DESCRIPTION OF MATLAB PROGRAMS FOR:

"Identification and Inference in Nonlinear Difference-In-Differences Models"

by Susan Athey & Guido Imbens

The following programs are used to calculate the tables in the supplementary materials to the paper "Identification and Inference in Nonlinear Difference-In-Differences Models." They include a general set of programs that can be used to calculate estimates for the CIC model. The main program is **cic.m**, which calculates the estimates for the continuous CIC model, and for the discrete CIC model with and without conditional independence assumption (for the latter case it calculates the lower and upper bounds). In the description below the labels generally refer to the labels used in the programs. The files are described in alphabetical order. To run the programs put all the files in one directory and from the Matlab command line type "main_mvd_05dec19" (to create Tables 1-3) or "main_sim_05dec19_start" (to create Table 4). The latter program can take a long time (up to a week), due to the bootstrapping. At any point it can be stopped with most of the simulations saved. They can be restarted by running the program "main_sim_05dec19_continue".

More recent versions of the programs as well as additional programs will be made available on our websites (from July 2006 onwards at the department of economics at Harvard University, http://www.economics.harvard.edu/).

1. **cdf.m**

   This function takes as input a scalar $y$, a vector of cumulative probabilities $\mathbf{P}$ and a vector of support points $\mathbf{YS}$. The vectors $\mathbf{P}$ and $\mathbf{YS}$ are the same length, say $N$. For a discrete random variable $Y$ with support $\mathbf{YS}$, and cumulative distribution function at the support points equal to $\mathbf{P}$ (that is, $\mathbf{P}_i = F_Y(\mathbf{YS}_i)$), the function **cdf.m** calculates the cumulative distribution function at the point $y$, $F_Y(y) = \Pr(Y \leq y)$. The scalar $y$ can take values outside the support of $Y$.

2. **cdf_bar.m**

   With the same inputs as the function **cdf.m**, the function **cdf_bar.m** calculates the probability that the random variable $Y$ is strictly less than $y$, (rather than less than or equal to), $\underline{F}_Y(y) = \Pr(Y < y)$. (See equation (41) in the paper).

3. **cdfinv.m**

   This function takes as input a scalar $q$ that is between zero and one (possibly equal to zero or one), a vector of cumulative probabilities $\mathbf{P}$ and a vector of support points $\mathbf{YS}$. The function **cdfinv.m** calculates the inverse of the cumulative distribution function of the random variable $Y$ at the point $q$, based on the formula in equation (8).

4. **cdfinv_bracket.m**

   With the same inputs as the function **cdfinv.m**, the function **cdfinv_bracket.m** calcu-

lates the alternative definition of the inverse of the cumulative distribution function of the random variable $Y$ at the point $q$, based on the formula in equation (24).

5. **cic.m**

   This is the main function for calculating the cic estimates. It takes as input seven arguments. The first four are the vectors of observations by group and time period, $\mathbf{Y}_{00}$, $\mathbf{Y}_{01}$, $\mathbf{Y}_{10}$, $\mathbf{Y}_{11}$. Next is a vector of quantiles $\mathbf{q}$. This vector can be of arbitrary length. In addition to returning estimates of the average effect of the treatment the program returns estimates of quantile effect for all the quantiles in this vector. This vector should not be empty but may contain just a single quantile. The next input is a scalar *standard_error*. If it is equal to one, analytic standard errors will be calculated for the estimates of the average effects. The final input argument is a scalar *bootstrap*. If it is zero, no bootstrap standard errors will be calculated. If it is a number greater than zero, bootstrap standard errors will be calculated with the number of bootstrap simulations equal to *bootstrap*.

   The function returns two outputs. The first is a row vector of estimates **est**. Let $Q$ be the dimension of the vector of quantiles $\mathbf{q}$. Then the length of the vector of estimates is $4 \cdot (Q + 1)$. The first $(Q + 1)$ estimates are for the continuous CIC estimator. The first estimate is for the average effect, followed by the quantile estimates for each of the quantiles in the $Q$ vector $\mathbf{q}$. The second $(Q + 1)$ estimates are for the discrete CIC with conditional independence estimator. The third $(Q + 1)$ estimates are for the lower bound of the discrete CIC estimator. The fourth $(Q + 1)$ estimates are for the upper bound of the discrete CIC estimator (these point estimates are identical to those for the continuous CIC estimator).

   The second output is the matrix **se**. The first row of this vector contains analytic estimates of the standard error in the same order as in the row vector **est**. If *standard_error* differs from one, then all elements of this row are equal to zero. If *standard_error* is equal to one, the elements corresponding to the estimates of the average effect contain the analytic standard errors described in the paper and in more computational detail in the supplementary materials. (No analytic standard errors are currently available for the quantile estimates. For the discrete case this is partly because asymptotic normality does not hold). If *bootstrap* is equal to zero then the matrix **se** contains only a single row. If *bootstrap* is greater than zero the second row of the matrix **se** contains the bootstrap standard errors for all estimates.

6. **cic_con.m**

   This function calculates the continuous CIC estimate. It takes as input seven arguments. The first four are the vectors of probabilities, $\mathbf{f}_{00}$, $\mathbf{f}_{01}$, $\mathbf{f}_{10}$, and $\mathbf{f}_{11}$ for four discrete random variables with support equal to the sixth argument, $\mathbf{YS}$. The fifth argument is the vector of quantiles $\mathbf{q}$ for which quantile effects will be calculated (in addition to the average

effect that will always be calculated. The sixth argument is the vector of support points for all outcomes. The last argument is the vector of support points $\mathbf{YS}_{01}$ for the second period control group.

The output is a row vector of estimates of length $(Q+1)$, where $Q$ is the dimension of the vector of quantiles $\mathbf{q}$. The first estimate is the estimator for the average effect, followed by the estimates for the quantile effects.

The calculations are based on first calculating the cdf of $Y_{11}^N$ using equation (9), followed by calculating the average effect based on this cdf.

7. **cic_dci.m**
   This function calculates the discrete CIC estimate under conditional independence. It takes as input seven arguments. The first four are the vectors of probabilities, $\mathbf{f}_{00}$, $\mathbf{f}_{01}$, $\mathbf{f}_{10}$, and $\mathbf{f}_{11}$ for four discrete random variables with support equal to the sixth argument, $\mathbf{YS}$. The fifth argument is the vector of quantiles $q$ for which quantile effects will be calculated (in addition to the average effect that will always be calculated. The last argument is the vector of support points $\mathbf{YS}_{01}$ for the second period control group.

   The output is a row vector of estimates of length $(Q+1)$, where $Q$ is the dimension of the vector of quantiles $\mathbf{q}$. The first estimate is the estimator for the average effect, followed by the estimates for the quantile effects.

   The calculations are based on first calculating the cdf of $Y_{11}^N$ using equation (29), followed by calculating the average effect based on this cdf.

8. **cic_low.m**
   This function calculates the lower bound for the discrete CIC estimator. It takes as input seven arguments. The first four are the vectors of probabilities, $\mathbf{f}_{00}$, $\mathbf{f}_{01}$, $\mathbf{f}_{10}$, and $\mathbf{f}_{11}$ for four discrete random variables with support equal to the sixth argument, $\mathbf{YS}$. The fifth argument is the vector of quantiles $q$ for which quantile effects will be calculated (in addition to the average effect that will always be calculated. The last argument is the vector of support points $\mathbf{YS}_{01}$ for the second period control group.

   The output is a row vector of estimates of length $(Q+1)$, where $Q$ is the dimension of the vector of quantiles $\mathbf{q}$. The first estimate is the estimator for the average effect, followed by the estimates for the quantile effects.

   The calculations are based on first calculating the cdf of $Y_{11}^N$ using equation (25), followed by calculating the average effect based on this cdf.

9. **cic_upp.m**
   This function calculates the upper bound for the discrete CIC estimator. It takes as input seven arguments. The first four are the vectors of probabilities, $\mathbf{f}_{00}$, $\mathbf{f}_{01}$, $\mathbf{f}_{10}$, and $\mathbf{f}_{11}$ for four discrete random variables with support equal to the sixth argument, $\mathbf{YS}$. The

fifth argument is the vector of quantiles $q$ for which quantile effects will be calculated (in addition to the average effect that will always be calculated. The last argument is the vector of support points $\mathbf{YS}_{01}$ for the second period control group.

The output is a row vector of estimates of length $(Q+1)$, where $Q$ is the dimension of the vector of quantiles $\mathbf{q}$. The first estimate is the estimator for the average effect, followed by the estimates for the quantile effects. These estimates are all identical to the corresponding estimates for the continuous CIC estimator.

The calculations are based on first calculating the cdf of $Y_{11}^N$ using equation (25), followed by calculating the average effect based on this cdf.

10. **cumdf.m**

    This function calculates the empirical cumulative distribution function given a set of observations. The function takes two inputs, a scalar $y$ at which the empirical distribution function is to be calculated, and an $N-$vector of observations $\mathbf{X}$. The output is a scalar:

$$\hat{F}_x(y) = \frac{1}{N}\sum_{i=1}^{N} 1\{\mathbf{X}_i \leq y\}.$$

11. **cumdfinv.m**

    This function calculates the inverse of the empirical cumulative distribution function given a set of observations. The function takes two inputs, a scalar $q$ at which the inverse of the empirical distribution function is to be calculated, and an $N-$vector of observations $\mathbf{X}$. The output is a scalar:

$$\hat{F}_X^{-1}(q) = \inf_y \left\{ -\infty < y < \infty \left| \frac{1}{N}\sum_{i=1}^{N} 1\{\mathbf{X}_i \leq y\} \geq q \right. \right\}.$$

12. **fden.m**

    This function estimates a marginal density function nonparametrically using kernel methods. The function takes two inputs, a scalar $y$ and an $N$-vector $\mathbf{Y}$. The estimated density will be evaluated at $y$. The observations $\mathbf{Y}$ are used to estimate the density. The bandwith is based on Silverman's rule of thumb:

$$h = 1.06 \cdot N^{-1/5} \cdot \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{Y}_i - \bar{Y})^2},$$

    where $\bar{Y} = \sum_i \mathbf{Y}_i/N$ is the sample average. The kernel is the Epanechnikov kernel:

$$K(u) = \begin{cases} (1-u^2/5)\cdot\frac{3}{4\cdot\sqrt{5}} & \text{if } |u| \leq \sqrt{5} \\ 0 & \text{otherwise.} \end{cases}$$

13. **gen.m**

    This program generates the artificial data sets for the simulations. It takes as inputs four scalar parameters, $\beta_{00}$, $\beta_{01}$, $\beta_{10}$, and $\beta_{11}$, and four subsample sizes $N_{00}$, $N_{01}$, $N_{10}$, and $N_1$. It returns three sets of data sets, the continuous data, the discrete data and the binary data. Each of the three data sets consists of four vectors. For the continuous data case these are labelled $\mathbf{Y}_{00}$, $\mathbf{Y}_{01}$, $\mathbf{Y}_{10}$, and $\mathbf{Y}_{11}$, for the discrete data case these are labelled $\mathbf{D}_{00}$, $\mathbf{D}_{01}$, $\mathbf{D}_{10}$, and $\mathbf{D}_{11}$, and for the binary data case these are labelled $\mathbf{B}_{00}$, $\mathbf{B}_{01}$, $\mathbf{B}_{10}$, and $\mathbf{B}_{11}$. The twelve vectors $\mathbf{Y}_{gt}$, $\mathbf{D}_{gt}$, and $\mathbf{B}_{gt}$ are all of length $N_{gt}$. First consider $\mathbf{Y}_{gt}$. The distribution of $\mathbf{Y}_{gt}$ has support $[0,1]$. Its probability density function is $\alpha_{gt} + \beta_{gt} \cdot y$ for $y \in [0,1]$ and zero elsewhere. The value of $\beta_{gt}$ is one of the inputs into the function. The value of $\alpha_{gt}$ follows from the value of $\beta_{gt}$ by the restriction that the density integrates out to one.

    The $i$th element of $\mathbf{D}_{gt}$ is calculated by rounding the $i$th value of $\mathbf{Y}_{gt}$ up to the next 0.1, so that $\mathbf{D}_{gt,i}$ can take on one of ten values, $\mathbf{D}_{gt,i} \in \{0.1, 0.2, \ldots, 1.0\}$. The $i$th element of $\mathbf{B}_{gt}$ is calculated by rounding the $i$th value of $\mathbf{Y}_{gt}$ up to the next 0.5, so that $\mathbf{B}_{gt,i}$ can take on one of two values, $\mathbf{B}_{gt,i} \in \{0.5, 1.0\}$.

14. **main_mvd_05dec19.m**

    This program calculates the values reported in the supplementary materials in Tables 1-3. It reads in the data and transforms them into the format used in the programs, that is, in four vectors $\mathbf{Y}_{00}$, $\mathbf{Y}_{01}$, $\mathbf{Y}_{10}$, and $\mathbf{Y}_{11}$.

15. **main_sim_05dec19_continue.m**

    See discussion for **main_sim_05dec19_start.m**.

16. **main_sim_05dec19_start.m**

    This program, in combination with the program **main_sim_05dec19_continue.m** carries out the simulations for Table 4. The program **main_sim_05dec19_start.m** starts the simulations. Every 100 simulations the results are saved to the file **out_sim_05dec19.mat**. If the program gets stopped the program **main_sim_05dec19_continue.m** can be used to continue. This program reads in the preliminary simulation results from the file **out_sim_05dec19.mat** and continues the simulations till 10,000 simulations have been carried out.

    The parameters for the simulations (the number of simulations, the number of bootstrap replications, and whether analytic standard errors are calculated are set in this program.

17. **mvd.dat**

    This ascii file contains the data used in the program **main_mvd_05dec11.m**. In the calculations for the paper we only use the the three variables injury duration ("y" in the programs, variable 1 in the data set), the time indicator ("after" in the programs, variable

3 in the data set) and the group indicator ("high" in the program, variable 4 in the data set). We only use the 5626 observations from Kentucky. The variables used are

18. **ols.m**
    The function returns ols estimates of a linear regression. It takes two inputs, an $N$-vector of outcomes $y$ and an $N \times K$ matrix of regressors $x$ (which should include a vector of ones if an intercept is to be included in the regression). The output is a $K \times 3$ matrix with the first column equal to the regression coefficients, the second column equal to the estimated standard errors (based on homoskedasticity, with the degrees of freedom for estimating the error variance equal to $N - K$), and the third column contains the t-statistics, equal to the ratio of estimates and standard errors.

19. **out_sim_05dec19.mat**
    This matlab data file contains the output from the simulations, including the number of simulations done and a variable **tabel4** which gives the entries for Table 4.

20. **prob.m**
    This function estimates the vector of probabilities of a discrete random variable given observations from its distribution. The function takes two inputs, an $N$-vector of observations $\mathbf{Y}$ and an $M$-vector of support points $\mathbf{YS}$. All the values in $\mathbf{Y}$ are required to be elements of $\mathbf{YS}$. The output is an $M$-vector $\mathbf{p}$, with each element in $\mathbf{p}$ equal to the proportion of observations in $\mathbf{Y}$ equal to the corresponding element in $\mathbf{YS}$:

    $$\mathbf{p}_m = \sum_{i=1}^{N} 1\{\mathbf{Y}_i = \mathbf{YS}_m\},$$

    for $m = 1, \ldots, M$.

21. **sim_theory.m**
    This program calculates the pseudo-true values for the simulations. For the continuous data this is done partly using numerical integration (although this calculation does not require actually calculating the estimator). For the lower and upper bound analytic calculations are done for the parameter values used in the simulations.

22. **supp.m**
    This function takes as input an $N$-vector $\mathbf{Y}$ and returns as output an $M$-vector $\mathbf{YS}$ containing all the distinct values in $\mathbf{Y}$, ordered from smallest to largest. If all the values of $\mathbf{Y}$ are distinct, then $M = N$ and $\mathbf{YS}$ is just the ordered version of $\mathbf{Y}$.

23. **tables123.mat**
    Output from the program **main_mvd_05dec19.m**, containing the results for Tables 1-3.