
Subject: Re: UNDER FIVE MORTALITY

Posted by [Trevor-DHS](#) on Tue, 29 May 2018 23:20:32 GMT

[View Forum Message](#) <> [Reply to Message](#)

Here is a fairly simplistic piece of code that follows the approach given in the Guide to DHS Statistics, and produces estimates for five five-year periods.

It doesn't produce standard errors or confidence intervals, but allows you to see how the calculations are done:

* Example of early childhood mortality rates calculations

* Trevor Croft, March 9, 2018

* Change directory to the data directory

```
cd "C:\Users\xxxx\Data"
```

* Open DHS dataset - births recode file

```
use v005 v008 b3 b5 b7 using "IABR71FL.DTA", clear
```

* Create variables for time period limits - need to use variables as these change from case to case

```
gen t1 = .
```

```
gen t2 = .
```

* Initialize local variable lists used later

```
local vlist
```

```
local vlist2
```

* Loop through 5-year time periods

```
forvalues period = 0/4 {
```

```
  * Calculate upper limit of time period
```

```
  replace t2 = v008 - 60*`period'
```

```
  * Calculate lower limit of time period
```

```
  replace t1 = t2 - 60
```

```
  * List age group lower limits
```

```
  local agegroups 0 1 3 6 12 24 36 48 60
```

```
  * Turn these into tokens to use for the upper limits of the age groups
```

```
  tokenize `agegroups'
```

```
  * Loop through the age groups
```

```
  foreach age of numlist `agegroups' {
```

```
    * Ignore the 60+ age group - this was just to set the upper limit for the last age group - see a2
```

```
    if (`age' < 60) {
```

```
      * Create local for lower limit of age group - use locals as these are constants
```

```
      local a1 = `age'
```

```
      * Create local for upper limit of age group = the lower limit of the next age group
```

```
      local a2 = `2'
```

```
    * Cohort A numerator
```

```
    gen numA`age'`period' = ((`a1' <= b7 & b7 < `a2') & (t1 - `a2' <= b3 & b3 < t1 - `a1'))
```

```
    * Cohort B numerator
```

```
    gen numB`age'`period' = ((`a1' <= b7 & b7 < `a2') & (t1 - `a1' <= b3 & b3 < t2 - `a2'))
```

```
    * Cohort C numerator
```

```
gen numC`age'`period' = ((`a1' <= b7 & b7 < `a2') & (t2 - `a2' <= b3 & b3 < t2 - `a1'))
```

```
* Cohort A denominator
```

```
gen denA`age'`period' = ( (b5 == 1 | `a1' <= b7) & (t1 - `a2' <= b3 & b3 < t1 - `a1'))
```

```
* Cohort B denominator
```

```
gen denB`age'`period' = ( (b5 == 1 | `a1' <= b7) & (t1 - `a1' <= b3 & b3 < t2 - `a2'))
```

```
* Cohort C denominator
```

```
gen denC`age'`period' = ( (b5 == 1 | `a1' <= b7) & (t2 - `a2' <= b3 & b3 < t2 - `a1'))
```

```
* Count half for deaths for cohort C, except for the last period where all deaths are counted
```

```
local f = 0.5
```

```
if (`period' == 0) {
```

```
    local f = 1
```

```
}
```

```
* Sum numerators from cohorts A, B and C for this case
```

```
gen num`age'`period' = 0.5*numA`age'`period' + numB`age'`period' + numC`age'`period'*`f'
```

```
* Sum denominators from cohorts A, B and C for this case
```

```
gen den`age'`period' = 0.5*denA`age'`period' + denB`age'`period' + denC`age'`period'*0.5
```

```
* Generate list of numerator and denominator variables for period and age for collapse
```

```
command below
```

```
local vlist `vlist' num`age'`period' den`age'`period'
```

```
* Similarly generate list of numerator and denominator variables for period only for reshape
```

```
command below
```

```
if (`period' == 0) {
```

```
    local vlist2 `vlist2' num`age'__ den`age'__
```

```
}
```

```
}
```

```
* Shift the token list to the next age group
```

```
mac shift
```

```
}
```

```
}
```

```
* Sum all numerators and denominators - weighted sum
```

```
collapse (sum) `vlist' [pw=v005/1000000]
```

```
* Add a variable to act as ID for the reshape
```

```
gen x = 0
```

```
* Reshape long by age group
```

```
reshape long `vlist2', i(x) j(period)
```

```
* Drop the underscore (_) on the end of variable names
```

```
rename * _ *
```

```
* Reshape now for periods
```

```
reshape long num den, i(period) j(a1)
```

```
* Drop the x variable as we no longer need it
```

```
drop x
```

```

* Generate the upper bounds of the age groups
gen a2 = a1[_n+1]
replace a2 = 60 if a1 == 48

* Calculate the age group mortality probabilities
gen death = num / den
* Calculate the age group survival probabilities
gen surv = 1 - death

* Generate product of survival probabilities:
gen prodsurv = surv if a1 == 0
replace prodsurv = surv * prodsurv[_n-1] if a1 > 0
* Generate product of survival probabilities for child mortality rate, starting at 12 months
gen prodsurv2 = surv if a1 == 12
replace prodsurv2 = surv * prodsurv2[_n-1] if a1 > 12

* Neonatal mortality rate
gen nmr = 1000*(1-prodsurv) if a2 == 1
* Postneonatal mortality rate (calculated later)
gen pnmr = .
* Infant mortality rate
gen imr = 1000*(1-prodsurv) if a2 == 12
* Child mortality rate
gen cmr = 1000*(1-prodsurv2) if a2 == 60
* Under-five mortality rate
gen u5mr = 1000*(1-prodsurv) if a2 == 60

* Capture just the rates
collapse (min) nmr pnmr imr cmr u5mr, by(period)

* Postneonatal mortality rate = IMR - NMR
replace pnmr = imr - nmr

```

```

* Now see the results
listAnd the results basically match the syncmrates program

```

```

+-----+
| period   nmr   pnmr   imr   cmr   u5mr |
+-----+
1. |    0  29.46365  11.2654  40.72905  9.390652  49.73727 |
2. |    1  31.49295  12.24667  43.73962  11.31612  54.56078 |
3. |    2  33.03296  13.47327  46.50623  12.88736  58.79426 |
4. |    3  36.41945  15.01405  51.4335  16.21401  66.81353 |
5. |    4  40.38089  18.37582  58.75671  19.31465  76.93649 |
+-----+

```