# Input Datasets

There are 6 input datasets:

* Section A: HH Roster
* Section B: Nonfood (and an alternate form 2)
* Section C: Food
* Section D: Durable Goods
* Section E: Housing
* Section F: Food Away from Home

Note that these are all fake, artificially generated datasets, and may have very weird properties depending on how you look at them. They also tend to have a shorter list of items and a longer list of questions than any actual questionnaire would; different methodologies use different subsets of questions.

The following auxiliary datasets are also provided:

* Conversion factors – for each item, for the two nonstandard units, provides the weight in kg per unit.
* CPI – provides monthly headline CPI values for the period of interest.
* Food composition – calories per 100g for each food item

# Poverty Estimation Programs

Poverty estimation is done via a series of 9 programs:

**0 initialize:** sets global macros for file locations and parameters / options.

**1 hh basics:** constructs hh-level variables like hh size and equivalency (**temp\hh\_char.dta**) and saves PSU-level weight data (**temp\weights.dta**).

**2 food and prices:** Compiles price data for use in valuing own production, constructing deflators and constructing the poverty line. Value of food consumption of each item is constructed (**temp\food.dta**).

There are the following fragments that can be called from this program, assuming different formats of the data, and implementing different methodologies. The choices are:

1. *Classic* (quantity consumed from purchases and total cost of purchases over recall period) vs *LSMS* (quantity and cost of last single purchase) format of the data.
2. Use conversion factors to put everything in *kg* or construct separate prices for each *unit*
3. Construct prices down to *cluster* level or at a higher *domain.*
4. If constructing at a higher domain, *deflate* prices using the CPI, or add a time variable such as *quarter* to the construction, or do not account for temporal variation ONLY IF the country has very low inflation and little seasonality or a short fieldwork period.

The fragments of code provided are as below, this is far from an exhaustive list of the reasonable options. Each one constructs a dataset **ph** of local prices.

* prices\_classic\_kg\_cluster.do
* prices\_classic\_unit\_cluster.do
* prices\_lsms\_kg\_cluster.do
* prices\_lsms \_kg\_domain\_quarter.do
* prices\_lsms\_kg\_domain\_deflated.do
* prices\_lsms\_kg\_domain\_none.do

For valuation the fragments of code provided are:

* food\_selfreport.do – assume “classic” format of data, including respondent-estimated value of consumption of own production. Use this and reported cost of consumption from purchase, no constructed prices used for food consumption value.
* food\_valuation\_classic\_kg\_cluster.do – value own production using **temp\ph\_classic\_kg\_cluster.dta**
* food\_valuation\_classic\_unit\_cluster.do – value own production using **temp\ph\_classic\_psu\_cluster.dta**
* food\_valuation\_lsms\_kg\_cluster.do – value consumption from purchases proportionally to cost of last purchase, and value own production using **temp\ph\_lsms\_kg\_cluster.dta**
* food\_valuation\_lsms\_kg\_domain\_quarter.do – same as above but use **temp\ph\_lsms\_kg\_domain\_quarter.dta**

Note that you can’t use direct self-valuation of consumption from purchases with LSMS style data.

And then it also does FAFH….

**3 nonfood general.do:** compiles nonfood value of consumption, saving **temp\nonfood.dta**

4 health and education.do

**5 durable goods.do:** compiles use value of durable goods, saving **temp\durables.dta**. Code fragments for 4 different methods are provided:

* durables\_maxlife.do – assumes linear depreciation, where the use value is the current value of the good divided by the number of years of life left. The number of years of life is constructed as the 99th percentile of the age of all items by type in the dataset.
* durables\_purchase.do – assumes constant geometric depreciation (usual assumption) and constructs depreciation rates using the purchase price and the current value.
* durables\_replacement.do – assumes constant geometric depreciation and constructs depreciation rates using the replacement cost and the current value.
* durables\_regression.do – assumes constant geometric depreciation and constructs depreciation rates by regressing log of current value on the age of goods in the dataset.

6 housing.do

* self-reported
* hedonic regression, single model
* hedonic regression, multiple models
* hedonic regression combining self-reports

Programs 2 to 6 all compile hh-item level data. These datasets contain the variable *hhid* (and sometimes other key variables like *psu*, *hhweight*, *admin1* etc), *item* which is a unique item code across all the sections (see Table 1 below for coding), *consexp* which is the estimated annual value of consumption of the item, and *source* which is whether the consumption comes from purchases (1), own production (2) or is a constructed use value (3).

Table 1: Summary of Programs 2 to 6

|  |  |  |  |
| --- | --- | --- | --- |
| **program** | **input dataset** | **temp dataset produced** | **values of item** |
| 2 food.do | Section C | food.dta | 1-20, same as initial coding of food items and 21-25, 20 plus initial code of FAFH |
| 3 nonfood general.do | Section B | nonfood.dta | 1001-1010, 1000 + initial code of nonfood items |
| 4 health and education.do | Section A | healtheduc.dta | 1101-1120 |
| 5 durable goods.do | Section D | durables.do | 1201-1210, 1200 + initial code of durable goods |
| 6 housing.do | Section E | housing.do | 1300 |

**7 compile.do** This program compiles the hh-item level datasets constructed by programs 2 to 6, adds COICOP coding and saves a master hh-item level dataset (**dataout\item\_consumption.dta**). It then aggregates to the household level, construct the nominal consumption aggregate and merges in the variables constructed in program 1.

**8 deflators.do** Different methods for deflators are implemented, these include:

* def\_DZ.do – Deaton and Zaidi’s original recommendation, a Paasche index using cluster-level prices (**temp\ph\_lsms\_kg\_cluster.dta**) and hh-level weights (thus, implicitly a joint spatial-temporal index).
* def\_joint\_Paasche\_kg.do – a joint spatial-temporal Paasche index with prices and weights at the domain-quarter level (uses **temp\ ph\_lsms\_kg\_domain\_quarter.dta**). Provides code for both plutocratic and democratic weights.
* def\_joint\_Laspeyres\_kg.do – same as above but constructs a Laspeyres index instead of a Paasche index.
* def\_joint\_Paasche\_Jevons.do – uses price data constructed with separate prices for each unit (**temp\ph\_lsms\_unit\_domain\_quarter.dta**) and combines the different unit prices for each item using a Jevons index (geometric mean of relative prices with no weights), then combines these with item-level weights at the domain-quarter level.
* def\_joint\_Paasche\_modal.do – another approach using price data constructed with separate prices for each unit (**temp\ph\_lsms\_unit\_domain\_quarter.dta**). Use only the prices in the modal (most commonly used) unit for each item at the domain-quarter level to construct the index.
* def\_spatial\_Paasche.do – uses prices that have already been deflated using the CPI (**temp\ph\_lsms\_kg\_domain\_deflated.dta**) to construct just a spatial index at the domain level.
* def\_spatial\_implicit.do – implicit spatial deflators constructed via relative poverty lines at the domain level. Assumes CPI is used for temporal deflation and uses **temp\ph\_lsms\_kg\_domain\_deflated.dta.**

**9 poverty.do** Constructs real welfare aggregate by applying deflators and adjusting for household size and composition. Constructs a cost-of-basic needs poverty line.