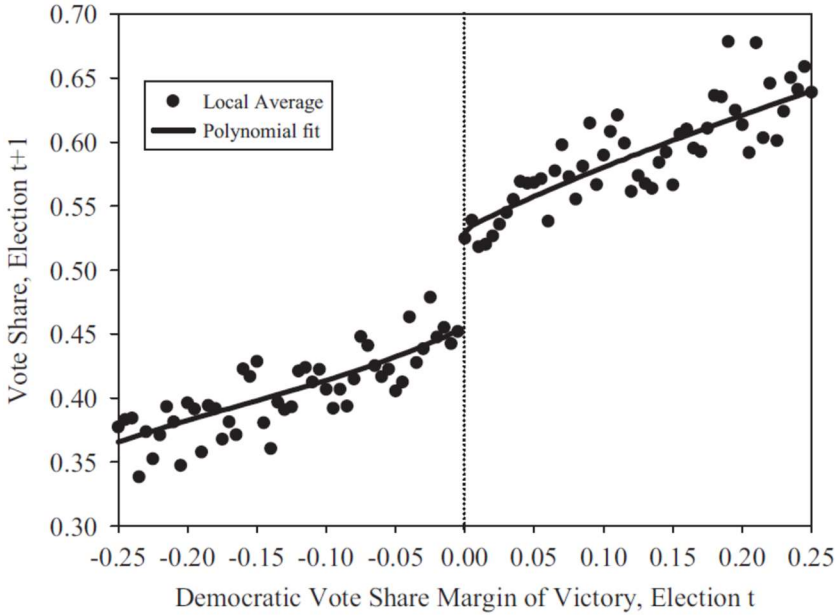
Figure 3:
Comparison of Methods Given a Simulated Treatment that had No Effect



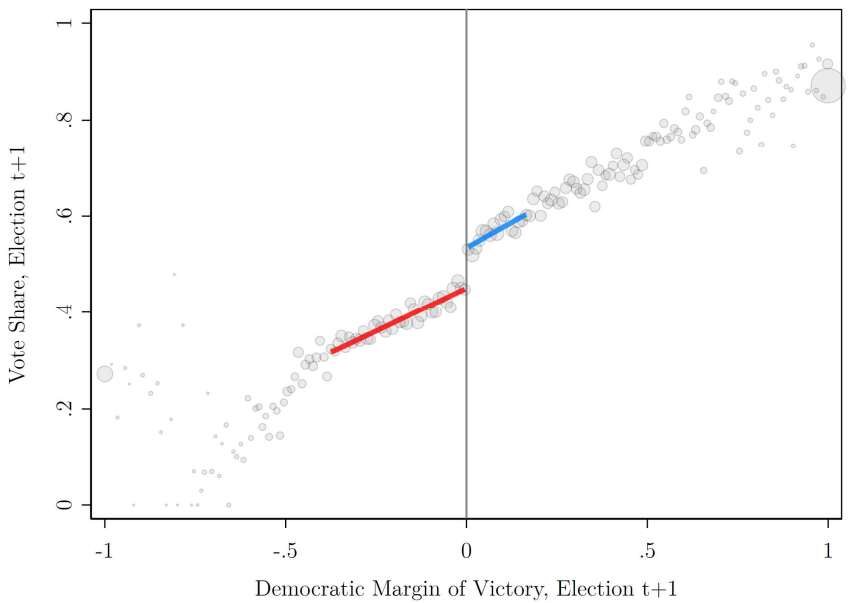
Note: The figure shows 54 simulated impact estimates (where the simulated impact is a flat 0, i.e., no effect). The threshold of the simulated treatment is moved from 189.5 to 242.5 and the treatment effect is to the right of the threshold. IK's method is estimated using Stata software and the *rd* command written by Nichols(2011). CCT's method is estimated using Stata software and the *rdrobust* command written by Calonico, Cattaneo, Farrell, and Titiunik (2018). LR's method is estimated using Stata software and the *next* command written by Long and Rooklyn (2020). Next's specification search was conducted across polynomial orders ranging from 1 to 3.

Figure 4:
Next Algorithm Applied to Figure 4a in Lee (2008)

Panel A: Reprint of Lee's Figure 4a



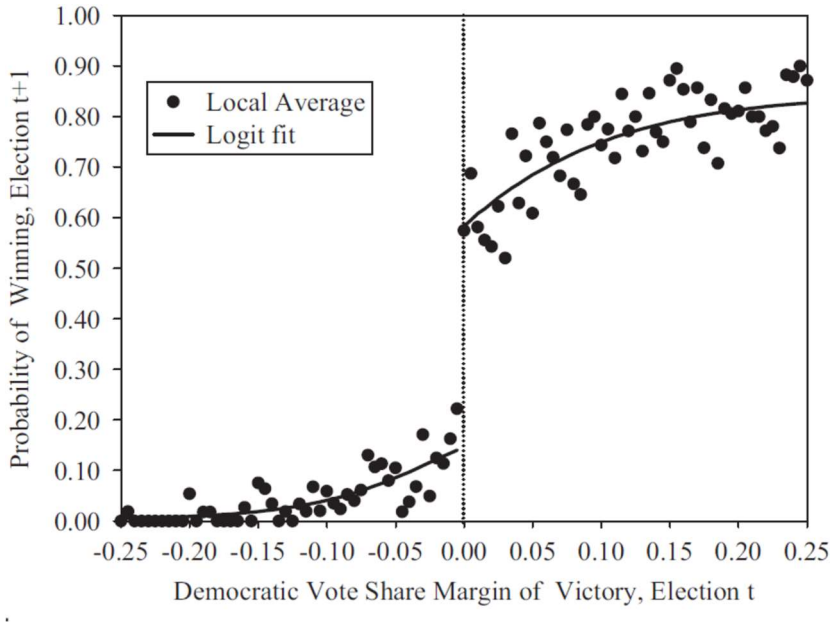
Panel B: Next Algorithm Results



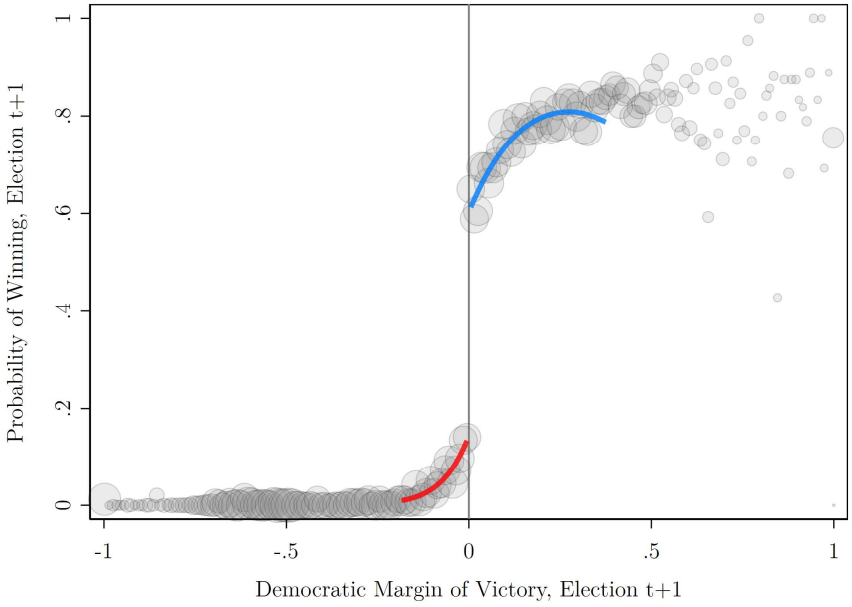
Note: Next's specification search was conducted across polynomial orders ranging from 0 to 6. Data were placed into 200 bins for the purpose of specification search and for the graph above. Data on the left (right) side are weighted by an Epanechnikov (uniform) kernel. The specification is linear in x on both sides.

Figure 5:
Next Algorithm Applied to Figure 2a in Lee (2008)

Panel A: Reprint of Lee's Figure 2a



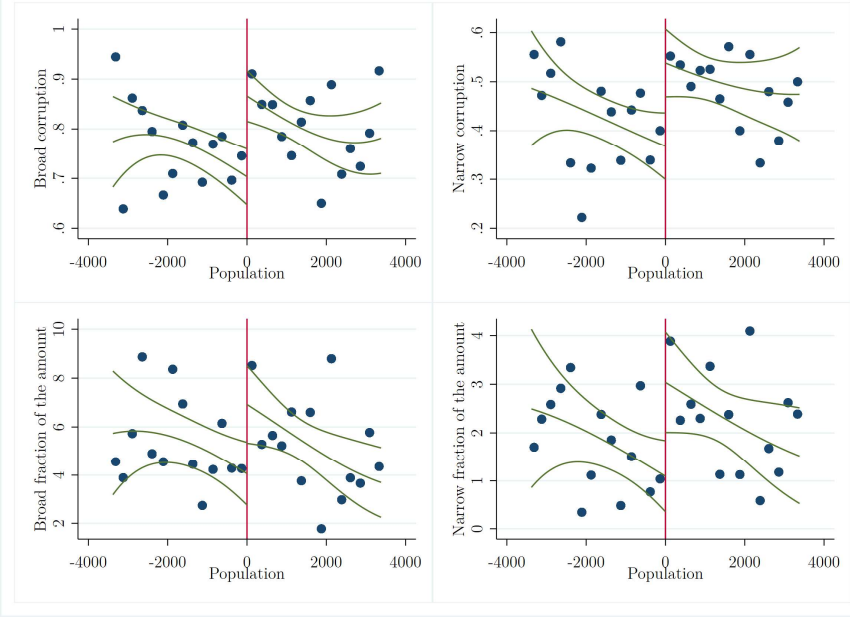
Panel B: Next Algorithm Results



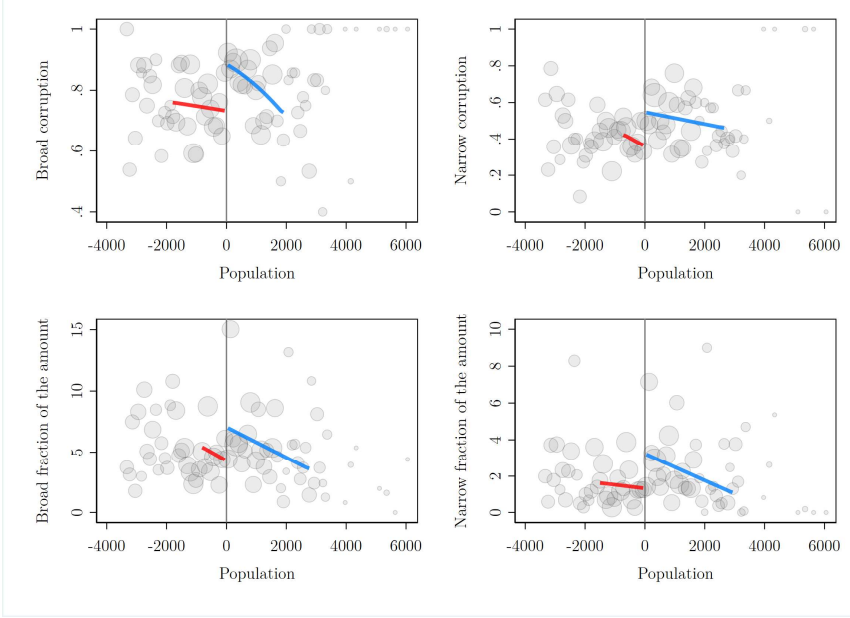
Note: Next's logit specification search was conducted across polynomial orders ranging from 1 to 4. Data were placed into 200 bins for the purpose of specification search and for the graph above. Data on both sides are weighted by an Epanechnikov kernel. The specification is linear (quadratic) in x on the left (right) side.

Figure 6:
Next Algorithm Applied to Estimates in Brollo et al. (2013)

Panel A: Replication of Brollo et al.'s Figure 2



Panel B: Next Algorithm Results



Note: Replication of Brollo et al.'s results were achieved by applying the code and data they provide online with their article. Next's specification search was conducted across polynomial orders ranging from 1 to 3. Logits were used for the first two outcomes, "Broad corruption" and "Narrow corruption", while linear specifications were used for the latter two outcomes. Data were placed into 200 bins for the purpose of specification search and for the graph above. Next identified that the optimal specification is linear in x for both sides of the threshold for all four outcomes. The following kernel weights were selected by the algorithm for the left (right) sides: Epanechnikov (uniform) for "Broad corruption"; uniform (uniform) for "Narrow corruption"; Epanechnikov (uniform) for "Broad fraction of the amount"; and triangular (Epanechnikov) for "Narrow fraction of the amount".

Online Appendices for

Next: Machine Learning for Regression Discontinuity

Next Algorithm Applied to Chen, Ebenstein, Greenstone, and Li (2013)

Our final case study is a replication of a prominent paper by Chen et al. (2013) that alarmingly concludes that "an arbitrary Chinese policy that greatly increases total suspended particulates (TSPs) air pollution is causing the 500 million residents of Northern China to lose more than 2.5 billion life years of life expectancy" (p. 12936). This policy established free coal to aid winter heating of homes north of the Huai River and Qinling Mountain range. Chen et al. used the distance from this boundary as the assignment variable with the treatment discontinuity being the border itself.

Chen et al. use a global cubic polynomial specification with a treatment jump at the discontinuity for each outcome and their results are shown in Panels A and B of Figure 7. They estimate that being north of the boundary significantly raises TSP by 248 points and significantly lowers life expectancy by 5.04 years. Yet, Gelman and Zelizer (2015) critique these findings by noting, "(t)he large, statistically significant estimated treatment effect at the discontinuity depends on the functional form employed. …the headline claim, and its statistical significance, is highly dependent on a model choice that may have a data-analytic purpose, but which has no particular scientific basis" (pp. 3-4).

We have attempted to replicate these results with the results shown in panels A and B. Unfortunately, the primary data are proprietary and not easy to obtain; permission for their use can only be granted by the Chinese Center for Disease Control.[1] Rather than use the underlying primary data, we are treating the data shown in their Figures 2 and 3 as if it were the actual data. To do so, we have manually measured the x and y coordinates of each data point in these figures as well as the diameter of each circle (where the circle's area is proportional to the population of localities represented in the bin).[2] Our replication attempt is shown in Panels C and D of Figure 7. We obtain similar results, although the magnitudes are smaller and less significant; our replication of their specification produces estimates that being north of the boundary raises TSP by 178 points (p-value 0.069) and insignificantly lowers life expectancy by 3.94 years (p-value 0.389).

---

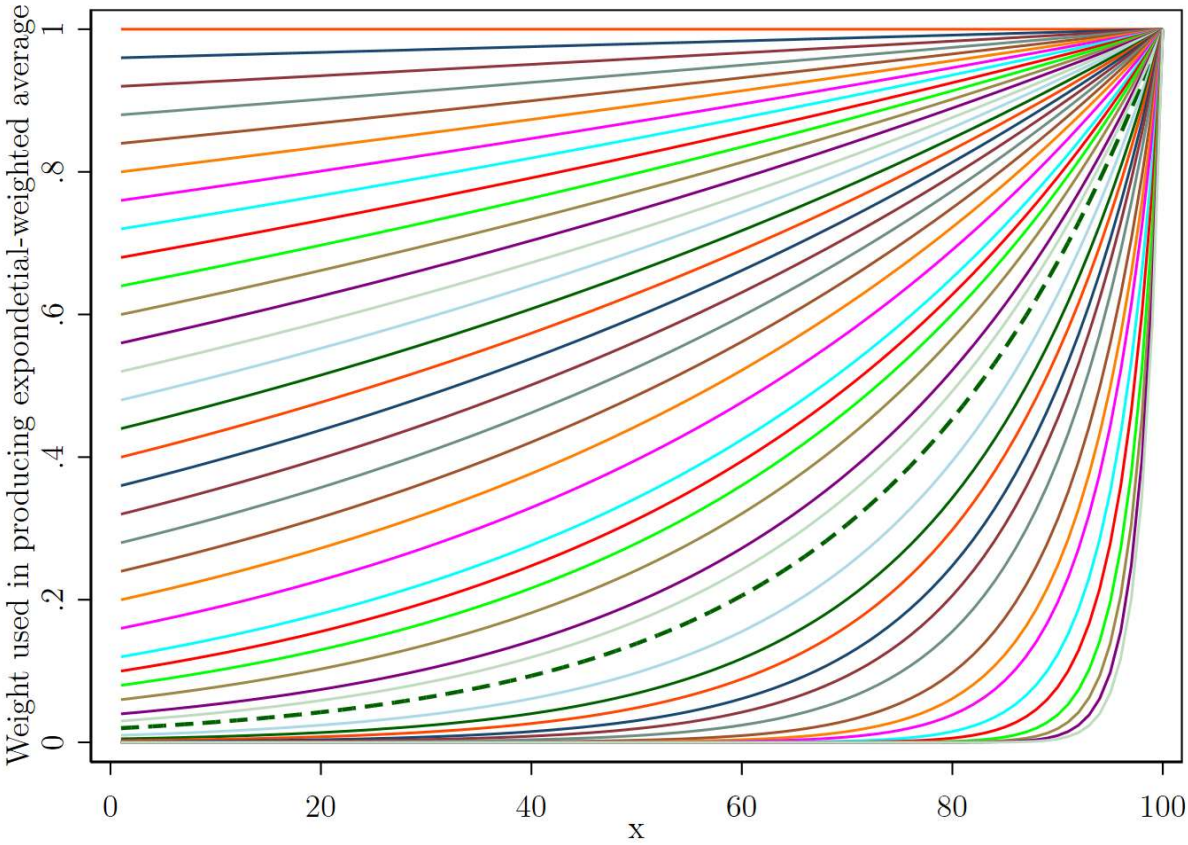
[1] Personal communication with Michael Greenstone, March 16, 2015.

[2] We have taken two separate measurements for each figure and use the average of these two measurements for the x and y coordinates and the median of four measurements of the diameter of each circle.

We apply our Next method to estimate these treatment effects, with results graphically shown in panels E and F of Figure 7. Frankly, our method is not well suited to these data as there are few distinct values of x on each side of the discontinuity with which to learn the data-generating process. Our specifications and treatment estimates are radically different from Chen et al. For TSP, the Next algorithm suggests a linear specification with triangular kernel weights on both sides, using data from the latter 68% of the span of the data on the left (8 observations), and the data in the first 28% of the span of the data on the right (5 observations). Based on this specification, we estimate that the treatment caused TSP to be reduced by 54 (s.e.=3), which is far below the point estimate of our replication, 178. For Life Expectancy, the Next algorithm suggests a quadratic (cubic) specification for the left (right) side, with uniform kernel weights on both sides, using all of the data on the left (12 observations), and the data in the first 60% of the span of the data on the right (10 observations). Based on this specification, we estimate that the treatment caused Life Expectancy to be *reduced* by 8.6 years (s.e.=1.2), which is a stunning and opposite conclusion from Chen et al. To be clear, given the sparseness of the data, we do not believe that our Next algorithm is appropriate for this analysis and we do not believe that our results should be taken as revealing good causal estimates.

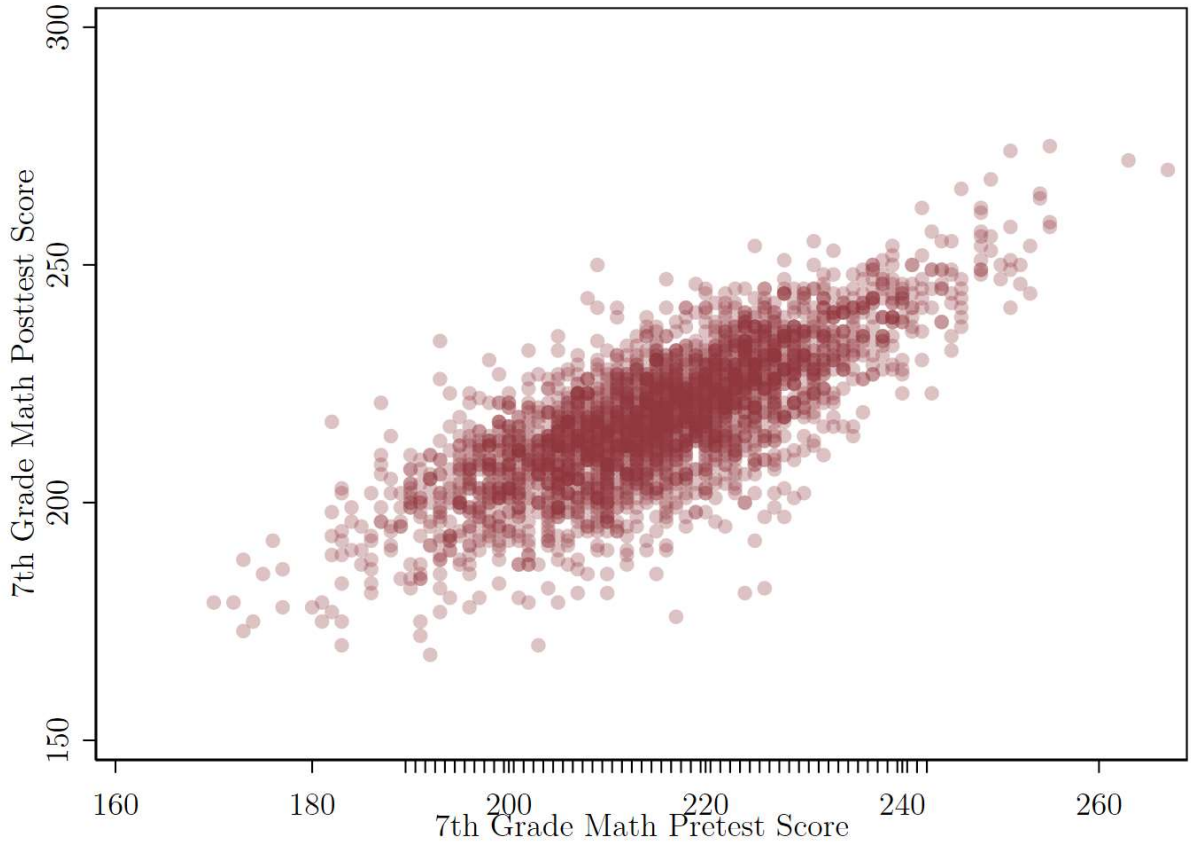
The fragility of the Chen et al. results should not be surprising given a visual inspection of the scatterplot, which does not reveal a clear pattern to the naked eye. We agree with Gelman and Zelizer's (2015) critique that the result "indicates to us that neither the linear nor the cubic nor any other polynomial model is appropriate here. Instead, there are other variables not included in the model which distinguish the circles in the graph" (p. 4).

Appendix Figure 1:
Illustration of the Comprehensiveness of the Series of Weights Produced Using Various Tested Candidate Values of α_{BW} (and the Set Value of α_{SPE})



Note: The 44 lines shown in this figure correspond to 44 different levels of α_{BW} that are tested, including 1, 0.96, 0.92, …, 0.12, 0.10, …, 0.02, 0.01, 0.005, 0.0025, 0.001, 0.0004, 10^{-04}, 10^{-05}, 10^{-06}, 10^{-7}, 10^{-9}, 10^{-11}, 10^{-14}, 10^{-17}, 10^{-20}, and 10^{-23}. The dashed line corresponds to the value of α_{SPE} that is set as a default (i.e., 0.02).

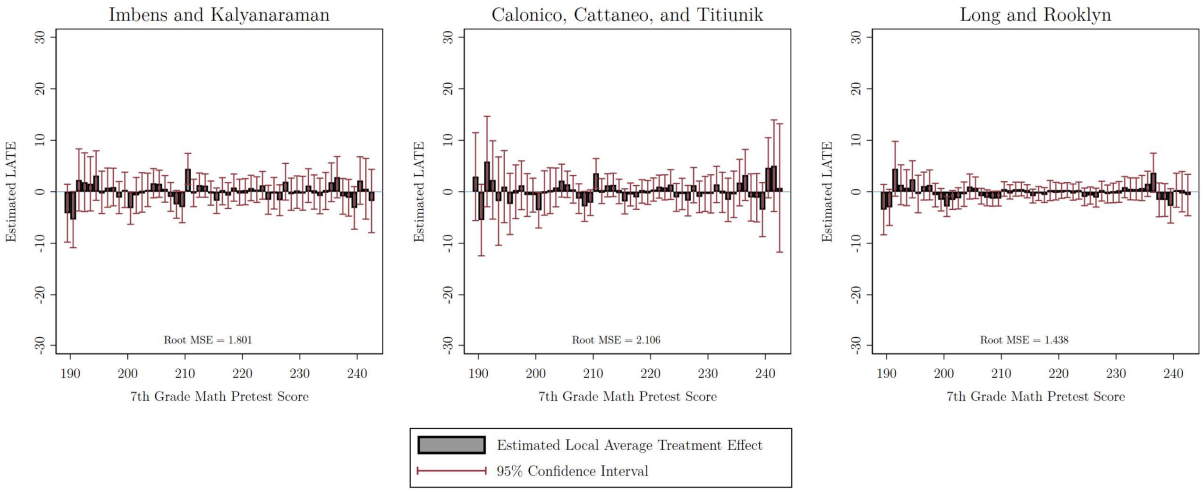
Appendix Figure 2:
Raw Data from Jacob et al. (2012) Used in Evaluating Performance of Next Algorithm for Estimating Simulated Treatment Effects



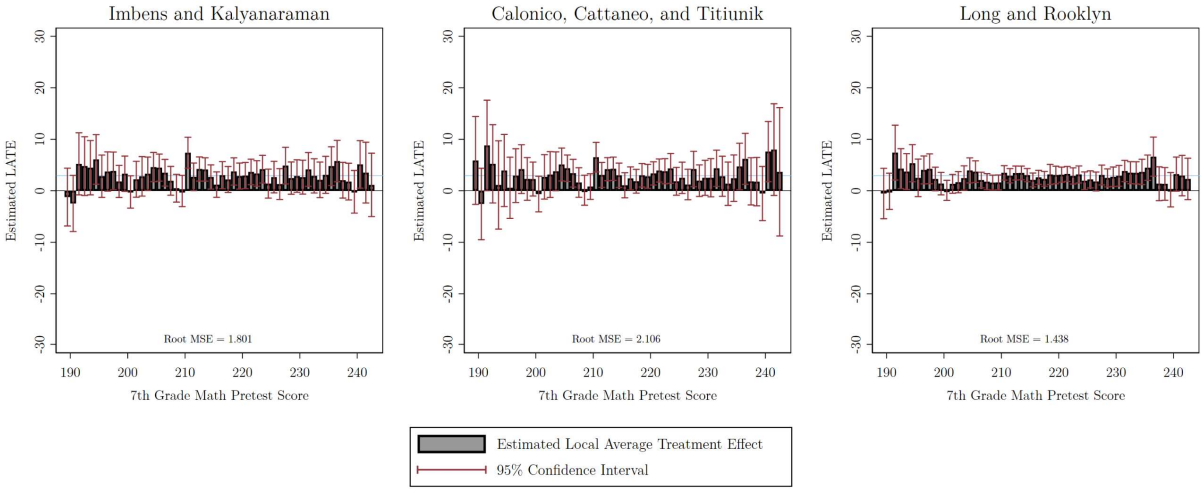
Note: Tick marks along the x-axis show the locations of thresholds for which simulated treatment effects are applied to students on the right of the threshold.

Appendix Figure 3:
Comparison of Methods Given Simulated Treatments of Various Sizes and Shapes

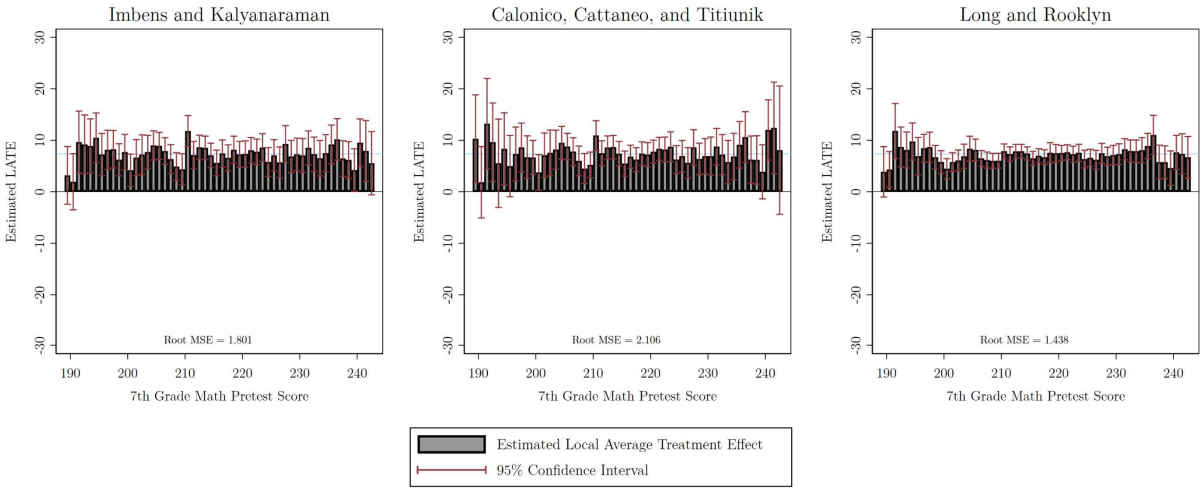
Panel A: Treatment Effect = 0



Panel B: Treatment Effect = 0.2σ

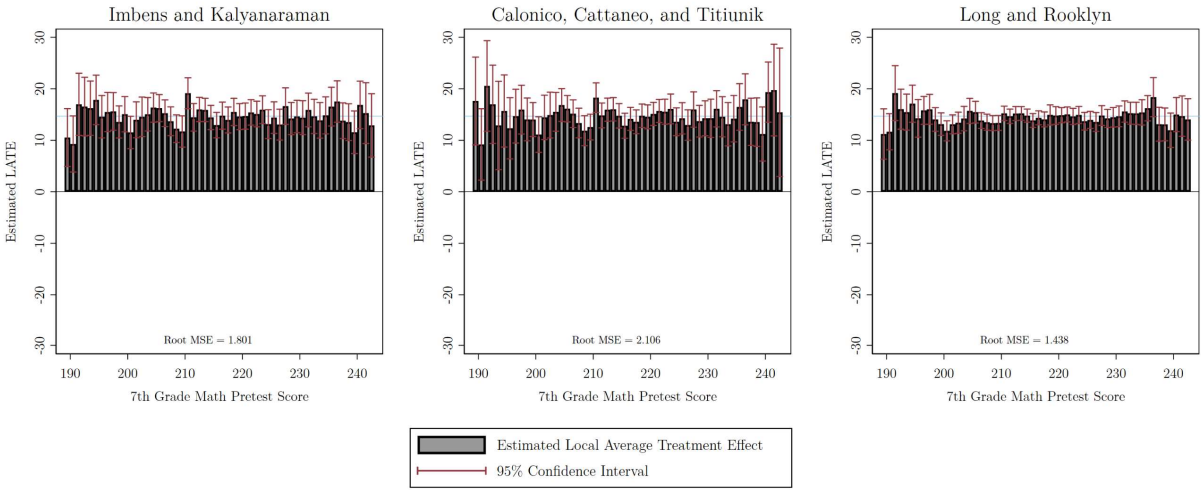


Panel C: Treatment Effect = 0.5σ

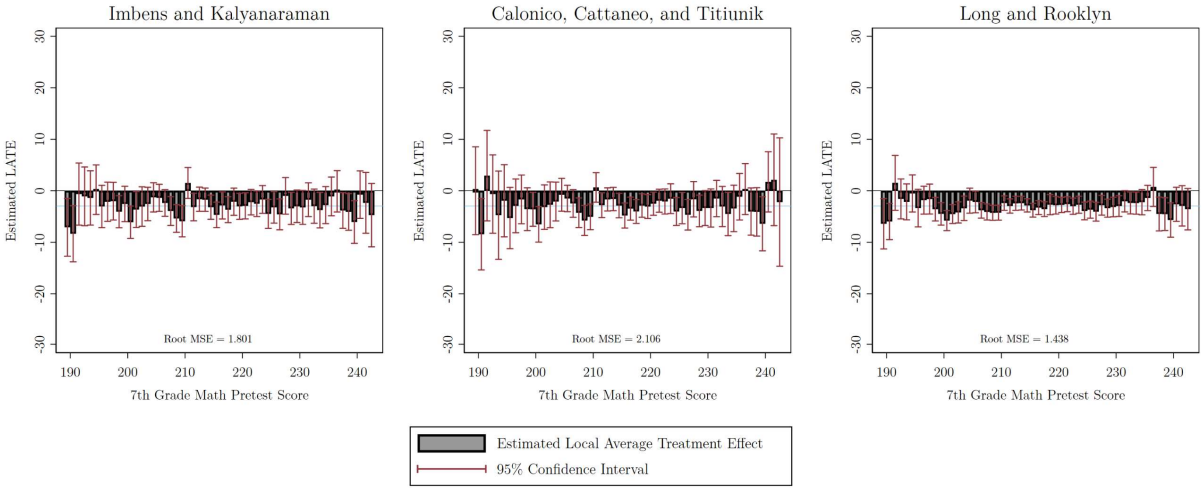


Appendix Figure 3: Continued

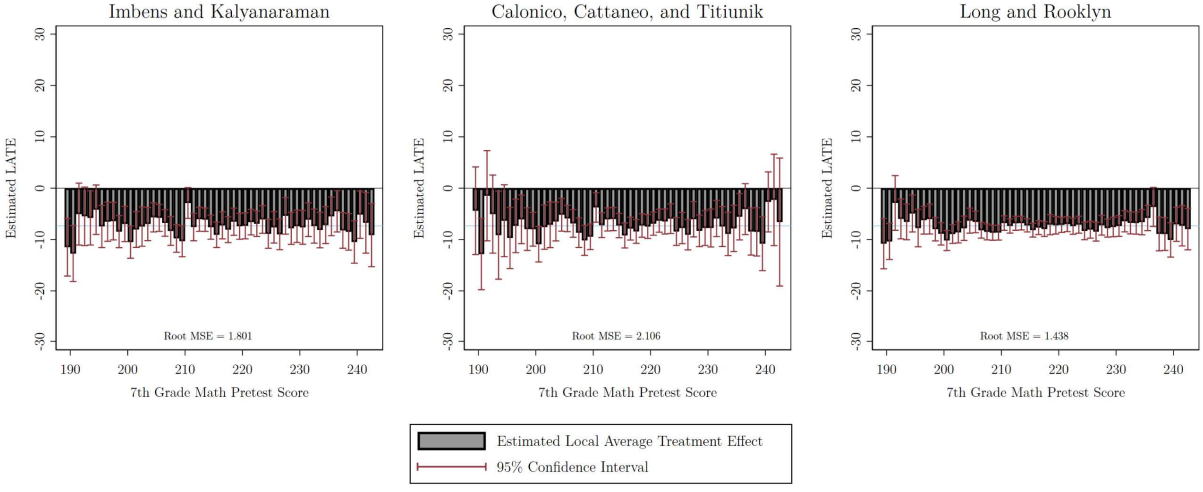
Panel D: Treatment Effect = σ



Panel E: Treatment Effect = -0.2σ

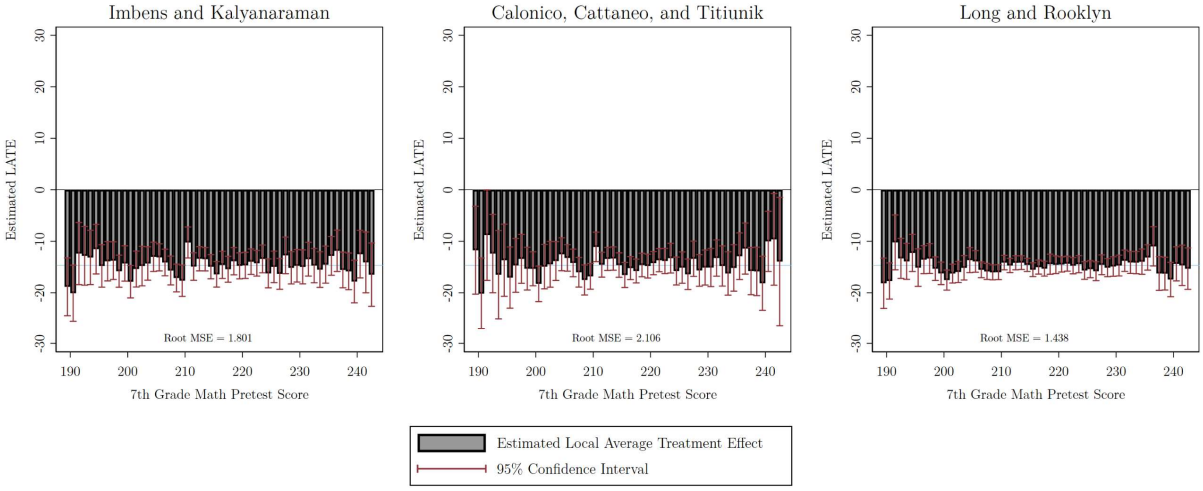


Panel F: Treatment Effect = -0.5σ

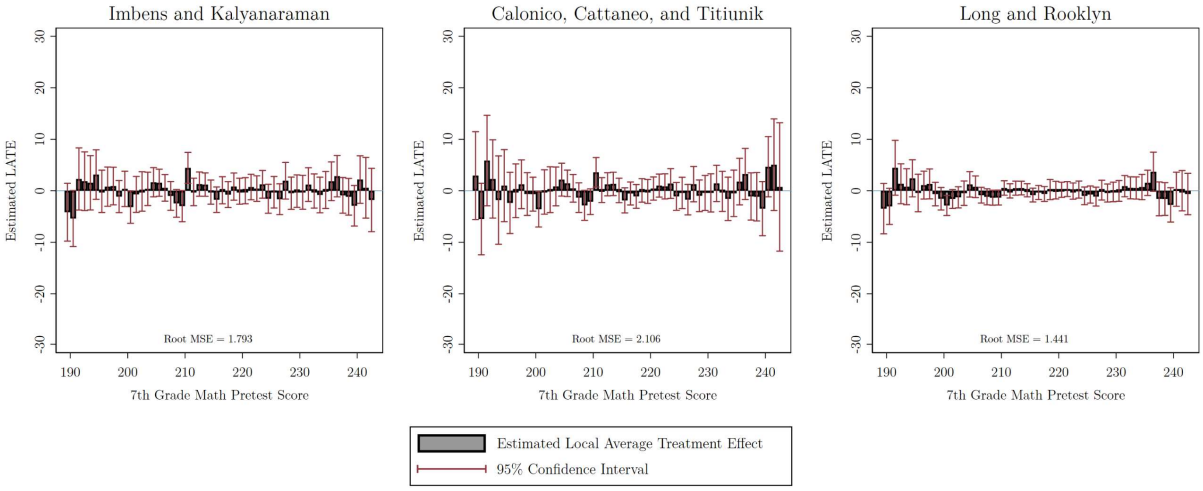


Appendix Figure 3: Continued

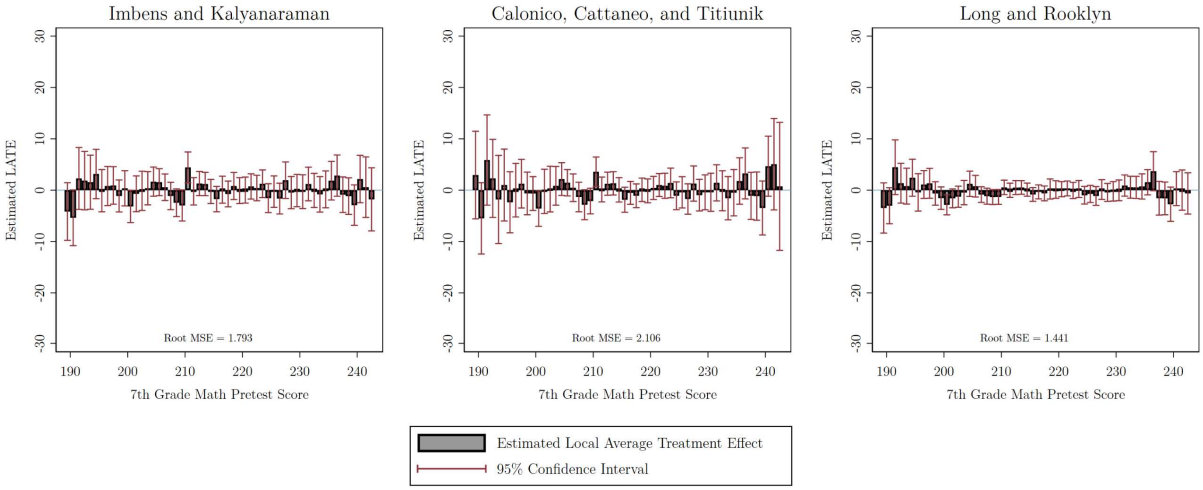
Panel G: Treatment Effect = $-\sigma$



Panel H: Treatment Effect = $0.2\sigma\theta$

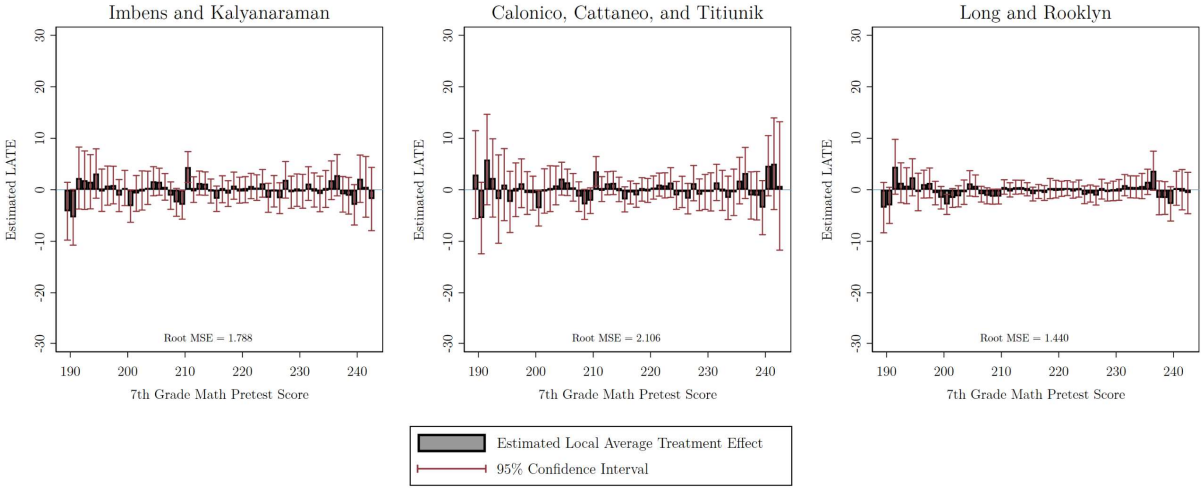


Panel I: Treatment Effect = $0.5\sigma\theta$

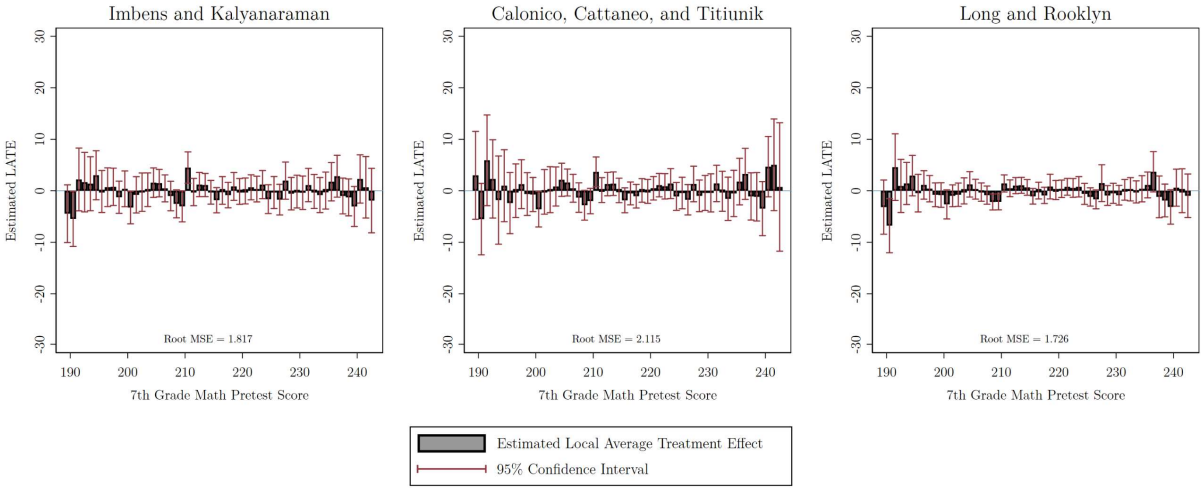


Appendix Figure 3: Continued

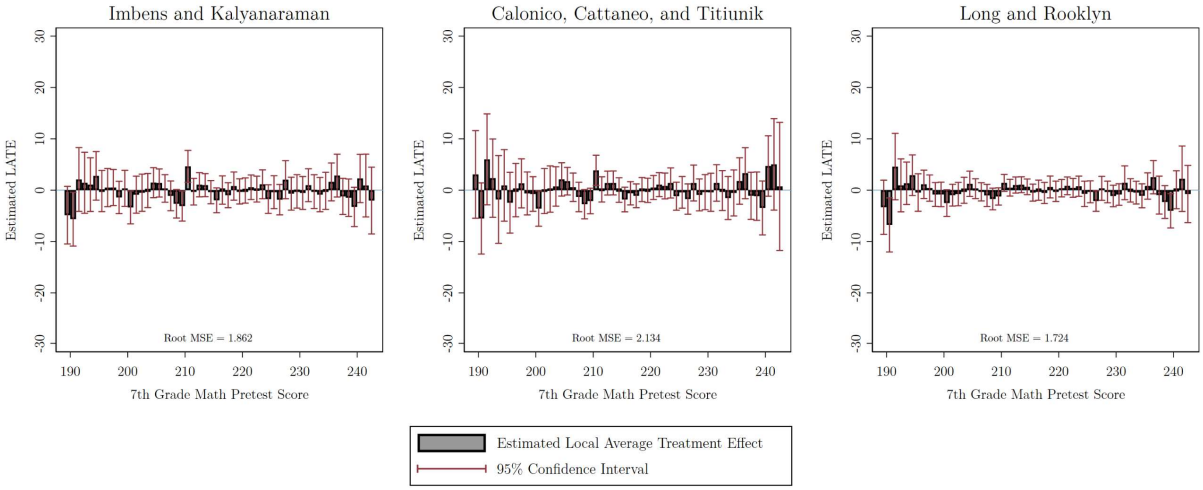
Panel J: Treatment Effect $= \sigma\theta$



Panel K: Treatment Effect $= 0.2\sigma 2^{\theta}$

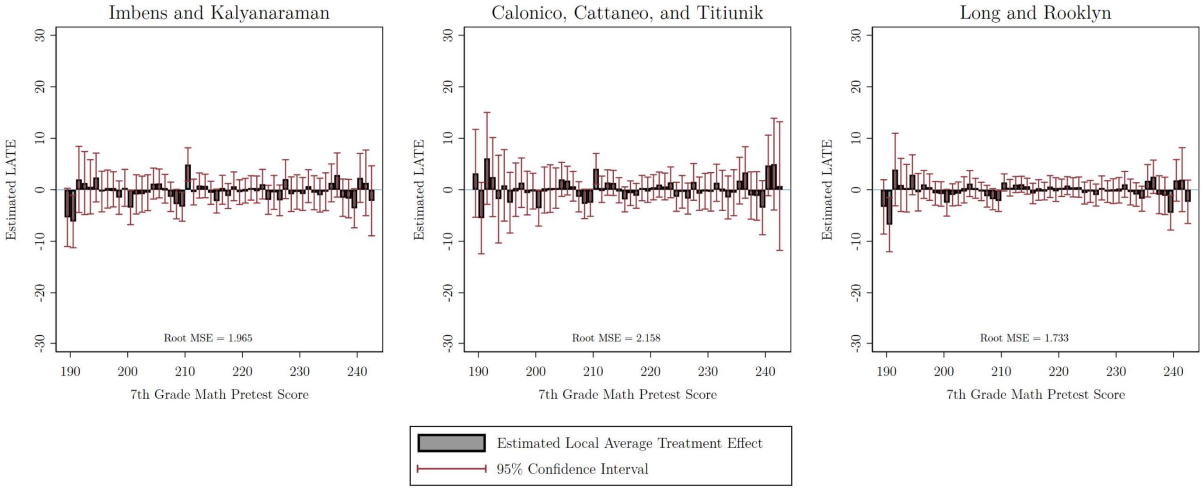


Panel L: Treatment Effect $= 0.5\sigma 2^{\theta}$

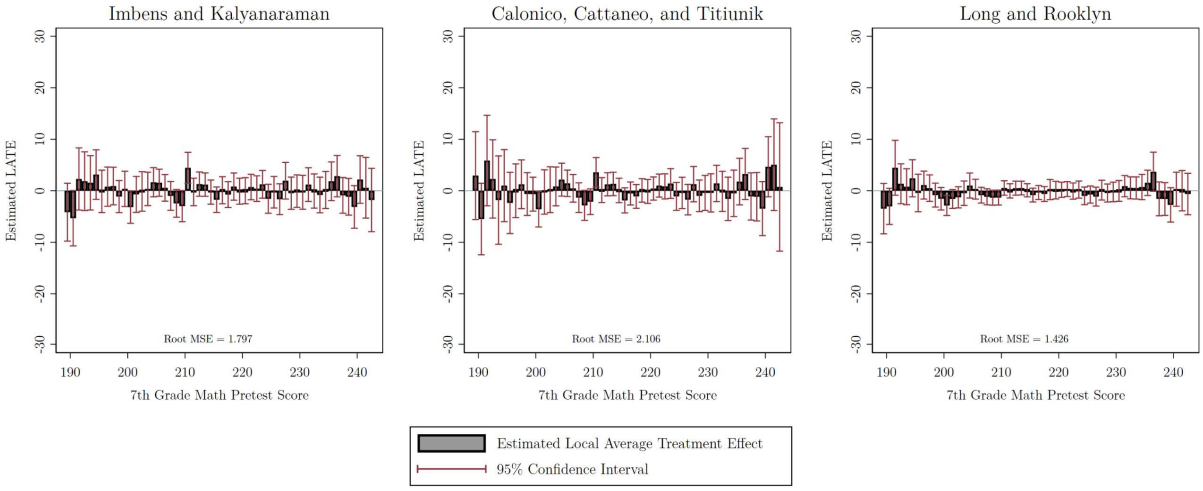


Appendix Figure 3: Continued

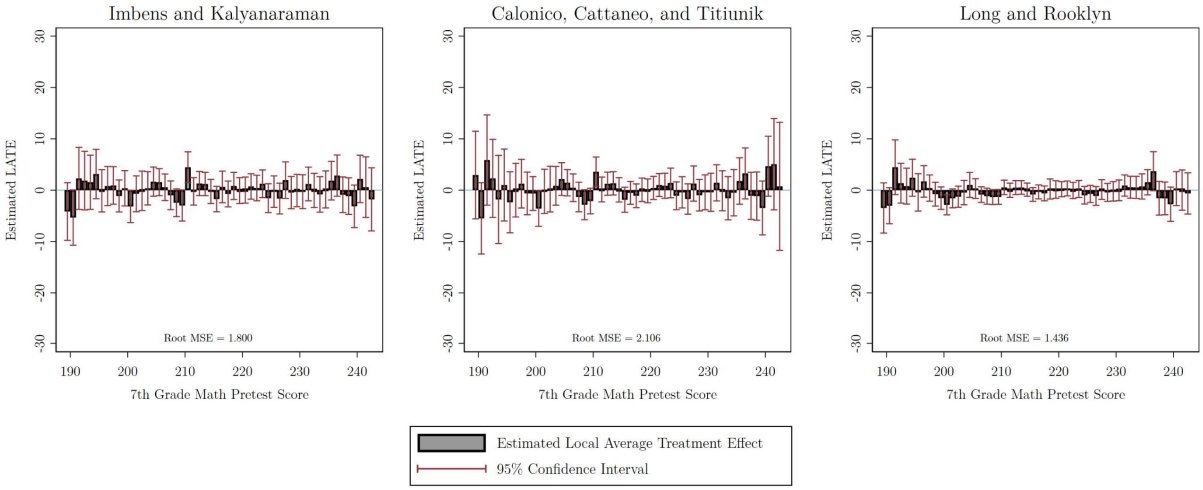
Panel M: Treatment Effect = $\sigma 2^{\theta}$



Panel N: Treatment Effect = $-0.2\sigma\theta$

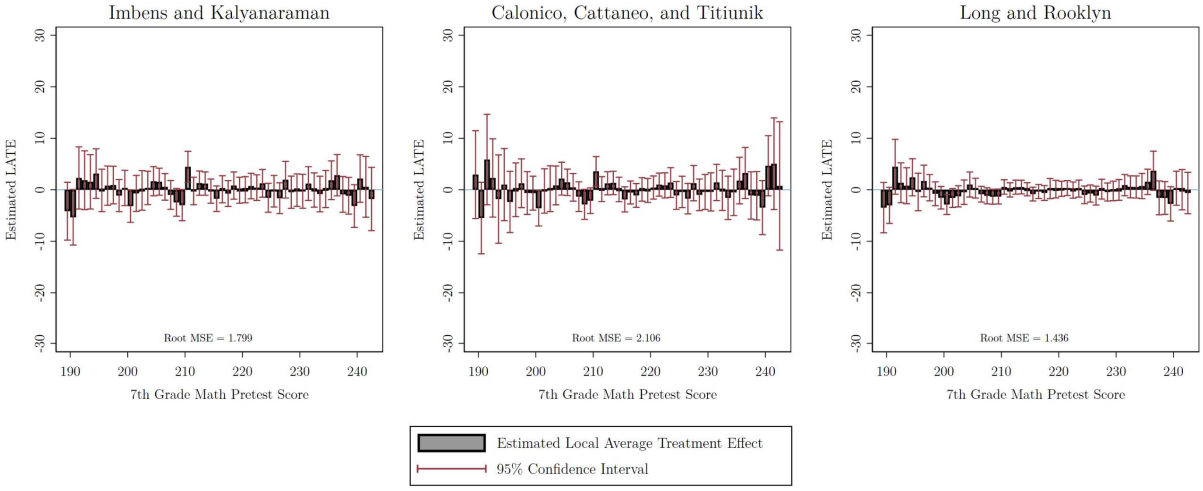


Panel O: Treatment Effect = $-0.5\sigma\theta$

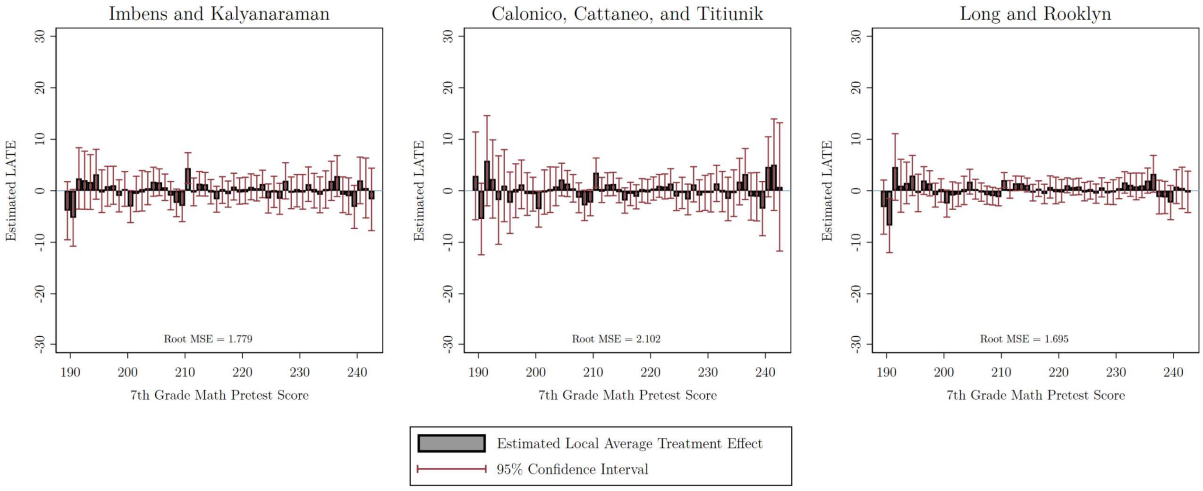


Appendix Figure 3: Continued

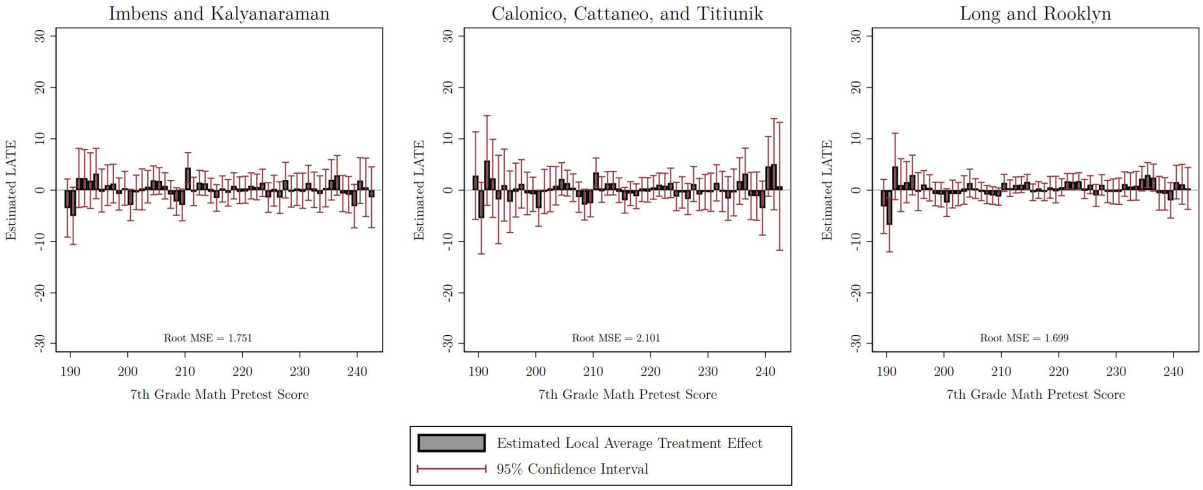
Panel P: Treatment Effect $= -\sigma\theta$



Panel Q: Treatment Effect $= -0.2\sigma 2^{\theta}$

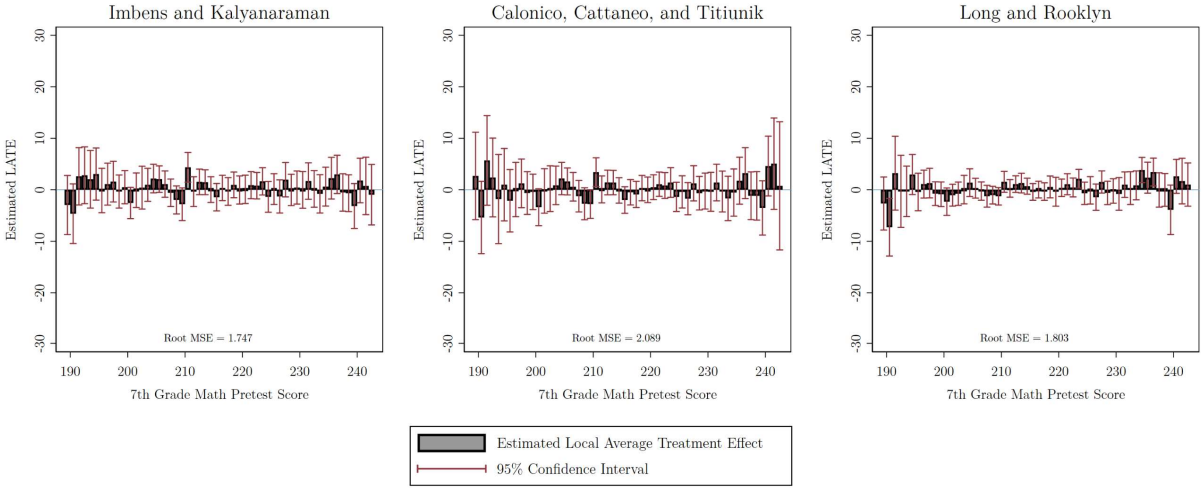


Panel R: Treatment Effect $= -0.5\sigma 2^{\theta}$

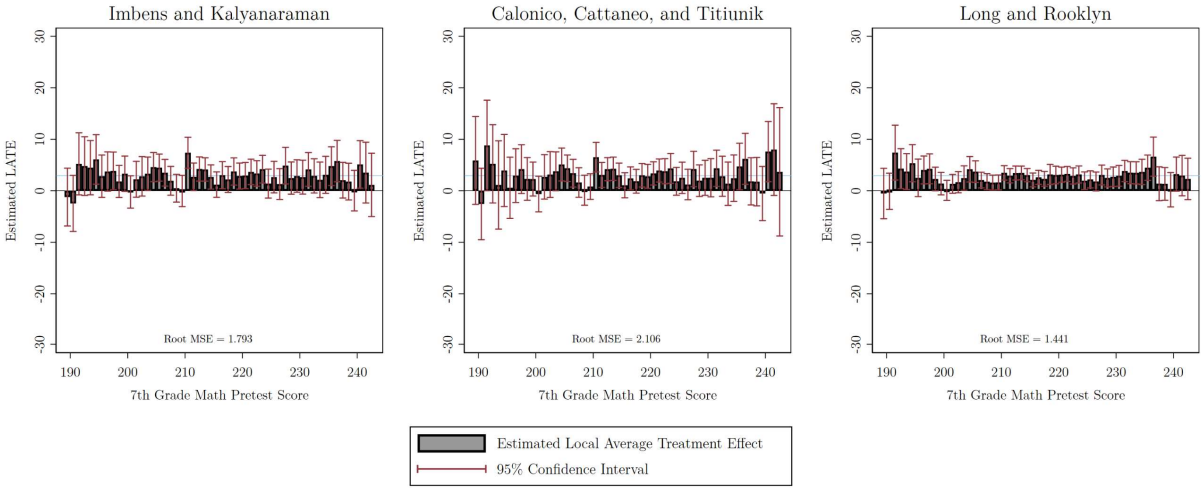


Appendix Figure 3: Continued

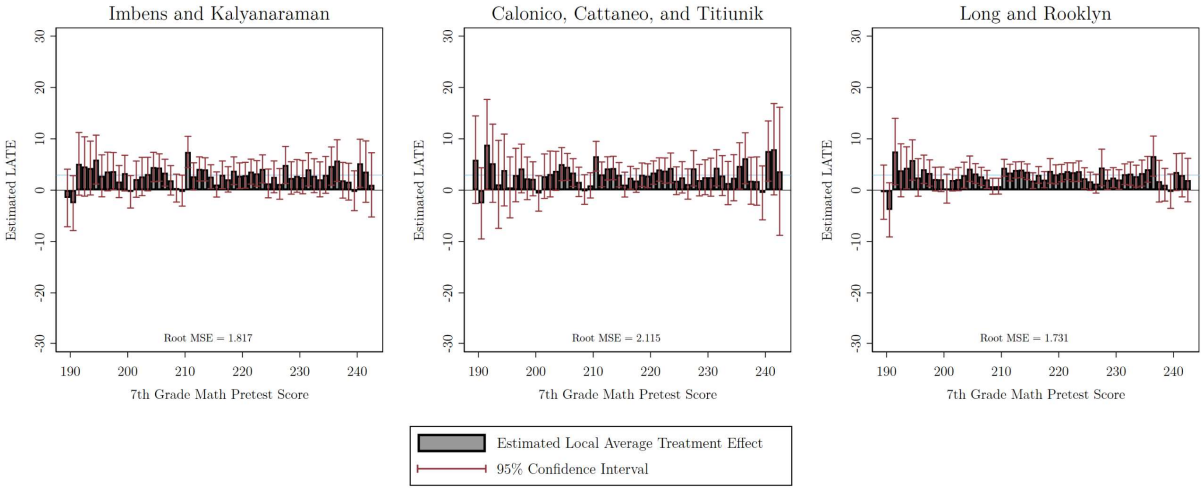
Panel S: Treatment Effect $= -\sigma 2^{\theta}$



Panel T: Treatment Effect $= 0.2\sigma + 0.2\sigma\theta$

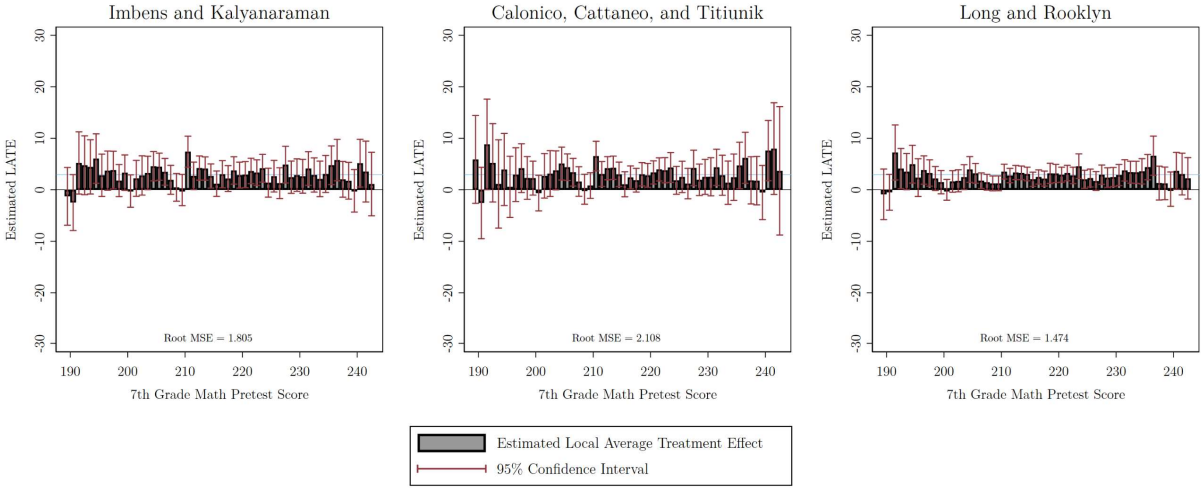


Panel U: Treatment Effect $= 0.2\sigma + 0.2\sigma 2^{\theta}$

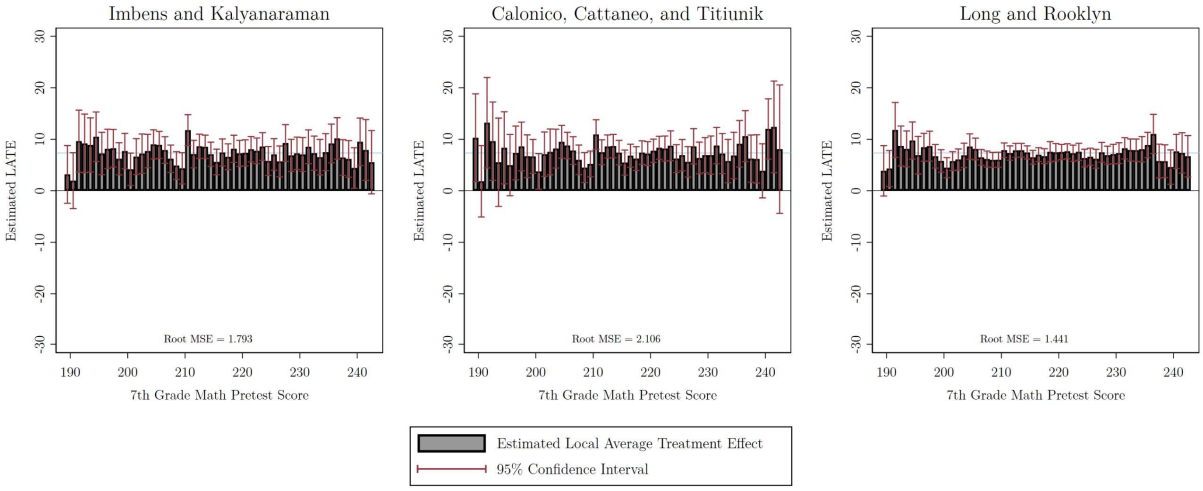


Appendix Figure 3: Continued

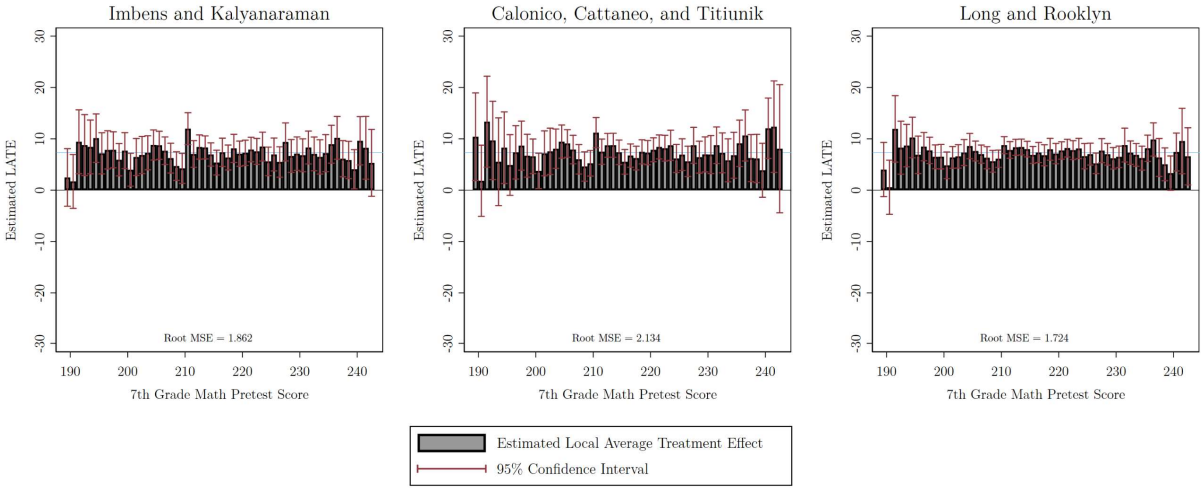
Panel V: Treatment Effect $= 0.2\sigma/2^{\theta}$



Panel W: Treatment Effect $= 0.5\sigma + 0.5\sigma\theta$

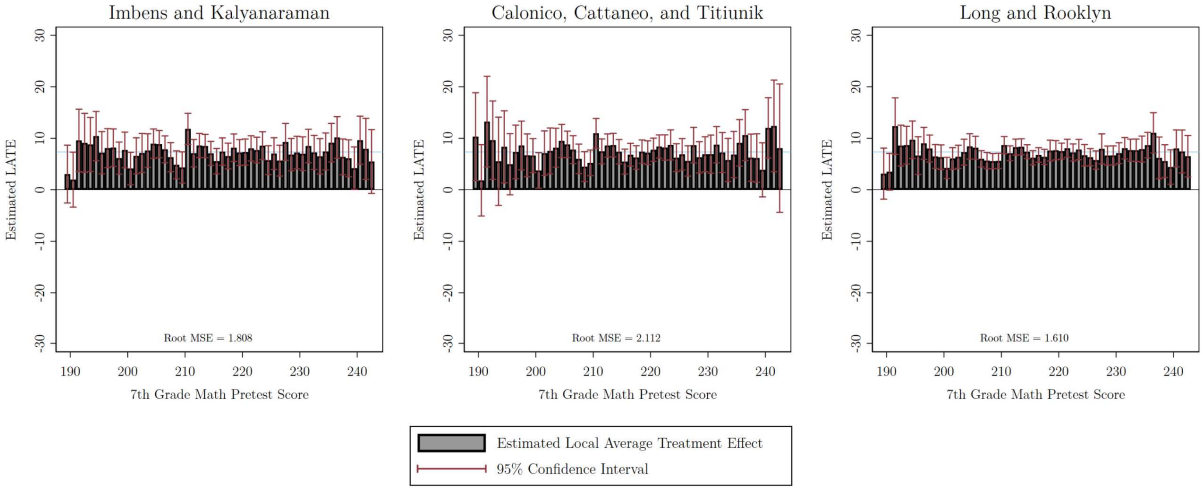


Panel X: Treatment Effect $= 0.5\sigma + 0.5\sigma 2^{\theta}$

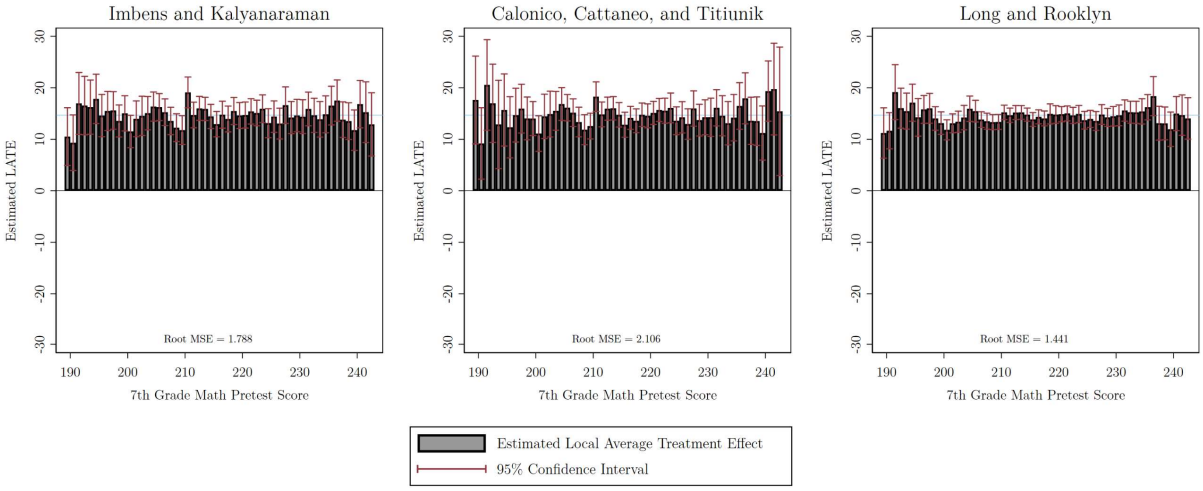


Appendix Figure 3: Continued

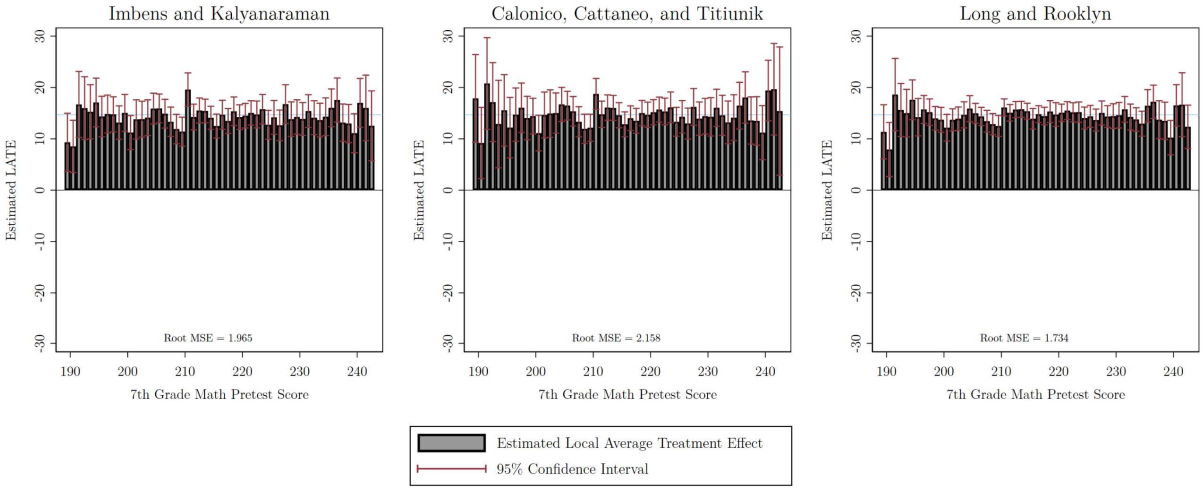
Panel Y: Treatment Effect $= 0.5\sigma/2^{\theta}$



Panel Z: Treatment Effect $= \sigma + \sigma\theta$

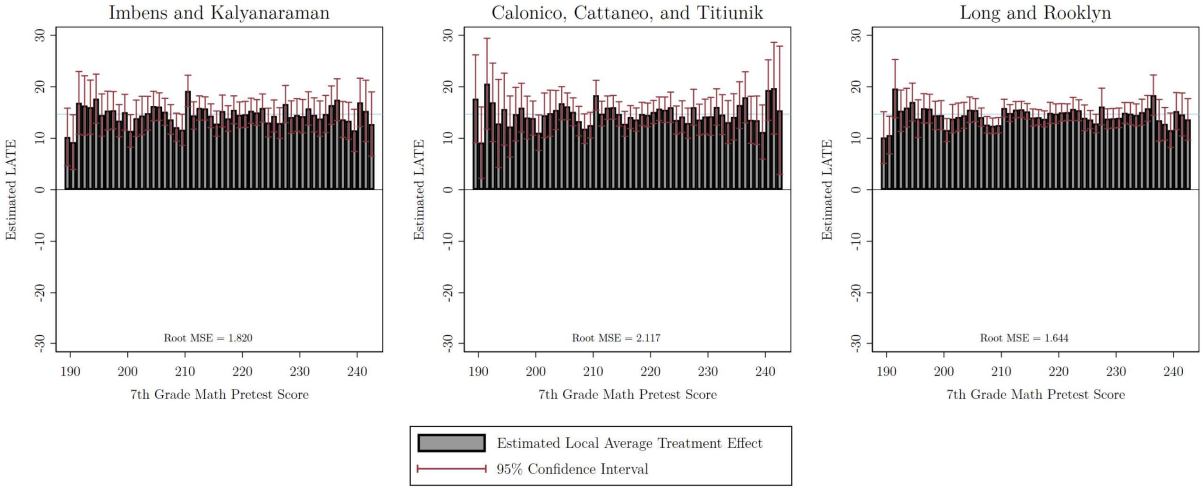


Panel AA: Treatment Effect $= \sigma + \sigma 2^{\theta}$

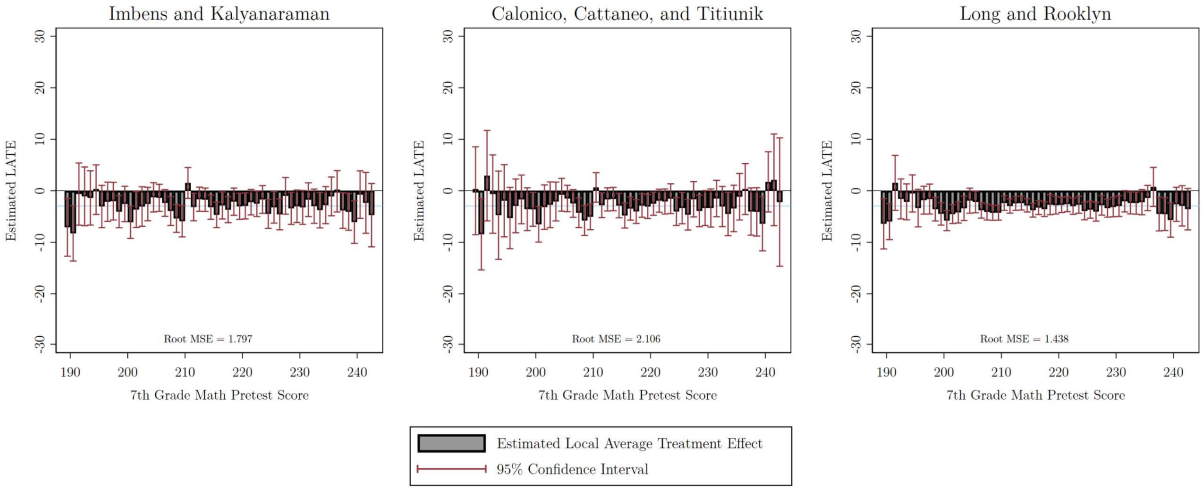


Appendix Figure 3: Continued

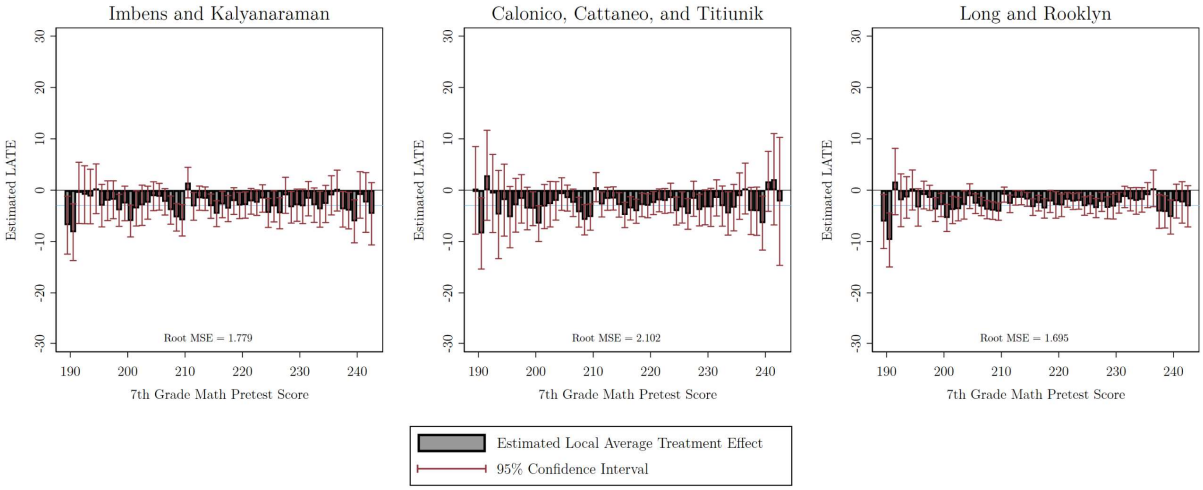
Panel AB: Treatment Effect $= \sigma/2^{\theta}$



Panel AC: Treatment Effect $= -0.2\sigma - 0.2\sigma\theta$

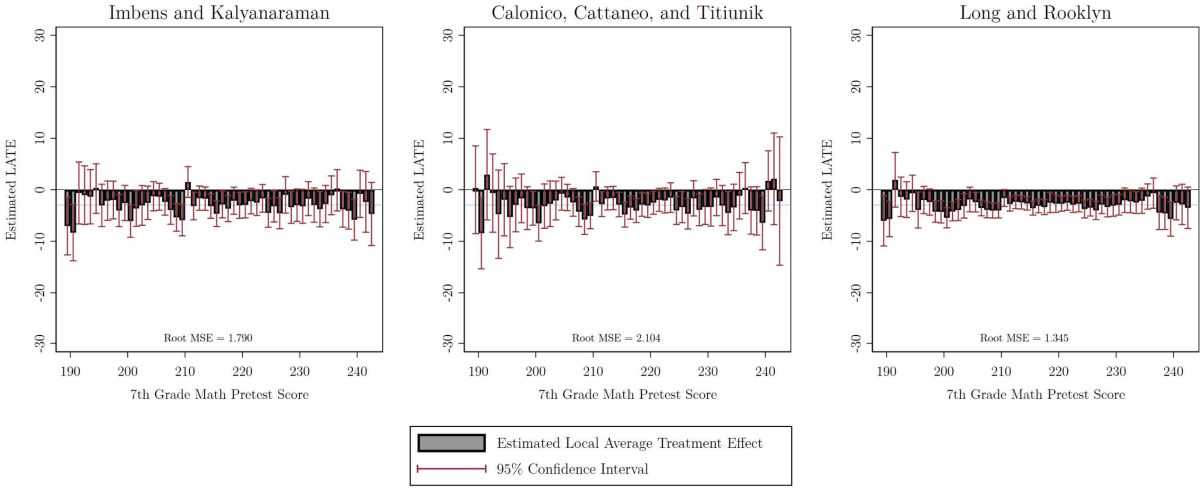


Panel AD: Treatment Effect $= -0.2\sigma - 0.2\sigma 2^{\theta}$

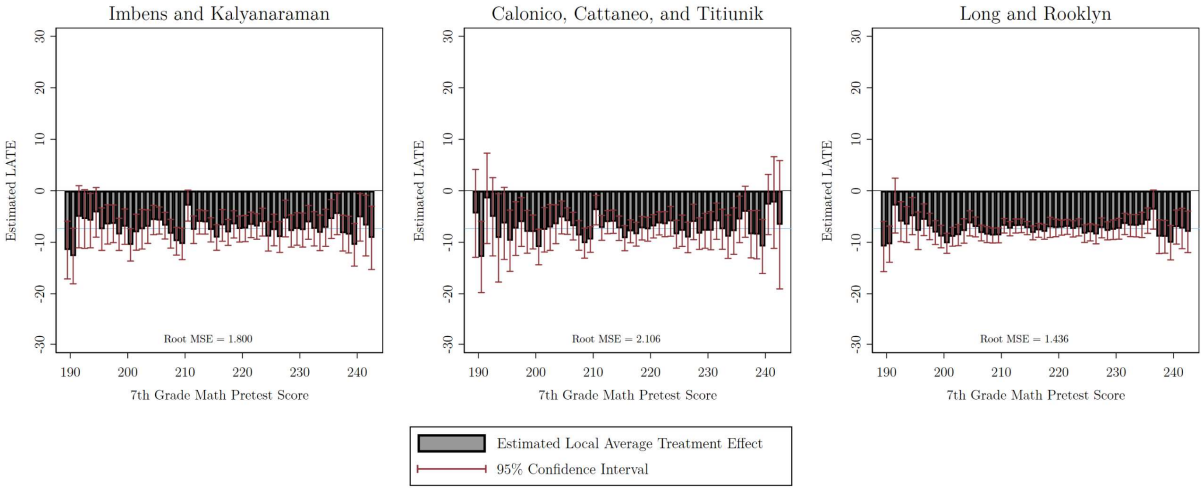


Appendix Figure 3: Continued

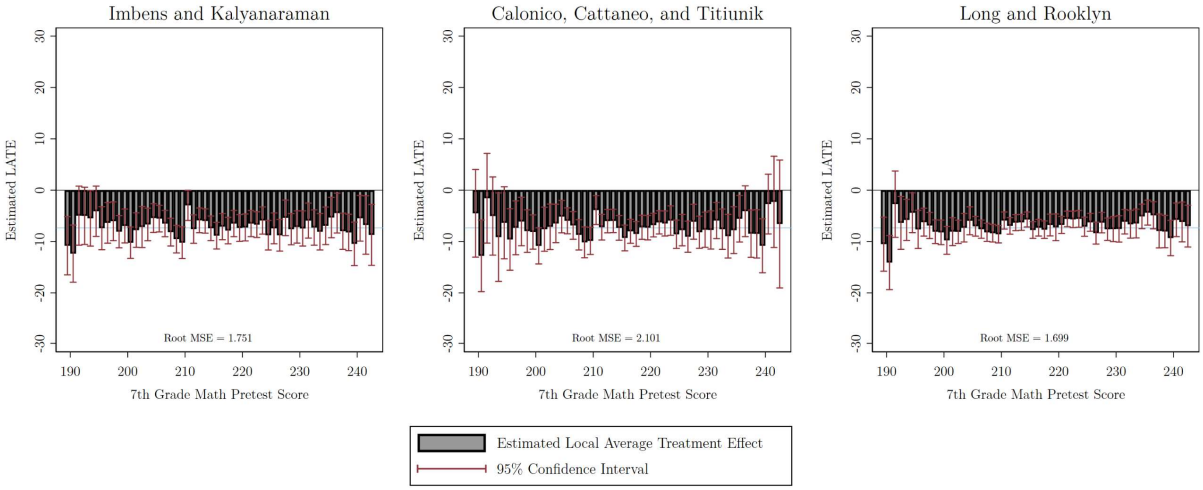
Panel AE: Treatment Effect $= -0.2\sigma/2^{\theta}$



Panel AF: Treatment Effect $= -0.5\sigma - 0.5\sigma\theta$

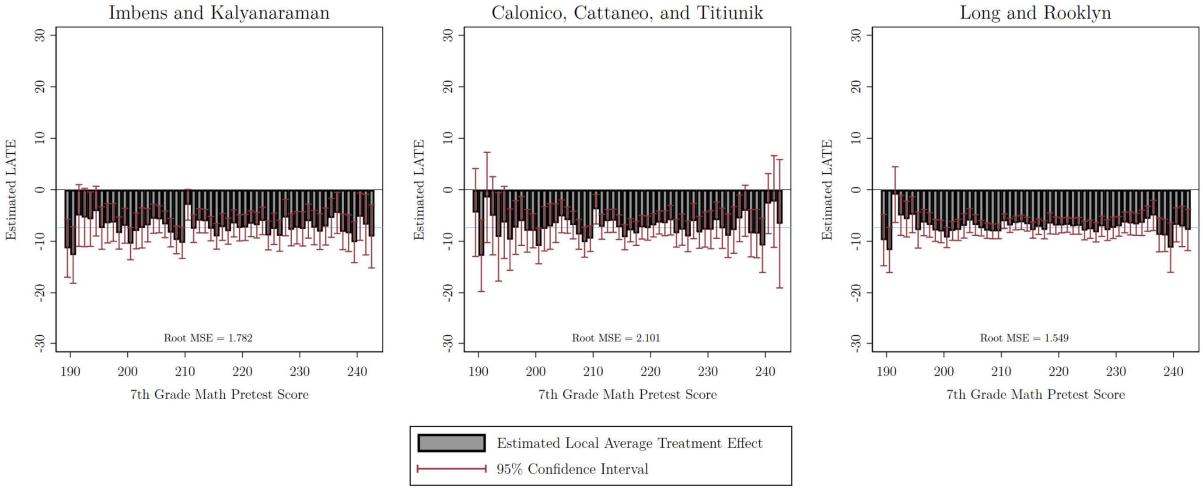


Panel AG: Treatment Effect $= -0.5\sigma - 0.5\sigma 2^{\theta}$

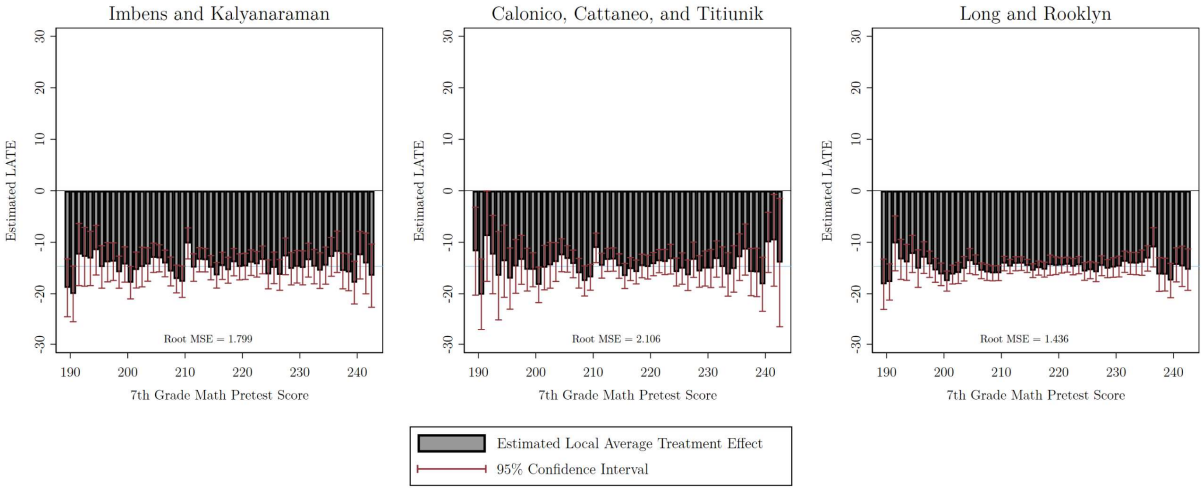


Appendix Figure 3: Continued

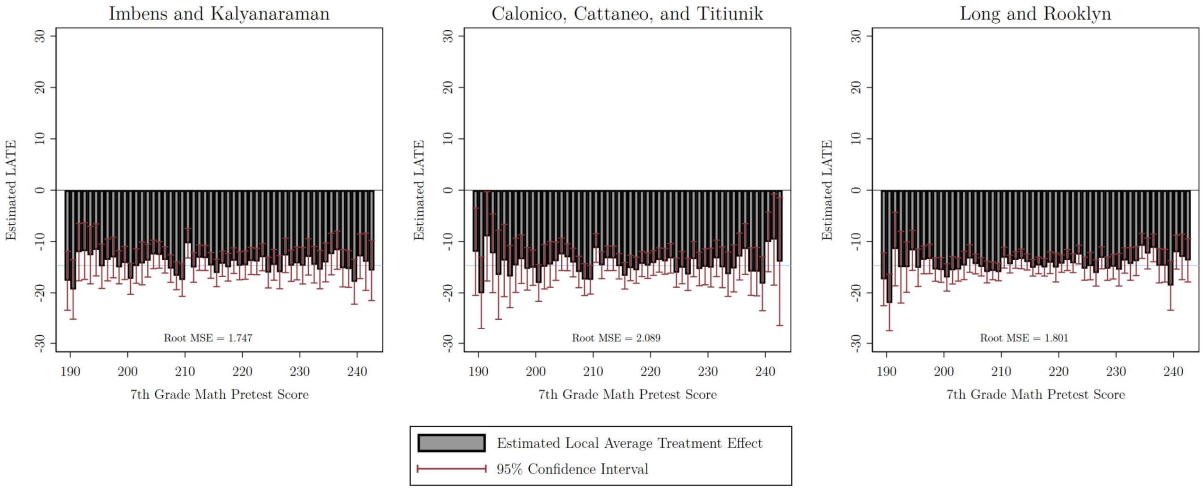
Panel AH: Treatment Effect $= -0.5\sigma/2^{\theta}$



Panel AI: Treatment Effect $= -\sigma - \sigma\theta$

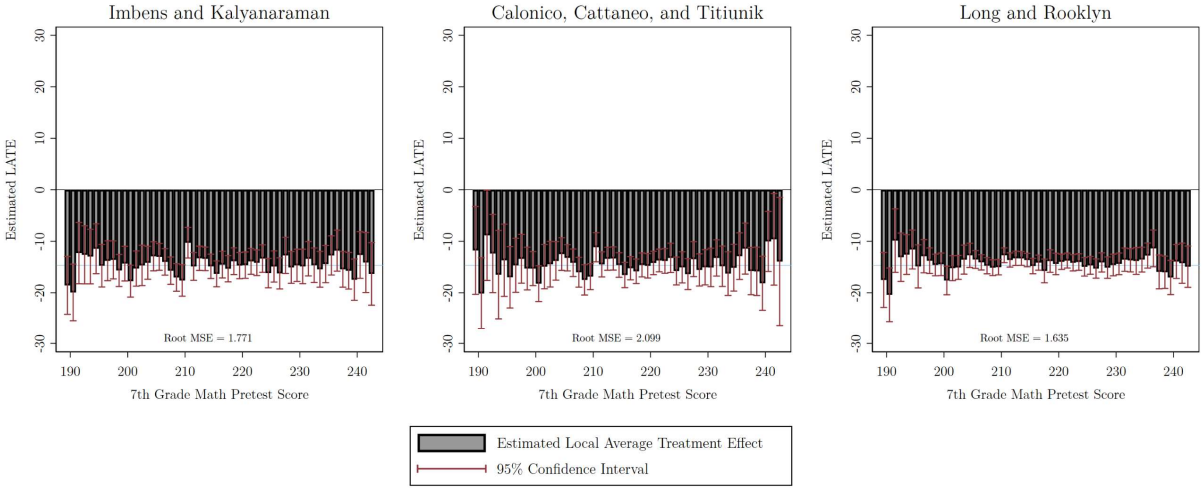


Panel AJ: Treatment Effect $= -\sigma - \sigma 2^{\theta}$



Appendix Figure 3: Continued

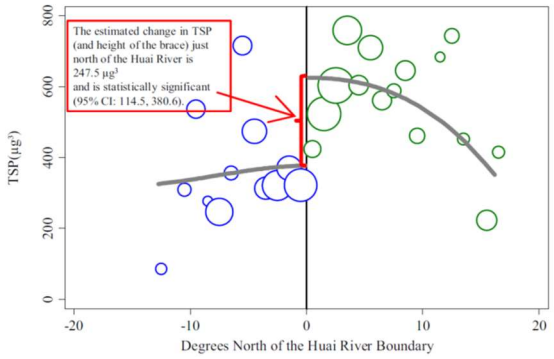
Panel AK: Treatment Effect $= -\sigma/2^{\theta}$



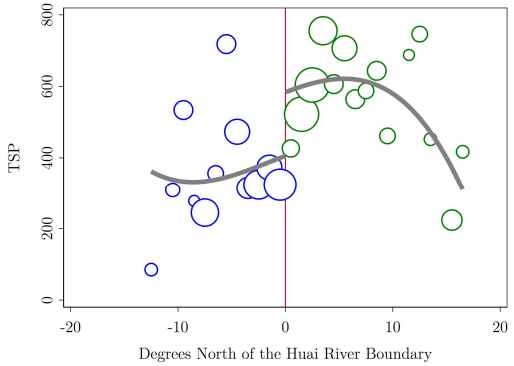
Note: Each figure shows 54 simulated impact estimates. The threshold of the simulated treatment is moved from 189.5 to 242.5 and the treatment effect is to the right of the threshold. The simulated LATE is shown by the light blue line in each figure. IK's method is estimated using Stata software and the *rd* command written by Nichols(2011). CCT's method is estimated using Stata software and the *rdrobust* command written by Calonico, Cattaneo, Farrell, and Titiunik (2018). LR's method is estimated using Stata software and the *next* command written by Long and Rooklyn (2020). Next's specification search was conducted across polynomial orders ranging from 1 to 3.

Appendix Figure 4:
Next Algorithm Applied to Estimates in Chen et al. (2013)

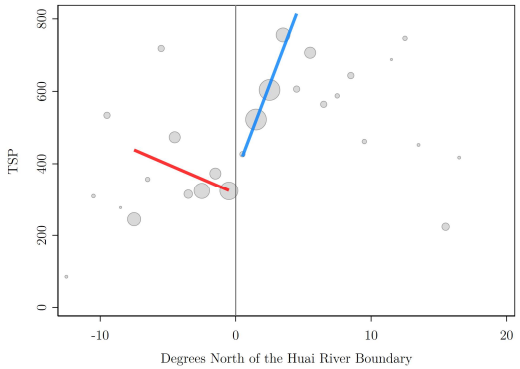
Panel A: Reprint of Chen et al.'s Figure 2



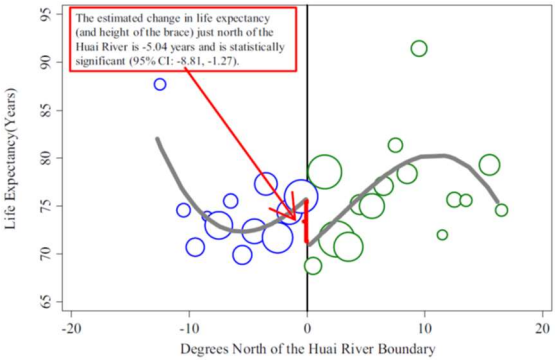
Panel C: Replication of Chen et al.'s Figure 2



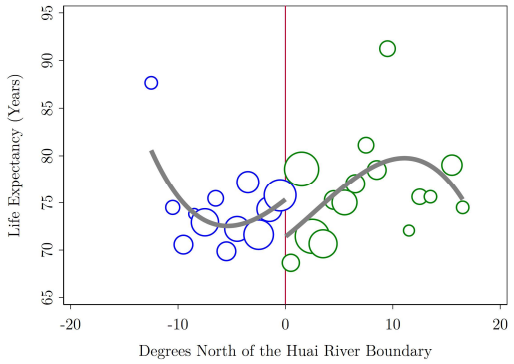
Panel E: Next Algorithm Results



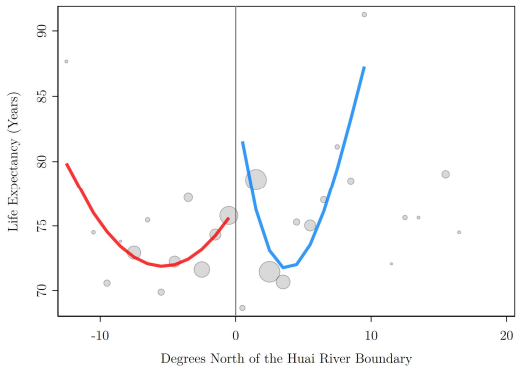
Panel B: Reprint of Chen et al.'s Figure 3



Panel D: Replication of Chen et al.'s Figure 3



Panel F: Next Algorithm Results



Note: Replication of Chen et al.'s results was attempted by hand measurement of the location and sizes of circles in Panels A and B. Next's specification search was conducted across polynomial orders ranging from 1 to 3. For Panel E, data on both sides are weighted by a triangular kernel and the specification is linear in x on both sides. For Panel F, data on both sides are weighted by a uniform kernel and the specification is quadratic (cubic) in x on the left (right) side.