\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* TABLE OF CONTENTS:

\* 1. INTRODUCTION

\* 2. SET ENVIRONMENT

\* 3. SPECIFY TRAINING/TEST AND PERFORM INITIAL PREDICTIONS

\* 4. PERFORM CALCULATIONS FOR FIGURES

\* 5. OUTPUT FIGURES

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* 1. INTRODUCTION \*\*\*

/\*

Date created: December 24, 2017 (Updated October 1, 2018)

Created by: Aaron Berman

Project: Universal Basic Incomes versus Targeted Transfers

PIs: Rema Hanna, Ben Olken

Description: Analyzes coded ENAHO data and outputs paper Figure 7.

Uses: cleandata.dta (Peru ENAHO version)

Creates: coded\_test\_data.dta, plotting\_data.dta, coded\_test\_data with figure 7 variables.dta, Figure 7 graph

\*/

\*\*\* 2. SET ENVIRONMENT \*\*\*

version 14.1

set more off

clear all

pause on

capture log close

//setup

\*PLUG IN LOCAL FILE PATH HERE

\*global surveypath = ".../181009 Programs to publish"

global dopath = "$surveypath/Coding do files"

global datapath = "$surveypath/Data to use for Peru analysis"

global figurepath = "$surveypath/Paper Figures Combined"

if "$surveypath" == "" {

display "Enter the directory, WITHOUT a final slash: " \_request(surveypath)

}

if "$dopath" == "" {

display "Enter the directory, WITHOUT a final slash: " \_request(surveypath)

}

//set directory

cd "$datapath"

//open new ENAHO data

use "cleandata.dta"

duplicates drop

\*\*\* 3. SPECIFY TRAINING/TEST AND PERFORM INITIAL PREDICTIONS \*\*\*

set seed 83635628

generate random = runiform()

sort random

generate training = ceil(2 \* \_n/\_N) - 1

drop random

\*\*\*

//generate thresholds for 0.75x (extreme poor) and 1.5x (near poor) poverty line

gen povertyline75pct = 0.75\*povertyline

gen povertyline150pct = 1.5\*povertyline

gen extreme\_poor = (percapitaconsumption < povertyline75pct) //there are no missing values of percapitaconsumption

gen near\_poor = (percapitaconsumption < povertyline150pct) //so no need to worry about misleading 0s

//regress ln consumption on asset dummies for TRAINING data only

gen lnpercapitaconsumption = ln(percapitaconsumption)

reg lnpercapitaconsumption d\_\* if training == 1, vce(robust)

//predict ln consumption in the TEST set

predict lncaphat if training == 0

//also predict percapitaconsumption in test set

reg percapitaconsumption d\_\* if training == 1, vce(robust)

predict percapitahat if training == 0

//predict percapitaconsumption with ADDED NOISE in test set for Figure 7

predict percapitahat\_residuals if training == 0, residuals

//don't want to use above 95th percentile residuals

sum percapitaconsumption, d

gen sample\_use = (percapitaconsumption < r(p95)) if training == 0

replace percapitahat\_residuals = . if sample\_use == 0

//get standard deviation of residuals

egen percapitahat\_residuals\_sd = sd(percapitahat\_residuals) if training == 0

//set seed for rnormal()

set seed 70043812

//generate new predicted per-capita consumption values (in test set) with added noise

gen percapitahat\_morenoise = percapitahat + percapitahat\_residuals\_sd\*rnormal() if training == 0

\*\*\* 4. PERFORM CALCULATIONS FOR FIGURES \*\*\*

//drop training set so subsequent calculations run faster

drop if training == 1

//Inclusion/exclusion calculations

//first, create 101 different cutoff values corresponding to %iles of lncaphat.

gen c\_0 = 0

forvalues i = 1/99 {

egen c\_`i' = pctile(lncaphat), p(`i') //already excludes those for whom lncaphat is missing

}

egen c\_100 = max(lncaphat)

replace c\_100 = c\_100 + 1 //adding 1 arbitrarily so that max value of lncaphat is actually captured/included

//create local for overall program budget that can easily be tweaked later

local national\_num\_households = 6750000

local program\_budget\_monthly = 880000000/12

//this comes from Peruvian document regarding "Juntos" program budget

//count number of households in (non-missing) predicted test set (for calculating proportions)

egen samplesize = count(lncaphat)

//generate dummies for if individual is included or excluded based on 101 different cutoffs

forvalues i = 0/100 {

quietly{

//dummy if household is included

gen incl\_c\_`i' = (lncaphat < c\_`i') if !missing(lncaphat)

//number of individuals included

gen num\_incl\_`i' = incl\_c\_`i' \* h\_hhsize if !missing(lncaphat)

//inclusion error for poor, extreme poor, near poor curves

gen incl\_error\_normal`i' = (incl\_c\_`i' == 1) if !missing(lncaphat) & poor == 0

gen incl\_error\_extreme`i' = (incl\_c\_`i' == 1) if !missing(lncaphat) & extreme\_poor == 0

gen incl\_error\_near`i' = (incl\_c\_`i' == 1) if !missing(lncaphat) & near\_poor == 0

//exclusion error for poor, extreme poor, near poor curves

gen excl\_error\_normal`i' = (incl\_c\_`i' == 0) if !missing(lncaphat) & poor == 1

gen excl\_error\_extreme`i' = (incl\_c\_`i' == 0) if !missing(lncaphat) & extreme\_poor == 1

gen excl\_error\_near`i' = (incl\_c\_`i' == 0) if !missing(lncaphat) & near\_poor == 1

//calculate total number and % of households included for each cutoff

egen households\_incl\_`i' = total(incl\_c\_`i')

gen pct\_households\_incl\_`i' = households\_incl\_`i' / samplesize

//calculate (NATIONAL) per-household benefits for each cutoff

gen national\_hh\_incl`i' = `national\_num\_households' \* pct\_households\_incl\_`i'

gen per\_hh\_benefits`i' = `program\_budget\_monthly' / national\_hh\_incl`i'

//for each household, sum per-capita consumption and per-capita benefits (if included for this cutoff)

gen benefits\_received\_`i' = 0 if !missing(lncaphat)

replace benefits\_received\_`i' = per\_hh\_benefits`i' if incl\_c\_`i' == 1

gen percapita\_benefits\_received\_`i' = benefits\_received\_`i' / h\_hhsize

gen income\_`i' = percapitaconsumption + percapita\_benefits\_received\_`i'

//calculate individual CRRA utility

gen crra`i' = ((income\_`i')^(-2))/(-2) if !missing(lncaphat)

}

}

//generate percentile scores for ACTUAL percapitaconsumption for horizontal equity calculations

sort percapitaconsumption

gen income\_percentile = ((\_n - 1)/ \_N) \* 100

replace income\_percentile = floor(income\_percentile)

replace income\_percentile = . if missing(lncaphat) // want to exclude missing lncaphat FOR NOW

//calculate horizontal equity by percentile for each cutoff value

forvalues i = 0/100 { //NOTE: THIS STEP TAKES A LONG TIME. Could abbreviate to i= 0(10)100 for faster runtime

quietly {

//initialize inclusion/exclusion PERCENTAGES given cutoff i

gen included\_within\_band`i' = 0 if !missing(income\_percentile)

gen excluded\_within\_band`i' = 0 if !missing(income\_percentile)

//iterate through each income percentile and calculate percentage included within +/- 5%, given cutoff i

forvalues j = 0/99 {

local jminus5 = `j' - 5

local jplus5 = `j' + 5

egen pct\_included = mean(incl\_c\_`i') if income\_percentile >= `jminus5' & income\_percentile <= `jplus5'

replace included\_within\_band`i' = pct\_included if income\_percentile == `j'

replace excluded\_within\_band`i' = 1 - pct\_included if income\_percentile == `j'

drop pct\_included

local numloops = `i'\*100 + `j'

noisily display "`numloops'"

}

gen pct\_treated\_different`i' = included\_within\_band`i' if incl\_c\_`i' == 0

replace pct\_treated\_different`i' = excluded\_within\_band`i' if incl\_c\_`i' == 1

}

}

//incorporate discontinuous jump in per-household benefits and welfare for ubi only

gen bonus\_perhh = 2.235/12

//note: this comes from costs of 4.7million USD / year

//multiplied by 3.21 soles/USD

//divided by 12 in order to make sure we are dealing with montly benefits

gen bonus\_percapita = bonus\_perhh / h\_hhsize

//for per-household benefits discontinuity

gen disc\_per\_hh\_benefits100 = per\_hh\_benefits100 + bonus\_perhh if !missing(lncaphat)

//for per-capita consumption / social welfare discontinuity

gen disc\_income\_100 = income\_100 + bonus\_percapita if !missing(lncaphat)

gen disc\_crra100 = ((disc\_income\_100)^(-2))/(-2) if !missing(lncaphat)

//save the coded test dataset as-is

save "coded\_test\_data", replace

//calculate inclusion/exclusion error rate and total number of HH/individuals included for each cutoff value

collapse (mean) incl\_error\_\* excl\_error\_\* included\_within\_band\* excluded\_within\_band\* pct\_treated\_different\* national\_hh\_incl\* per\_hh\_benefits\* disc\_per\_hh\_benefits\* (sum) incl\_c\_\* num\_incl\_\* crra\* disc\_crra\*

//reshape long so we have 1 observation (inclusion error, 1-exclusion error) for each cutoff point

gen nth = (\_n)

reshape long incl\_error\_normal incl\_error\_near incl\_error\_extreme excl\_error\_normal excl\_error\_near excl\_error\_extreme incl\_c\_ num\_incl\_ ///

national\_hh\_incl per\_hh\_benefits disc\_per\_hh\_benefits crra disc\_crra included\_within\_band excluded\_within\_band pct\_treated\_different, i(nth) j(cutoff)

drop nth

rename incl\_c\_ households\_included

rename num\_incl\_ individuals\_included

//calculate 1-exclusion rate for plotting on y-axis

foreach x of varlist excl\_error\_normal excl\_error\_near excl\_error\_extreme {

gen oneminus\_`x' = 1 - `x'

}

//incorporate discontinuity for UBI

gen per\_hh\_benefits\_with\_disc = per\_hh\_benefits

replace per\_hh\_benefits\_with\_disc = disc\_per\_hh\_benefits if cutoff == 100

gen crra\_with\_disc = crra

replace crra\_with\_disc = disc\_crra if cutoff == 100

replace disc\_per\_hh\_benefits = per\_hh\_benefits[101] if cutoff == 99

replace disc\_crra = crra[101] if cutoff == 99

gen x\_disc = 1

//gen equity measure

gen horizontal\_equity = 1 - pct\_treated\_different

//save data with plot points

save "plotting\_data", replace

\*\*\* 5. OUTPUT FIGURES \*\*\*

//Note: in this file, we only output Figure 7, which is specific to Peru

cd "$figurepath"

//reopen "coded\_test\_data"

cd "$datapath"

use "coded\_test\_data", clear

//Figure 7: Expected tax rate given 1x poverty line cutoff (NOTE THIS IS NOT 1.5x AS IN INDONESIA)

gen receive\_benefit = (percapitahat < povertyline) if !missing(percapitahat)

//fit lpoly function to (estimated) benefit receipt status, for 1x targeting

lpoly receive\_benefit percapitaconsumption if sample\_use == 1 & !missing(percapitahat), bw(75) generate(prob\_receive\_benefit) at(percapitaconsumption)

//take first derivative numerically

dydx prob\_receive\_benefit percapitaconsumption if sample\_use == 1 & !missing(percapitahat), generate(derivative\_prob\_receive)

//benefit amount: 100 soles/month per household

gen benefit\_amount = 100 if !missing(derivative\_prob\_receive)

//generate variable to plot

gen tax = -1 \* derivative\_prob\_receive \* benefit\_amount

//do same calculations with "more noise" version of percapitahat

gen receive\_benefit\_morenoise = (percapitahat\_morenoise < povertyline) if !missing(percapitahat\_morenoise)

//fit lpoly function to (estimated) benefit receipt status

lpoly receive\_benefit\_morenoise percapitaconsumption if sample\_use == 1 & !missing(percapitahat\_morenoise), bw(75) generate(prob\_receive\_benefit\_morenoise) at(percapitaconsumption)

//take first derivative numerically

dydx prob\_receive\_benefit\_morenoise percapitaconsumption if sample\_use == 1 & !missing(percapitahat), generate(deriv\_prob\_receive\_morenoise)

//generate variable to plot

gen tax\_morenoise = -1 \* deriv\_prob\_receive\_morenoise \* benefit\_amount

//save

save "coded\_test\_data with figure 7 variables", replace

//plot and save

cd "$figurepath"

//Implied tax rate, comparing normal vs. "more noise" version

twoway (lpoly tax percapitaconsumption if sample\_use == 1, bw(75)) (lpoly tax\_morenoise percapitaconsumption if sample\_use == 1, bw(75)), ///

ytitle(Implied Tax Rate) yline(0, lwidth(medthin) lcolor(black)) xtitle(Monthly per-capita household consumption (S.)) title(Peru) legend(order(1 "Normal OLS Prediction" 2 "OLS Prediction with Doubled Noise")) ///

xscale(range(0 1200)) xlabel(0 300 600 900 1200) graphregion(color(white))

graph export "Figure 7 Peru normal and added noise\_whitebg.png", replace

graph export "Figure 7 Peru normal and added noise\_whitebg.eps", replace

graph export "Figure 7 Peru.eps", replace