

DoItAllAtOnce.do

* Creates the welfare aggregate, poverty line, and HH master file

```
version 8
clear all
set mem 200m
capture log close
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"
do IToSections /* creates 7 files for each section of the s0000c.dta */
do Food1 /* transforms the food consumption info from wide to long */
do Roster /* creates a roster file, and a file with effective HH size: hsize.dta */
do Food2 /* estimates the value of food consumption - uv's, outliers, adds restaurants */
do Housing /* computes imputed rental values for owner-occupied houses */
do Durables /* generates a long file with ownership of durables */
do Durable_uservalue /* estimates the user value of a stock of 13 durables reported in the survey */
do Privileges /* estimates the value of HUS, medical and transport privileges */
do Medical /* estimates the value of out-of-pocket medical spending */
do pq_vector /* estimates a Laspeyres price index for food items, by region and separate for rural/urban
areas */
do Consumption /* generates a file with the consumption aggregate w/o food, by component */
do Pline /* estimates HH specific poverty lines */
do Master_HH /* creates Master_Ind and Master_HH, files with individual/HH characteristics */
do MasterHH_short /* creates MasterHH_short, file with the preferred welfare aggregate and poverty
counts */
```

IToSections.do

* creates 7 files for each section of the s0000c.dta

```
version 8
clear all
set mem 150m
set more off
capture log close
log using IToSections.log, replace
tempfile weights
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"
use s0000c
generate double id = a001ter*100000000+ a002nast*10000+ a005nom /* generate the id variable */
generate urban = cond(a003pt == 8,2,1)
keep id kvzv urban
sort id
save `weights'

local i=1
foreach sec in roster employment pensions allowances bftcateg health awareness {
    use l0000c, clear
    generate double id = a001ter*100000000+ a002nast*10000+ a005nom /* generate the id variable */
    order id
    keep id a007id a`i'*
    renpfix a`i'
```

```

        sort id
        merge id using `weights'
        tabulate _merge
        drop _merge
    compress
    save p`i`_`sec', replace
    local i=`i' + 1
}
log close

```

Food1.do

```

*****
* Create a Food Consumption -- long format      *
*****
version 8
clear all
set mem 200m
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"
capture log close
log using Food1.log, replace
use s0000c, clear
* generate the id variable
gen double id = a001ter*100000000+ a002nast*10000+ a005nom
* keep id plus the food variables for the first 9 food groups
keep id r401m011 - r401m094
renprefix r401m a
local list 01 02 03 04 05 06 07 08 09
foreach j of local list {
    forvalues i = 1/4 {
        ren a`j`i' a`i`j'
    }
}
reshape long a10 a20 a30 a40, i(id) j(fid 1-9)
ren a10 a1
ren a20 a2
ren a30 a3
ren a40 a4
save food, replace

use s0000c, clear
* generate the id variable
gen double id = a001ter*100000000+ a002nast*10000+ a005nom
* keep id plus the food variables for the last 58 food groups
keep id r401m101 - r401m674
renprefix r401m a
gen a641 = .
gen a643 = .
gen a644 = .
forvalues j = 10/67 {
    forvalues i = 1/4 {

```

```

        ren a`j`i' a`i'b`j'
    }
}
reshape long a1b a2b a3b a4b, i(id) j(fid 10-67)
ren a1b a1
ren a2b a2
ren a3b a3
ren a4b a4
append using food
recode a1 -7=.
recode a2 -7=.
recode a3 -7=.
recode a4 -7=.

lab var id "HH id"
lab var fid "Food item code"
lab var a1 "Quantity purchased"
lab var a2 "Value purchased"
lab var a3 "Gifts"
lab var a4 "Own production"

#delimit ;
lab def fid
1 "Flour "
2 "Cereals "
3 "White bread "
4 "Rye- or other bread "
5 "Other bakery products and pastry"
6 "Pasta "
7 "Other farinaceous goods"
8 "Other cereal goods"
9 "Beef, veal"
10 "Pork"
11 "Lamb and goat's meet"
12 "Poultry meat including by-products"
13 "Meat of other domestic animals "
14 "Meat of wild animals "
15 "By-products "
16 "Sausages "
17 "Smoked meat and meat delicacies "
18 "Meat preserves "
19 "Convenience and ready meat food "
20 "Fish and seafood: live and frozen"
21 "Fish and seafood: salted, smoked, dried (herring excluded)"
22 "Sturgeon and salmon caviar (by weight and canned)"
23 "Salted herring"
24 "Fish preserves "
25 "Convenience and ready fish food "
26 "Fresh milk, liters"
27 "Preserved milk"
28 "Yogurt, cream, sour cream"

```

29 "Other dairy products"
 30 "Cheese"
 31 "Cottage cheese, curds"
 32 "Butter"
 33 "Margarine and other fats"
 34 "Vegetable oil"
 35 "Citrus fruit"
 36 "Apples"
 37 "Stone fruit"
 38 "Other fruit"
 39 "Water-melons, melons"
 40 "Grapes"
 41 "Other berries"
 42 "Dried fruit including grapes"
 43 "Nuts, stones, and edible seeds "
 44 "Frozen and canned fruit"
 45 "Cabbage"
 46 "Other green vegetables"
 47 "Cucumbers and tomatoes"
 48 "Gourds and other vegetables"
 49 "Onions and garlic"
 50 "Beet-roots, carrots, and other edibles roots"
 51 "Mushrooms"
 52 "Potatoes"
 53 "Legumes"
 54 "Vegetable preserves"
 55 "Convenience and ready vegetable food"
 56 "Sugar"
 57 "Jam, fruit butter"
 58 "Fruit preserves"
 59 "Natural honey"
 60 "Chocolate, chocolate candies "
 61 "Other sweets"
 62 "Ice cream"
 63 "Eggs"
 64 "Other food products"
 65 "Tea, coffee, cocoa (kg)"
 66 "Mineral water, soft drinks, juices, liters"
 67 "Alcoholic beverages, liters";
 #delimit cr
 lab val fid fid
 egen check = rsum(a1 a2 a3 a4)
 drop if check == 0
 drop check
 sort id fid
 save food, replace
 log close

Roster.do

```
* Household Roster - long format
version 8
clear all
set mem 200m
capture log close
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"
log using roster.log, replace
use s0000c, clear
gen double id = a001ter*100000000+ a002nast*10000+ a005nom
keep id r1a01m0c- r1a10m6
sort id
save roster, replace
renprefix r1a p
local list 01 02 03 04 05 06 07 08 09 10
foreach i of local list {
    ren p`i`m0c p1`i'
    ren p`i`m1g p2`i'
    ren p`i`m2d p3`i'
    ren p`i`m2m p4`i'
    ren p`i`m2y p5`i'
    ren p`i`m3      p6`i'
    ren p`i`m4      p7`i'
    ren p`i`m5      p8`i'
    ren p`i`m6      p9`i'
}
forvalues i=1/9 {
    forvalues j=1/9 {
        ren p`i`0`j' p`i`j'
    }
}
reshape long p1 p2 p3 p4 p5 p6 p7 p8 p9, i(id) j(pid)
keep if p5!=.
replace p5 = 1900+p5 if p5>3
replace p5 = 2000+p5 if p5<4
forvalues i = 1/4 {
    recode p`i` -100/0 = 0
}
forvalues i = 6/9 {
    replace p`i`= . if p`i`<=0
}
gen female = p2==2
drop p1 p2
ren p3 bday
ren p4 bmonth
ren p5 byear
ren p6 mabs
ren p7 rmabs
ren p8 yabs
ren p9 ryabs
```

```

order id pid female
lab var id "HH id"
lab var pid "Person id"
lab var bday "Day of birth"
lab var bmonth "Month of birth"
lab var byear "Year of birth"
lab var mabs "Absent in last 3 months"
lab var rmabs "Why?"
lab var yabs "Absent last 12 months"
lab var ryabs "Why?"
sort id pid
save roster, replace
use s0000c, clear
gen double id = a001ter*100000000+ a002nast*10000+ a005nom
keep id a011m1d a011m2m kvzv
gen w = kvzv
sort id
merge id using roster
drop _
keep if bday!=.
gen age = ((2003-byear)*12 + (a011m2m-bmonth) + (a011m1d-bday)/30.5)/12
gen sage = ((2003-byear)*12 + (a011m2m-bmonth-7) + (a011m1d-bday)/30.5)/12 /* school age */
keep id pid female age sage mabs rmabs yabs ryabs w
gen age_i = round(age)
gen sage_i = round(sage)
drop sage
ren sage_i sage
recode age -1/0 = 0
lab var age "Age, in year"
lab var age_i "Age, in rounded years"
lab var id HH_Id
lab var female Female
order id pid female age age_i sage
sort id pid
save roster, replace
recode mabs 90/99=90
gen absent = mabs/90
* drop if mabs>15 & mabs!=.
collapse (count) hsize=pid (sum) absent (mean) w, by(id)
gen ehsize = hsize - absent
lab var hsize "HH size"
lab var ehsize "Effective HH size"
lab var w "Weights"
lab var absent "% time absent during last 3 months"
sort id
save hsize, replace
log close

```

Food2.do

```
version 8
clear all
set more off
set mem 200m
capture log close
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"
log using Food2.log,replace

* keep the weighting/cluster/regional variables and merge them with the food file
use s0000c, clear
gen double id = a001ter*100000000+ a002nast*10000+ a005nom
keep id a001ter a003pt kvzv
egen psu = group(kvzv)
gen region = a001ter
gen strata = a003pt
gen w = kvzv
keep id region strata psu w
sort id
merge id using hsize
drop if _<3
drop _
sort id
merge id using food
drop if _<3
drop _
order id region strata psu w fid a1 a2 a3 a4
gen uv = a2/a1

* correction: price and quantity outliers
* price outliers: less than 20% of 5 times higher than the median price per item per region
* they are replaced by 20% of, or 5 times the median price per region
egen uv_rf = median(uv), by(fid region)
egen uv_rf1 = median(uv), by(fid)
replace uv_rf = uv_rf1 if uv_rf==.
drop uv_rf1
* Generate Table A3
table fid if (uv>5*uv_rf) & uv!=., c(mean uv mean uv_rf n uv) f(%4.2f)
replace uv = 5*uv_rf if (uv>5*uv_rf) & uv!=.
* Generate Table A4
table fid if uv<0.2*uv_rf, c(mean uv mean uv_rf n uv) f(%4.2f)
replace uv = 0.2*uv_rf if uv<0.2*uv_rf
drop uv_rf
* quantity outliers: per capita consumption > 5*median pc consumption, for those with positive
consumption, per item and area
* they are replaced by 5*median pc consumption
egen qtot = rsum(a1 a3 a4)
gen qpc = qtot/ehsize
egen qpc_med = median(qpc), by(fid region)
gen qpc_max = qpc_med*5
```

```

* Generate Table A5
table fid if qpc>qpc_max & qpc!=., c(mean qpc mean qpc_max mean qpc_med n qpc) f(%4.2f)
replace qpc = qpc_max if (qpc>qpc_max) & qpc!=.
replace qtot = qpc*ehsize
gen q1 = a1
replace q1 = qtot if qtot<a1
gen q2 = qtot-q1
drop qpc_med qpc_max
lab var q1 "Quantity purchased, no outliers"
lab var q2 "Quantity in-kind, no outliers"
lab var uv "Unit values, no outliers"
lab var qtot "Quantity consumed from all sources, total"
lab var qpc "Quantity consumed per capita from all sources"

* concentric imputation of prices
egen uv1 = median(uv), by(fid psu)
egen uv2 = median(uv), by(fid region strata)
egen uv3 = median(uv), by(fid region)
egen uv4 = median(uv), by(fid)
replace uv = uv1 if uv==.
replace uv = uv2 if uv==.
replace uv = uv3 if uv==.
replace uv = uv4 if uv==.
drop uv1-uv4

* estimate value of food purchases, and in-kind
gen v1 = q1*uv /* purchased */
replace v1 = a2 if fid==64 /* add value spend on "other food" items */
gen v2 = q2*uv /* in-kind */
lab var v1 "Value food, purchased"
lab var v2 "Value food, in-kind"
sort id fid
save food1, replace

* aggregate at HH level, separate food and alchoolic bevarages
generate div = cond(fid==67, 2, 1) /* alchoolic beverages */
egen v = rsum(v1 v2)
lab var v "Food consumption"
collapse (sum) v, by(id div)
reshape wide v, i(id) j(div)
mvencode _all, mv(0) override
lab var v1 "Food consumption"
lab var v2 "Alchoolic beverages"
sort id
save food2, replace

* add information on estimated spending and spending in estaurants/canteens over last 14 days
use s0000c, clear
gen double id = a001ter*100000000+ a002nast*10000+ a005nom
keep id r402-r406
local list r402 r403 r404m1 r404m2 r404m3 r405m1 r405m2 r405m3 r406

```

```

foreach x of local list {
    recode `x' -100/0 = 0
}
egen v3 = rsum(r405m1 r405m2 r405m3)
lab var v3 "Restaurants, Canteens"
ren r403 foodlevel
ren r406 nutrition
keep id r402 v3 foodlevel nutrition
order id r402 v3 foodlevel nutrition
sort id
merge id using food2
drop _
sort id
merge id using hsize
drop _
local list r402 v1 v2 v3
foreach x of local list {
    recode `x' . = 0
}
replace v1 = r402 if v1==0
drop r402
order id w hsize ehsize absent v1 v2 v3 foodlevel nutrition
ren v1 food
ren v2 alcohol
ren v3 eatout
sort id
save food3, replace
log close

```

Housing.do

```

version 8
set more off
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"
capture log close
log using Housing.log, replace

```

* 1. Generate the housing file

```

use s0000c, clear
* generate the id variable
gen double id = a001ter*100000000+ a002nast*10000+ a005nom
* keep id, regional and housing variables
keep id kvzv a001ter a002nast a003pt a004nush a005nom a006hhnn a007hht a008hrrr r201-r214m4
r216m1-r225m1 l
egen psu = group(kvzv)
order id psu kvzv
sort id
save housing, replace

```

* Impute ownership and living space at median PSU level

```

recode r202 -7/0=.
recode r208m02 -7/0=.
bysort psu: egen r202m = median(r202)
recode r202m 1.5=1 2.5=2 3.5=3
replace r202 = r202m if r202==. | r202<0
bysort psu: egen r208m02m = median(r208m02)
replace r208m02 =r208m02m if r208m02 ==. | r208m02<0
drop r202m r208m02m

```

* 2. Test the variables rent and estimated rental value

```

local list r202 r204 r205 r206 r207 r208m01 r208m02 r211 r212m5 r213 r214m1 r224
foreach x of local list {
    recode `x' -100/0 =.
}
forvalues i = 16/23 {
    local list r2`i'm1 r2`i'm2 r2`i'm3 r2`i'm4 r2`i'm5 r2`i'm6
        foreach x of local list {
            recode `x' -100/0 =.
        }
}
local list r207 r209 r212m1 r212m2 r212m3 r212m4 r212m5 r212m6 r212m7
foreach x of local list {
    recode `x' 2=0
}

```

* Estimated rental value

```
tabstat r203, st(p5 p50 p95 n) by( a003pt)
```

* Housing maintenance net of subsidy

```
tabstat r216m1, st(p5 p50 p95 n) by( a003pt)
```

* Full housing maintenance (including housing subsidies)

```
egen hm = rsum(r216m1 r216m4)
```

```
recode hm 0=.
```

```
tabstat hm, st(p5 p50 p95 n) by( a003pt)
```

```
gen rent1 = hm>0 & hm!=.
```

```
gen irent1 = r203>0 & r203!=.
```

```
tab rent1 irent1
```

```
tabstat r203 hm if rent1==1 & irent1==1, st(p5 p50 mean p95 n) c(st) f(%4.0f)
```

* Note: estimated rental value is about 5 times bigger than housing maintenance payment+subsidy

* 3. Estimate a hedonic rent equation based on estimated rent

```
gen lir = ln(r203) /* estimated rent */
```

```
qui xi: areg lir i.r201 i.r202 i.r205 r207 r208m02 r209 i.r210 i.r211 r212m1 r212m2 r212m3 r212m4
r212m5 r212m6 r212m7 i.a003pt, absorb(a001ter)
```

```
outreg using rent.out, replace
```

```
predict r
```

* estimation of a parsimonious housing regression

```
version 9
```

```
xi: stepwise, pr(0.2): reg lir i.r201 i.r202 i.r205 r207 r208m02 r209 i.r210 i.r211 r212m1 r212m2 r212m3
r212m4 r212m5 r212m6 r212m7 i.a003pt i.a001ter
```

```
version 8
```

```

* Impute rent for all households based on the "estimated rent" regression model
* generate the rent variable, predicted from regression model
gen rent = exp(r)
recode rent .=0
lab var rent "rent, all HHs"
* generate the rentp variable for the simulations, only for tenants
gen rentp = rent
recode rentp 0/110000 = 0 if r202==2
lab var rentp "rent, tenants only"
table r202, c(m rent m rentp n rent)

* some descriptive stats about the dwelling stock
* by ownership
gen rural = (a003pt==8)
tab r202 rural [aw=kvzv], nof col
* by type of dwelling
tab r201 rural [aw=kvzv], nof col
* who received subsidies?
gen subsidy = (r216m3!=.)
* who report housing maintenance (paid) and % receiving subsidies?
table r202 [aw=kvzv], c(p50 r216m1 n r216m1 sum subsidy) row
* ratio of estimated rent to housing maintenance plus subsidy
gen ratio = r203/hm
recode ratio -10/0=. 25/1000=.
histogram ratio
su ratio, d
* estimated rent, hm, and hm+subsidy by type of ownership and dwelling type
table r202 [aw=kvzv], c(p50 rent p50 r216m1 p50 hm) f(%4.0f)
table r201 [aw=kvzv], c(p50 rent p50 r216m1 p50 hm) f(%4.0f)
sort id
save rent_long, replace
keep id a001ter rural rent rentp
sort a001ter rural
save rent_temp, replace

* estimate the Laspeyres housing price index, regional*area
* estimating the population for each oblasts / area to use as weights
use hsize, clear
keep id hsize
sort id
merge id using rent_long
assert _merge == 3
drop _merge
gen pop1 = kvzv*hsize
bysort a001ter rural: egen pop = total(pop1)
drop pop1
* examining the characteristics of the average house
su r207 r208m02 r209 r212m1 r212m2 r212m3 r212m4 r212m5 r212m6 r212m7
tab1 r201 r202 r205 r210 r211
qui xi: reg lir i.r201 i.r202 i.r205 r207 r208m02 r209 i.r210 i.r211 r212m1 r212m2 r212m3 r212m4
r212m5 r212m6 r212m7 rural i.a001ter

```

```

adjust _Ir201_2=0 _Ir201_3=0 _Ir201_4=0 _Ir201_5=0 _Ir201_6=0 _Ir202_2=1 _Ir202_3=0 _Ir202_4=0
_Ir202_5=0 /*
*/ _Ir205_2=1 _Ir205_3=0 _Ir205_4=0 _Ir205_5=0 _Ir210_2=0 _Ir210_3=0 _Ir210_4=0 _Ir211_2=0 /*
*/ _Ir211_3=0 _Ir211_4=0 _Ir211_5=0 _Ir211_6=0 _Ir211_7=0 r212m3=1 r212m4=1 r212m5=1
r212m6=1 r212m7=1, by(a001ter rural) replace
keep a001ter rural xb pop
gen prent = exp(xb)
drop xb
su prent [aw = pop]
gen base = r(mean)
su base
replace prent = prent/base
su prent
table a001ter rural [aw=pop], c(m prent) f(%5.4f) row col
drop base pop
sort a001ter rural
save phousing, replace
merge a001ter rural using rent_temp
erase rent_temp.dta
assert _merge == 3
drop _merge
* eliminate rent outliers, values less than 200 RUR, 486 cases
recode rent 0/200=200
* deflate rent by the housing price index
gen rent_nd = rent
lab var rent_nd "rent, all HHs, current prices"
gen rentp_nd = rentp
lab var rentp_nd "rent, tenants only, current prices"
replace rent = rent_nd/prent
replace rentp = rentp_nd/prent
sort id
save rent_deflator, replace
keep id rent rentp
sort id
save rent, replace

log close

```

Durables.do

```

version 8
clear all
set more off
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"
capture log close
log using Durables.log, replace

* 1. Generate the file with info on HH durables
use s0000c, clear
* generate the id variable
gen double id = a001ter*100000000+ a002nast*10000+ a005nom

```

```

* keep id, regional and durables
keep id r409m011-r409m235
order id
renprefix r409m a
forvalues i = 1/23 {
    forvalues j = 1/5 {
        if (`i'<10) {
            rename a0`i`j' a`j'_'i'
        }
        else {
            rename a`i`j' a`j'_'i'
        }
    }
}
reshape long a1_ a2_ a3_ a4_ a5_, i(id) j(did)
rename a1_ d1
rename a2_ d2
rename a3_ d3
rename a4_ d4
rename a5_ d5
local list d1 d2 d3 d4 d5
foreach x of local list {
    recode `x' -10/0 = .
}
egen check = rsum(d1 d2 d3 d4 d5)
drop if check == 0
drop check
recode d4 2=0
lab var did "Id Durable"
#delimit ;
lab def did
    1 "TV" 2 "Video recorder" 3 "Video camera" 4 "Radio" 5 "Music center"
    6 "Tape recorder" 7 "Refrigerator" 8 "Freezer" 9 "Washing machine" 10 "Microwawe owen"
    11 "Dishwasher" 12 "Vacuum cleaner" 13 "Sewing machine" 14 "Knitting machine" 15 "Air-
conditioner"
    16 "PC" 17 "Mobile telephone" 18 "Bicycle" 19 "Passenger car" 20 "Motorcycle"
    21 "Truck, bus" 22 "Motor boat" 23 "Other vechicles"
;
#delimit cr
lab val did did
lab var d1 "# items"
lab var d2 "Year purchase"
lab var d3 "Cost at purchase"
lab var d4 "Purchased?"
lab var d5 "Market value today"
sort id did
save durables, replace

```

Durable_uservalue.do

```
version 8
clear all
set more off
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"
capture log close
log using Durable_uservalue.log, replace

* bring in location identifiers into durables.dta
use s0000c, clear
gen double id = a001ter*100000000+ a002nast*10000+ a005nom
ren a001ter ter
ren a002nast nast
ren a003pt pt
gen rural = pt == 8
keep id ter nast pt rural
sort id
save temp, replace
use durables, clear
sort id
merge id using temp
drop if _ == 2
drop _
erase temp.dta

* estimate age of durable good, in years
gen t = 2003 - d2
* estimate real value of the durable good in 2003; we do not use this in predictions
gen d3r = d3
replace d3r = d3r*2.216 if d2==1998
replace d3r = d3r*1.171 if d2==1999
replace d3r = d3r*1.062 if d2==2000
replace d3r = d3r*1.064 if d2==2001
replace d3r = d3r*1.015 if d2==2002
gen y = ln(d5)

* Issue 1: Impute resale prices for durables with missing values
* For how many durables we have to impute resale prices?
* Table 4
* # of durables, # with purchase prices, # with resale prices
table did, c(n t n d3 n d5)
gen aged = t
recode aged 1/5=1 6/10=2 11/15=3 16/103=4
tab did aged, nof row

gen ph = 0
forvalues i = 1/23 {
    tobit y t if did==`i', ll(5)
    mat B = e(b)
    gen phat = B[1,2]+B[1,1]*t if did==`i'
```

```

        replace ph = phat if did==`i'
        drop phat
        mat drop B
    }
table did, c(m ph n ph)

* inspecting the fit of the prediction
table did t if t<11, c(m y m ph) f(%4.1f)
gen p = exp(ph)
table did t if t<11, c(m d5 m p) f(%4.0f)

* implausible predictions for dishwasher (small sample), motor boats and other vehicles
* did = 11, 22 & 23
* replace value with the median resale value reported for 1998-2003
bysort did: egen mprice = median(d5) if d2==1998
bysort did: egen mprice1 = median(d5)
table did, c(m mprice n mprice m mprice1 n mprice1)
drop mprice
replace p = mprice1 if did==11
replace p = mprice1 if did>21

* place a floor of 10% of the median resale value of a new item
bysort did: egen mprice = median(d5) if d2>2001
bysort did: egen mpr = max(mprice)
drop mprice
replace mpr = 0.1*mpr
replace p = mpr if p<mpr | p==./* this price is imputed to durables without reported age as well */

* inspecting the fit of the prediction
table did t if t<11, c(m d5 m p) f(%4.0f)

* generate variable with resale price, reported if not imputed
gen resale = p
replace resale=d5 if d5!=.

* estimate dep: HH-level depreciation rate for a given durable function of its age
gen dep = 1-( d5/ d3r)^(1/t)
* Table 6
table did, c(p5 dep p50 dep p95 dep)
* calculate the median depreciation rate for each durable to minimize the influence of outliers, we prefer
median to mean
egen depm = median(dep), by(did)

* estimate the MONTHLY user value of durables from two components:
* depreciation, and opportunity cost of the money tied up in the durable, @ 5% real annual interest rate
(assumed)
gen uvalue = resale*(dep + 0.05)/(12*(1-depm))
* Table 7
table did, c(p5 uvalue mean uvalue p95 uvalue) f(%4.1f)
sort id did
save tempdurable, replace

```

```
keep id uvalue
collapse (sum) uvalue, by(id)
sort id
save uvalue, replace
```

```
log close
```

Privileges.do

```
*****
```

```
* Welfare derived from Privileges & other subsidies *
```

```
*****
```

```
version 8
set more off
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"
capture log close
log using Privileges.log, replace
```

```
* TRANSPORT
```

```
use p5_bftcateg, clear
local list "04 05 07 08 10 11 13 14"
foreach i of local list {
    recode e`i' -7/0 = 0 . = 0
}
```

```
* notresponse is low !
```

```
gen l_transp = e04*e05 + e07*e08 + e10*e11 + e13*e14*.5
collapse (sum) l_transp, by(id)
* Correct for outliers (cap to 5000 Rubles 37 large outliers out of 11000 cases)
recode l_transp 5000/55000 = 5000
sort id
save l_transp, replace
```

```
* MEDICAL
```

```
use p6_health.dta, clear
* Medicines (j20):
gen j20c = j20
recode j20c 1/36000 = 1
tab j20c j19
* from 11,000 indiv who are privileged, 50% did not use it, 25% cannot estimate
* impute the median saving by group (50% discount; 100% discount)
su j20 if j20c==1 & j19==2, d
di r(p50)
gen p50 = r(p50)
su j20 if j20c==1 & j19==3, d
di r(p50)
gen p100 = r(p50)
gen l_med = j20
recode l_med . = 0 -9 = 0
replace l_med = p50 if j20==7 & j19==2
```

```

replace l_med = p100 if j20==7 & j19==3
drop p50 p100 j20c
* Medical services (inpatient and outpatient)
* nonresponse is low
recode j09s1 -9/0 = 0
recode j09s2 -9/0 = 0
egen l_serv_i = rsum(j09s1 j09s2) if j08s2==2
replace l_serv_i = l_serv_i/3
forvalues i = 1/5 {
    recode j15s`i' -9/0 = 0
}
egen l_serv_o = rsum(j15s1 j15s2 j15s3 j15s4 j15s5) if j16s2==2
replace l_serv_o = l_serv_o/3
local list "l_med l_serv_i l_serv_o"
foreach i of local list {
    recode `i' 0=.
}
su l_med l_serv_i l_serv_o
* Correct for outliers (cap the distribution at 5000 Rubles for medical services: 118 large outliers)
recode l_med 5000/36000 = 1500
recode l_serv_i 5000/35000 = 5000
recode l_serv_o 5000/40000 = 5000
su l_med l_serv_i l_serv_o
egen l_med = rsum(l_serv_i l_serv_o l_med)
drop l_serv_i l_serv_o l_med
collapse (sum) l_med, by(id)
sort id
save l_med, replace

* HOUSING AND COMMUNAL SERVICES
use housing, clear
local list "r216m4 r217m4 r218m4 r219m4 r220m4 r221m4 r222m4 r223m4"
foreach i of local list {
    recode `i' -9/0 = 0
}
* about 5% non-response
egen l_hus = rsum(r216m4 r217m4 r218m4 r219m4 r220m4 r221m4 r222m4 r223m4)
keep id l_hus
sort id
merge id using l_med
drop _
sort id
merge id using l_transp
drop _
egen lgoti = rsum(l_hus l_med l_transp)
su
sort id
save privileges, replace

* RENT -- I account for the full welfare derived from the dwelling separately
* including rent privileges would result in double counting

```

```
erase l_medica.dta
erase l_transp.dta
```

```
log close
```

Medical.do

```
*****
* Medical costs, non-subsidized *
*****

version 8
set more off
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"
capture log close
log using Medical.log, replace
use p6_health.dta, clear

* Medicines (j18)
recode j18 -9/0=0 . = 0
* Medical services, outpatient (j09s1 j09s2), out of pocket
* nonresponse is low
recode j09s1 -9/0 = 0
recode j09s2 -9/0 = 0
egen med_o = rsum(j09s1 j09s2)
egen med_lgoti = rsum(j09s1 j09s2) if j08s2==2
replace med_o = med_o - med_lgoti
drop med_lgoti
* Medical services, inpatient (j15s1 j15s2 j15s3 j15s4), out of pocket
forvalues i = 1/5 {
    recode j15s`i' -9/0 = 0
}
egen med_i = rsum(j15s1 j15s2 j15s3 j15s4 j15s5)
egen med_lgoti = rsum(j15s1 j15s2 j15s3 j15s4 j15s5) if j16s2==2
replace med_i = med_i - med_lgoti
drop med_lgoti

egen medic = rsum(j18 med_o med_i)
collapse (sum) medic, by(id)

* Correct for outliers (cap the distribution to 5000 Rubles (9 large outliers))
recode medic 1500/36000 = 5000
su medic, d
sort id
save Medical, replace
```

Pq_vector.do

```
*****
* pq_vector.do
* Generates a price and quantity vector
* Input files: food1
* Output files: pq_vector
*****

clear all
set mem 250m
capture log close
set more off
log using pq_vector.log, replace
di "Log run at $$_TIME on $$_DATE"
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"

* get info on how many people are in the (expanded) sample each year
    use roster, clear
    gen one = 1
    quietly su one [aw=w]
    local n = r(sum)
    di `n'

* create a national average q vector for each year
    use food1, clear
    keep if qtot!=.
    gen double q = qtot
    keep fid w q
    drop if q==.
    collapse (sum) q [iw=w], by(fid)
    replace q = q/n
    sort fid
    save q, replace

* create a file with all possible combinations of item*region
    use food1
    keep if qtot!=.
    keep qtot fid
    collapse (count) qtot, by(fid)
    drop qtot
    save quant, replace
    use s0000c, clear
    gen region = a001ter
    keep region a002nast
    collapse (count) a002nast, by(region)
    drop a002nast
    cross using quant
    sort fid region
    save ir, replace

* create a file with all possible combinations of item*region*rural
```

```

use s0000c, clear
keep a002nast a003pt
gen rural = a003pt == 8
collapse (count) a002nast, by(rural)
keep rural
save rural, replace
cross using ir
sort fid region rural
save irr, replace

* estimate a median price vector for whole Russia
set more off
use food1, clear
drop if qtot==.
collapse (p50) uv [iw=w], by(fid)
ren uv pn
lab var pn "Median price, national"
sort fid
save pn, replace

* estimate a median price vector for each region
set more off
use food1, clear
drop if qtot==.
collapse (p50) uv [iw=w], by(fid region)
sort fid region
merge fid region using ir
drop _
egen pr = median(uv), by(fid)
replace pr = uv if uv!=.
sort fid region
keep fid region pr
sort fid region
save pr, replace

* estimate a median price vector by area and region
set more off
use food1, clear
drop if qtot==.
gen rural = strata==8
collapse (p50) uv [iw=w], by(fid region rural)
sort fid region rural
merge fid region rural using irr
drop _
egen pl = median(uv), by(fid)
replace pl = uv if uv!=.
sort fid region rural
* table fid region if rural==1, c(mean pl) f(%4.0f)
* table fid region if rural==0, c(mean pl) f(%4.0f)
keep fid region rural pl
sort fid region rural

```

```

save p, replace

set more off
    use p, clear
    sort fid region rural
    merge fid using q
    drop _
    sort fid region rural
    merge fid region using pr
    drop _
    sort fid region rural
    merge fid using pn
    drop _
    gen vl = pl*q /* value of the basket in local prices */
    gen vr = pr*q /* value of the basket in regional prices */
    gen vn = pn*q /* value of the basket in national prices */
    collapse (sum) vl vr vn, by(region rural)
    reshape wide vl vr vn, i(region) j(rural)
    gen pindex_r = vr/vn0
    gen pindex_rr = vl1/vn0
    gen pindex_ru = vl0/vn0
    keep region pindex*

#delimit ;
lab def region
1 "Altaiskiy krai"
3 "Krasnodarskiy krai"
4 "Krasnoyarskiy krai"
5 "Primorskiy krai"
7 "Stavropolskiy krai"
8 "Habarovskiye krai"
10 "Amurskaya oblast"
11 "Arkhangelskaya oblast"
12 "Astrakhanskaya oblast"
14 "Belgorodskaya oblast"
15 "Bryanskaya oblast"
17 "Vladimirskaaya oblast"
18 "Volgogradskaya oblast"
19 "Vologodskaya oblast"
20 "Voronejskaya oblast"
22 "Nijegorodskaya oblast"
24 "Ivanovskaya oblast"
25 "Irkutskaya oblast"
26 "Republic Ingushetiya"
27 "Kaliningradskaya oblast"
28 "Tverskaya oblast"
29 "Kalujskaya oblast"
30 "Kamchatskaya oblast"
32 "Kemerovskaya oblast"
33 "Kirovskaya oblast"
34 "Kostromskaya oblast"

```

36 "Samarskaya oblast"
37 "Kurganskaya oblast"
38 "Kurskaya oblast"
40 "St-Petersburg"
41 "Leningradskaya oblast"
42 "Lipetskaya oblast"
44 "Magadanskaya oblast"
45 "Moscow"
46 "Moskovskaya oblast"
47 "Murmanskaya oblast"
49 "Novgorodskaya oblast"
50 "Novosibirskaya oblast"
52 "Omskaya oblast"
53 "Orenburgskaya oblast"
54 "Orlovskaya oblast"
56 "Penzenskaya oblast"
57 "Permskaya oblast"
58 "Pskovskaya oblast"
60 "Rostovskaya oblast"
61 "Ryazanskaya oblast"
63 "Saratovskaya oblast"
64 "Sakhalinskaya oblast"
65 "Sverdlovskaya oblast"
66 "Smolenskaya oblast"
68 "Tambovskaya oblast"
69 "Tomskaya oblast"
70 "Tulskaya oblast"
71 "Tiumenskaya oblast"
73 "Uliyanovskaya oblast"
75 "Chelyabinskaya oblast"
76 "Chitinskaya oblast"
77 "Chukotskiy AO"
78 "Yaroslavskaya oblast"
79 "Republic Adygeya"
80 "Republic Bashkortostan"
81 "Republic Buryatiya"
82 "Republic Dagestan"
83 "Kabardino-balkarskaya Republic"
84 "Republic Altai"
85 "Republic Kalmykiya"
86 "Republic Kareliya"
87 "Republic Komi"
88 "Republic Mariy El"
89 "Republic Mordovia"
90 "Republic Severnaya Osetiya"
91 "Karachaevo-cherkesskaya Republic"
92 "Republic Tatarstan"
93 "Republic Tyva"
94 "Udmurtskaya Republic"
95 "Republic Hakasiya"
97 "Chuvashskaya Republic"

```

    98 "Republic Saha (Yakutiya)"
    99 "Evreiskaya AO"
;
lab val region region;
#delimit cr

```

```

    sort region
    save pindex, replace

```

```
log close
```

Consumption.do

```
*****
```

```
* Create a file with HH consumption, by component
```

```
* Input file:
```

```
local infile1 "s0000c.dta"
```

```
local infile2 "p1_roster.dta" /* or "l0000c.dta" */
```

```
local infile3 "food3.dta"
```

```
* Output file:
```

```
local outfile "cons0.dta"
```

```
local path "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"
```

```
*****
```

```
clear all
```

```
set mem 200m
```

```
set more off
```

```
capture log close
```

```
log using cons.log, replace
```

```
* ----- Durables: create variable with the value of the durable goods purchased in 2003
```

```
tempfile durables
```

```
use durables, clear
```

```
keep if d2==2003
```

```
replace d5=d3 if d5==.
```

```
collapse (sum) durables=d5, by(id)
```

```
sort id
```

```
save `durables', replace
```

```
* ----- Weights
```

```
tempfile weights
```

```
use `infile1', clear
```

```
generate double id = a001ter * 100000000 + a002nast * 10000 + a005nom /* generate the id variable */
```

```
keep id kvzv
```

```
sort id
```

```
save `weights', replace
```

```
* ----- 10. Education
```

```
tempfile edu
```

```
use `infile2', clear
```

```

rename b19 v111900
rename b20s1 v112001
rename b20s2 v112002
rename b20s3 v112003
rename b20s4 v112004
keep id a007id v*
  reshape long v1, i(id a007id) j(code)
drop if v1<=0
  collapse (sum) v1, by(id code)
generate v2 = 0
save `edu', replace

use `infile1', clear
generate double id = a001ter * 100000000 + a002nast * 10000 + a005nom /* generate the id variable */
rename a001ter region

* ---- Questions 17-23 utilities v1=paid and v2=subsidy or gift
forvalues i = 17/23 {
  rename r2`i'm1 v12`i' /* HUS paid */
  * rename r2`i'm4 v22`i' /* HUS subsidies: privileges or allowances */
} /* end forvalues */

* ---- Question B.7 Non-food goods with 30 days recall
forvalues i = 1/11 {
  if (`i'<10) {
    rename r407m0`i' v14070`i'
  }
  else rename r407m`i' v1407`i'
} /* end forvalues */

* ---- Question B.8 Non-food goods with 12 months recall
forvalues i = 1/28 {
  if (`i'<10) {
    rename r408m0`i'1 v14080`i'
    rename r408m0`i'2 v24080`i'
  }
  else {
    rename r408m`i'1 v1408`i'
    rename r408m`i'2 v2408`i'
  } /* end if */
} /* end forvalues */

* Question C.10 - Services
forvalues i = 1/22 {
  if (`i'<10) {
    rename r410m0`i' v14100`i'
  }
  else rename r410m`i' v1410`i'
} /* end forvalues */

keep id region v*

```

```

reshape long v1 v2, i(id region) j(code)
replace v1 =0 if v1<0
replace v2 =0 if v2<0
drop if (v1==0 | v1==.) & (v2==0 | v2==.)

append using `edu' /* append education expenses from personal file */

save `outfile', replace
sort code
merge code using codes
tabulate _merge
drop _merge
egen v = rsum(v1 v2)
replace v = v * 30 / mlt
drop if v == 0

* ---- Outliers
tempname med
egen `med' = median (v), by(code region)
replace v = `med'*5 if v>`med'*5
replace v = `med'*0.2 if v<`med'*0.2

collapse (sum) v, by(id div)
reshape wide v, i(id) j(div)

* ---- Rent
sort id
merge id using rent
tabulate _merge
drop _merge
replace rentp = 10000 if rentp>10000
replace rent = 10000 if rent>10000

mvencode _all, mv(0) override
sort id
merge id using `weights'
tabulate _merge
drop _merge
rename v0 other

sort id
merge id using `durables'
tabulate _merge
* drop if _merge==2
drop _merge
recode durables .=0

sort id
merge id using uvalue
tabulate _merge
* drop if _merge==2

```

```
drop _merge
recode uvalue .=0
```

```
sort id
merge id using privileges
tabulate _merge
drop _merge
```

```
sort id
merge id using medical
tabulate _merge
drop _merge
```

* Add subsidized consumption of HUS, health and transport to the respective categories

```
replace v4 = v4 + l_hus
replace v6 = v6 + l_medic + medic
replace v7 = v7 + l_transp
```

```
label variable v2 "Alcoholic beverages and tobacco"
label variable v3 "Clothing and footwear"
label variable v4 "Housing, water, electricity, gas and other fuels"
label variable v5 "Furnishings, household equipment and routine household maintenance"
label variable v6 "Health"
label variable v7 "Transport"
label variable v8 "Communication"
label variable v9 "Recreation and culture"
label variable v10 "Education"
label variable v11 "Restaurants and hotels"
label variable v12 "Miscellaneous goods and services"
label variable other "Others"
```

```
sort id
```

```
save `outfile', replace
```

```
*/ v4 does not have any rent or durable information, but includes the value of lgoti
su v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 rent rentp durables uvalue other l_hus l_medic l_transp lgoti
[aw= kvzv]
```

```
log close
```

```
Pline.do
```

```
version 8
clear
set mem 200m
set more off
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus"
```

```

capture log close
log using pline.log, replace

* Calculating total calorie requirements per hhold
use roster, clear
    recode mabs 90/99=90
    generate pr = cond(mabs==.,1,1-mabs/90)
    generate adlts = 1 if (age>=16 & age<60 & female==0) | (age>=16 & age<55 & female==1)
    generate chldrn = 1 if age<16
    generate req = 2730 if age>=16 & age<60 & female==0
    replace req = 2100 if age>=16 & age<55 & female==1
    replace req = 2000 if (age>=60 & female==0) | (age>=55 & female==1)
    replace req = 797 if age<1
    replace req = 1610 if age>=1 & age<7
    replace req = 2360 if age>=7 & age<16
    replace req = req * pr
collapse (sum) req adlts chldrn, by(id)
sort id
save req, replace

use s0000c, clear
    generate ter1 = a001ter + 1100
    generate double id = a001ter * 100000000 + a002nast * 10000 + a005nom /* generate the id variable
*/
    generate region = a001ter
    generate rural = a003pt == 8
keep ter1 region rural id
sort ter1
merge ter1 using reg_defs
    tabulate _merge
    drop _merge
sort id
save defs, replace

use cons0, clear
sort id
merge id using req
    tabulate _merge
    drop _merge
sort id
merge id using food3
    tabulate _merge
    drop _merge
sort id
merge id using defs
    tabulate _merge
    drop _merge
sort region
merge region using pindex
    tabulate _merge
    drop _merge

```

```

generate v1 = cond(food !=., food * 30 / 14, 0)
mvencode alcohol eatout v11 v2, mv(0) override
replace v2 = alcohol * 30 / 14 + v2
replace v11 = eatout * 30 / 14 + v11

forvalues i =1/2 {
    generate npc_`i' = v`i' / ehsize
} /* end forvalues */
forvalues i =3/12 {
    generate npc_`i' = v`i' / hsize
} /* end forvalues */
label variable npc_1 "Food and non-alcoholic beverages"
label variable npc_2 "Alcoholic beverages and tobacco"
label variable npc_3 "Clothing and footwear"
label variable npc_4 "Housing, water, electricity, gas and other fuels"
label variable npc_5 "Furnishings, household equipment and routine household maintenance"
label variable npc_6 "Health"
label variable npc_7 "Transport"
label variable npc_8 "Communication"
label variable npc_9 "Recreation and culture"
label variable npc_10 "Education"
label variable npc_11 "Restaurants and hotels"
label variable npc_12 "Miscellaneous goods and services"

* generate other per capita components for alternative consumption aggregates
generate npc_uvalue = uvalue/hsize
generate npc_drbles = durables/hsize
generate npc_rent = rent/hsize
generate npc_rentp = rentp/hsize
generate npc_lgoti = lgoti/hsize
generate npc_4l = npc_4 - l_hus/hsize
generate npc_6l = npc_6 - l_medic/hsize
generate npc_7l = npc_7 - l_transp/hsize

egen npc_cons = rsum (npc_1 npc_2 npc_3 npc_4 npc_5 npc_6 npc_7 npc_8 npc_9 npc_10 npc_11
npc_12) /* no rent no durables */
egen test = rsum(npc_cons rent durables uvalue)
drop if test == 0
drop test

* express all consumption groups in real terms; deflate consumption with regional food and non-food
indices

* adjust food consumption by regional food prices (pindex_r / national , pindex_rr /rural and pindex_ru /
urban)
gen pc_1 = npc_1/pindex_r /* food, real pc expenditure, using national food deflator NOBUS 2003*/
gen pc_1r = npc_1/pindex_rr if rural==1
replace pc_1r = npc_1/pindex_ru if rural==0

```

```

* adjust all else by regional non food price index HBS 2002
forvalues i=2/12 {
    gen pc_`i' = npc_`i'/def_nfood
}
local list "pc_uvalue pc_drbles pc_lgoti pc_4l pc_6l pc_7l"
foreach i of local list {
    gen `i' =n`i'/def_nfood
}
* rent values have been already deflated with the housing price index in housing.do
gen pc_rent = npc_rent
gen pc_rentp = npc_rentp

* estimate the basic consumption indicator, without rent or durables
egen pc_cons = rsum (pc_1 pc_2 pc_3 pc_4 pc_5 pc_6 pc_7 pc_8 pc_9 pc_10 pc_11 pc_12)

* estimate alternative per capita consumption indicators, for the simulation

* gold standard: with uvalue and rent (paid or imputed)
egen pc_cons0 = rsum(pc_cons pc_uvalue pc_rent)

* comparators: diferent treatment of durables, but with imputed rent
egen pc_cons1 = rsum(pc_cons pc_drbles pc_rent) /* with durables purchased in Q1 2003 */
egen pc_cons2 = rsum(pc_cons pc_rent) /* without any durable information */

* comparators: different treatment of rent, but with uvalue of durables
egen pc_cons3 = rsum(pc_cons pc_uvalue) /* pc consumption without rent */
egen pc_cons4 = rsum(pc_cons pc_uvalue pc_rentp) /* pc consumption with paid rent only */

* real pc consumption, adjusted for rural/urban price dif
gen pc_cons5 = pc_cons0 - pc_1 +pc_1r

* comparators: not including the value of subsidized received by privileged or poor households
gen pc_cons6 = pc_cons0 - pc_lgoti

* generate the HH specific food poverty line
generate fline = req / 1000 * (8.2 * 1.1311) * 30 / ehsize /* 1.1311 is price growth between 2q03 and 2002 */
generate foodwel = (pc_1) * 100 / fline
drop if fline==0 | fline==.

* estimate the non-food components of the poverty line - ignoring rural-urban food price dif
local varlist "pc_3 pc_4 pc_4l pc_5 pc_6 pc_6l"
foreach var in `varlist' {
    summarize `var' [aw = kvzv * hsize] if foodwel>=95 & foodwel<=105 /* obtain unadjusted MNFPLs */
    generate MNFPL_`var' = r(mean)
}
local varlist "pc_7 pc_7l pc_8"
foreach var in `varlist' {
    summarize `var' [aw = kvzv * hsize] if foodwel>=95 & foodwel<=105 & adlts>0/* obtain unadjusted MNFPLs */
}

```

```

        generate MNFPL_`var' = r(mean)
    }
summarize pc_10 [aw = kvzv * hsize] if foodwel>=95 & foodwel<=105 & chldrn>0/* obtain unadjusted
MNFPLs */
generate MNFPL_pc_10 = r(mean)

gen NFPL3 = MNFPL_pc_3/(hsize^.1)
summarize NFPL3 [aw = kvzv * hsize]
gen pl_pc_3 = NFPL3*MNFPL_pc_3/r(mean)

gen NFPL4 = MNFPL_pc_4/hsize
summarize NFPL4 [aw = kvzv * hsize]
gen pl_pc_4 = NFPL4*MNFPL_pc_4/r(mean)

gen NFPL4l = MNFPL_pc_4l/hsize
summarize NFPL4l [aw = kvzv * hsize]
gen pl_pc_4l = NFPL4l*MNFPL_pc_4l/r(mean)

gen NFPL5 = MNFPL_pc_5/hsize
summarize NFPL5 [aw = kvzv * hsize]
gen pl_pc_5 = NFPL5*MNFPL_pc_5/r(mean)

gen NFPL6 = MNFPL_pc_6
summarize NFPL6 [aw = kvzv * hsize]
gen pl_pc_6 = NFPL6*MNFPL_pc_6/r(mean)

gen NFPL6l = MNFPL_pc_6l
summarize NFPL6l [aw = kvzv * hsize]
gen pl_pc_6l = NFPL6l*MNFPL_pc_6l/r(mean)

gen NFPL7 = MNFPL_pc_7/(hsize^(1-adlts/hsize)) if adlts>0
summarize NFPL7 [aw = kvzv * hsize] if adlts>0
gen pl_pc_7 = NFPL7*MNFPL_pc_7/r(mean) if adlts>0
recode pl_pc_7 . = 0

gen NFPL7l = MNFPL_pc_7l/(hsize^(1-adlts/hsize)) if adlts>0
summarize NFPL7l [aw = kvzv * hsize] if adlts>0
gen pl_pc_7l = NFPL7l*MNFPL_pc_7l/r(mean) if adlts>0
recode pl_pc_7l . = 0

gen NFPL8 = MNFPL_pc_8/(hsize^(1-adlts/hsize)) if adlts>0
summarize NFPL8 [aw = kvzv * hsize] if adlts>0
gen pl_pc_8 = NFPL8*MNFPL_pc_8/r(mean) if adlts>0
recode pl_pc_8 . = 0

gen NFPL10 = MNFPL_pc_10/(hsize^(1-chldrn/hsize)) if chldrn>0
summarize NFPL10 [aw = kvzv * hsize] if chldrn>0
gen pl_pc_10 = NFPL10*MNFPL_pc_10/r(mean) if chldrn>0
recode pl_pc_10 . = 0

summarize pl_* [aw = kvzv * hsize]

```

```

egen pline = rsum(fline pl_pc_3 pl_pc_4 pl_pc_5 pl_pc_6 pl_pc_7 pl_pc_8 pl_pc_10)
egen plinel = rsum(fline pl_pc_3 pl_pc_4l pl_pc_5 pl_pc_6l pl_pc_7l pl_pc_8 pl_pc_10) /* without lgoti
*/

* ----- Pline component for uvalue
summarize pc_uvalue [aw = kvzv * hsize] if foodwel>=95 & foodwel<=105
generate puvalue = r(mean)

* ----- Pline component for durables
summarize pc_drbls [aw = kvzv * hsize] if foodwel>=95 & foodwel<=105
generate pdrbls = r(mean)

* ----- Pline component for rent
summarize pc_rent [aw = kvzv * hsize] if foodwel>=95 & foodwel<=105
generate prent = r(mean)

* ----- Pline component for rent paid (rentp)
summarize pc_rentp [aw = kvzv * hsize] if foodwel>=95 & foodwel<=105
generate prentp = r(mean)

* generating the set of poverty lines
egen pline0 = rsum(pline puvalue prent)
egen pline1 = rsum(pline pdrbls prent)
egen pline2 = rsum(pline prent)
egen pline3 = rsum(pline puvalue)
egen pline4 = rsum(pline puvalue prentp)
egen pline6 = rsum(pline1 puvalue prent) /* without lgoti */

generate z = cond(pc_cons<pline,100,0)
generate z0 = cond(pc_cons0<pline0,100,0)
generate z1 = cond(pc_cons1<pline1,100,0)
generate z2 = cond(pc_cons2<pline2,100,0)
generate z3 = cond(pc_cons3<pline3,100,0)
generate z4 = cond(pc_cons4<pline4,100,0)
generate z5 = cond(pc_cons5<pline0,100,0)
generate z6 = cond(pc_cons6<pline6,100,0)

summarize pc* fline pl* z* [aw= kvzv * hsize]
keep id kvzv adlts chldrn hsize ehsize npc* pc* fline pline* z* def* pl* req puvalue prent prentp pdrbls
order id kvzv adlts chldrn hsize ehsize def* pc_1 pc_2

* gen w = kvzv*hsize
xtile dec =pc_cons / pline [pw=kvzv*hsize], nq(10)
xtile qui =pc_cons / pline [pw=kvzv*hsize], nq(5)
xtile dec0=pc_cons0 / pline0 [pw=kvzv*hsize], nq(10)
xtile qui0=pc_cons0 / pline0 [pw=kvzv*hsize], nq(5)
xtile dec1=pc_cons1 / pline1 [pw=kvzv*hsize], nq(10)
xtile qui1=pc_cons1 / pline1 [pw=kvzv*hsize], nq(5)
xtile dec2=pc_cons2 / pline2 [pw=kvzv*hsize], nq(10)
xtile qui2=pc_cons2 / pline2 [pw=kvzv*hsize], nq(5)
xtile dec3=pc_cons3 / pline3 [pw=kvzv*hsize], nq(10)

```

```

xtile qui3=pc_cons3 / pline3 [pw=kvzv*hsize], nq(5)
xtile dec4=pc_cons4 / pline4 [pw=kvzv*hsize], nq(10)
xtile qui4=pc_cons4 / pline4 [pw=kvzv*hsize], nq(5)
xtile dec5=pc_cons5 / pline0 [pw=kvzv*hsize], nq(10)
xtile qui5=pc_cons5 / pline0 [pw=kvzv*hsize], nq(5)
xtile dec6=pc_cons6 / pline6 [pw=kvzv*hsize], nq(10)
xtile qui6=pc_cons6 / pline6 [pw=kvzv*hsize], nq(5)

```

* estimate per capita consumption in real terms, to generate a relative pov line based on it

```
su pc_cons0 [aw = kvzv * hsize]
```

```
gen rline = .5*r(mean)
```

```
xtile dec7=pc_cons0 / rline [pw=kvzv*hsize], nq(10)
```

```
xtile qui7=pc_cons0 / rline [pw=kvzv*hsize], nq(5)
```

```
sort id
```

```
save cons, replace
```

* average per capita consumption, by commodity group

```
su pc_* [aw= kvzv* hsize]
```

* average per capita poverty line, by commodity group

```
su fline pl_pc_* pline* [aw= kvzv* hsize]
```

```
log close
```

Master_HH.do

* Master HH File *

```
version 8
```

```
clear all
```

```
set mem 200m
```

```
capture log close
```

```
cd "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\"
```

```
log using master.log, replace
```

```
tempfile t1 t2 t3 t4
```

```
use roster, clear
```

```
keep id w pid female age_i
```

```
sort id pid
```

```
save `t1', replace
```

```
use cons, clear
```

```
keep id kvzv z* pc_cons* fline pline* dec* qui* hsize ehsize chldrn rline
```

```
sort id
```

```
save `t2', replace
```

```
use s0000c, clear
```

* generate the id variable

```
gen double id = a001ter*100000000+ a002nast*10000+ a005nom
```

* keep id and regional variables

```
egen psu = group(kvzv)
```

```

gen strata = a003pt
gen region = a001ter
order id psu kvzv
keep id psu strata region
sort id
save `t3', replace

use l0000c, clear
* generate the id variable
gen double id = a001ter*100000000+ a002nast*10000+ a005nom
ren a007id pid
* keep education, employment, and sector of activity
keep id pid a1b01 a1b07 a1b14 a1b15 a1b16 a2v01 a2v02 a2v03 a2v05 a2v23 a2v24 a2v28 a2v29 a2v30
a2v31 a2v32 a2v33 a2v34

ren a1b01 rel_head
ren a1b07 educat
ren a1b14 scholarship
ren a1b15 sch_eligible
ren a1b16 sch_received
gen employment = 1 & a2v01!=.
recode employment 1=2 if a2v23!=.
recode employment 2=3 if a2v24>0 & a2v24<12
lab var employment Employment
lab def empl 0 Children 1 Employed 2 Unemployed 3 Inactive
lab val empl empl
drop a2v01 a2v02 a2v03 a2v23
ren a2v05 employed
lab var employed Employed
ren a2v24 inactive
lab var inactive Inactive
ren a2v28 U_registered
ren a2v29 UB_recipient
ren a2v30 UB_months
ren a2v31 UB_eligible
ren a2v32 UB_last3months
ren a2v33 UB_owed
ren a2v34 UB_owed_months
sort id pid
save `t4', replace

use `t1', clear
merge id using `t2'
keep if _==3
drop _
sort id
merge id using `t3'
keep if _==3
drop _
sort id pid
merge id pid using `t4'

```

```

keep if _==3
drop _

ren kvzv whh
order id pid whh strata psu region z qui dec pc_cons fline pline female age_i hsize ehsize chldrn
gen wr = pc_cons/pline
gen wr0 = pc_cons0/pline0
gen wr1 = pc_cons1/pline1
gen wr2 = pc_cons2/pline2
gen wr3 = pc_cons3/pline3
gen wr4 = pc_cons4/pline4
gen wr5 = pc_cons5/pline0
gen wr6 = pc_cons6/pline6
gen wrd = pc_cons0/rline

lab var whh Weights_HH
lab var w Weights_ind
lab var strata Strata
lab var psu PSU
lab var region Region
lab var z Poor
lab var wr Welfare_Ratio
lab var pc_cons PC_Consumption
lab var fline Food_Line
lab var pline Poverty_Line
lab var chldrn No_Children
svyset [pw=whh], strata(strata) psu(psu)
sort id pid
save MasterInd, replace
keep if rel_head==0

/***** getting rid of duplicates if more then one person had rel_head==0 in the hhold *****/
egen d = count(pid), by(id)
drop if d>1 & pid>1
/*****
*****/

drop pid rel_head d
drop if age<16
ren age agehh
drop scholarship sch_eligible sch_received UB_months UB_eligible UB_last3months UB_owed
UB_owed_months
do FRegion_labels.do
sort id
save MasterHH, replace

use MasterInd, clear
recode sch_received -7/0 = 0
recode UB_last3months -9/0 = 0
collapse (sum) scholar = sch_received UB = UB_last3months, by(id)
replace scholar = scholar /3

```

```

replace UB = UB/3
sort id
merge id using MasterHH
drop if _<3
drop _
lab var scholar Scholarship
lab var UB UnemploymentB
lab var U_registered U_registered
lab var UB_recipient UB_Recipient
sort id
save MasterHH, replace

```

```

su z* [aw=pw]
log close

```

MasterHH_short.do

```

*****
* Create MasterHH_short *
*****

```

```

use "C:\Documents and Settings\wb89165\My Documents\Russia\Nobus\MasterHH.dta", clear
drop scholar UB z qui dec pc_cons pc_cons1 pc_cons2 pc_cons3 pc_cons4 pc_cons5 pc_cons6 /*
    */ pline1 pline1 pline2 pline3 pline4 pline6 z1 z2 z3 z4 z5 z6 dec1 qui1 dec2 qui2 dec3 qui3 /*
    */ dec4 qui4 dec5 qui5 dec6 qui6 rline dec7 qui7 U_registered UB_recipient w wr wr1 wr2 wr3
wr4 wr6 wrd
lab var pw Weights_Population
lab var pc_cons0 "PC_consumption, constant PPP"
lab var pline0 "PC Poverty Line, constant PPP"
lab var z0 "Poverty status: Poor = 100, Non-poor = 0"
lab var regionr "Regions, only those representative"
lab var fregion "Federal regions"
lab var wr0 "Welfare ratio: ratio of PC cons/PC pov line"
lab var educat "Education status of HH head"
lab var female "Female HH Head"
lab var agehh "Age in years, HH Head:
lab var agehh "Age in years, HH Head"
lab var employed "Employment status, HH Head"
lab var inactive "Inactivity status, HH Head"
lab var employment "Labor market status, HH Head"
order id whh pw strata psu z0 pc_cons0 fline pline pline0 wr0 qui0 dec0 region regionr /*
    */ fregion hsize ehsize chldrn female agehh educat employment employed inactive
compress
save MasterHH_short, replace

```

FRegion_labels.do

#delimiter ;

lab def region

1 "Altaiskiy krai"
3 "Krasnodarskiy krai"
4 "Krasnoyarskiy krai"
5 "Primorskiy krai"
7 "Stavropolskiy krai"
8 "Habarovskiye krai"
10 "Amurskaya oblast"
11 "Arkhangelskaya oblast"
12 "Astrakhanskaya oblast"
14 "Belgorodskaya oblast"
15 "Bryanskaya oblast"
17 "Vladimirskaya oblast"
18 "Volgogradskaya oblast"
19 "Vologodskaya oblast"
20 "Voronezhskaya oblast"
22 "Nizhegorodskaya oblast"
24 "Ivanovskaya oblast"
25 "Irkutskaya oblast"
26 "Republic Ingushetiya"
27 "Kaliningradskaya oblast"
28 "Tverskaya oblast"
29 "Kaluzhskaya oblast"
30 "Kamchatskaya oblast"
32 "Kemerovskaya oblast"
33 "Kirovskaya oblast"
34 "Kostromskaya oblast"
36 "Samarskaya oblast"
37 "Kurganskaya oblast"
38 "Kurskaya oblast"
40 "St-Petersburg"
41 "Leningradskaya oblast"
42 "Lipetskaya oblast"
44 "Magadanskaya oblast"
45 "Moscow"
46 "Moskovskaya oblast"
47 "Murmanskaya oblast"
49 "Novgorodskaya oblast"
50 "Novosibirskaya oblast"
52 "Omskaya oblast"
53 "Orenburgskaya oblast"
54 "Orlovskaya oblast"
56 "Penzenskaya oblast"
57 "Permskaya oblast"
58 "Pskovskaya oblast"
60 "Rostovskaya oblast"
61 "Ryazanskaya oblast"
63 "Saratovskaya oblast"

64 "Sakhalinskaya oblast"
 65 "Sverdlovskaya oblast"
 66 "Smolenskaya oblast"
 68 "Tambovskaya oblast"
 69 "Tomskaya oblast"
 70 "Tulskaya oblast"
 71 "Tiumenskaya oblast"
 73 "Uliyanovskaya oblast"
 75 "Chelyabinskaya oblast"
 76 "Chitinskaya oblast"
 77 "Chukotskiy AO"
 78 "Yaroslavskaya oblast"
 79 "Republic Adygeya"
 80 "Republic Bashkortostan"
 81 "Republic Buryatiya"
 82 "Republic Dagestan"
 83 "Kabardino-balkarskaya Republic"
 84 "Republic Altai"
 85 "Republic Kalmykiya"
 86 "Republic Kareliya"
 87 "Republic Komi"
 88 "Republic Mariy El"
 89 "Republic Mordovia"
 90 "Republic Severnaya Osetiya"
 91 "Karachaevo-cherkesskaya Republic"
 92 "Republic Tatarstan"
 93 "Republic Tyva"
 94 "Udmurtskaya Republic"
 95 "Republic Hakasiya"
 97 "Chuvashskaya Republic"
 98 "Republic Saha (Yakutiya)"
 99 "Evreiskaya AO"

;

lab val region region;
 #delimit cr

* Generate regionr, a variable only for the representative regions

gen regionr = region

lab val regionr region

recode regionr 1=0 7=0 14=0 17=0 19=0 25/27=0 29=0 38=0 41=0 44=0 53=0 56/57=0 61/63=0 66=0 /*
 */ 69/70=0 73=0 77=0 84/86=0 88=0 90/91=0 93=0 95/97=0 99=0

* Generate fregion, a variable for federal regions

gen fregion = region

recode fregion 1=101 3=103 4=104 5=105 7=107 8=5 10=5 11=2 12=4 14=1 15=1 17=1 18=4 19=2 20=1
 /*

*/ 22=7 24=1 25=3 26=4 27=2 28=1 29=1 30=5 32=3 33=7 34=1 36=7 37=6 38=1 40=2 41=2
 42=1 /*

*/ 44=5 45=1 46=1 47=2 49=2 50=3 52=3 53=7 54=1 56=7 57=7 58=2 60=4 61=1 63=7 64=5
 65=6 /*

```
*/ 66=1 68=1 69=3 70=1 71=6 73=7 75=6 76=3 77=5 78=1 79=4 80=7 81=3 82=4 83=4 84=3
85=4 /*
*/ 86=2 87=2 88=7 89=7 90=4 91=4 92=7 93=3 94=7 95=3 97=7 98=5 99=5
recode fregion 101=3 103=4 104=3 105=5 107=4
lab def fregion 1 Central 2 "North-West" 3 Siberia 4 South 5 "Far-East" 6 Urals 7 Volga
lab val fregion fregion
```