

# Estimation and inference for quantiles and indices of inequality with survey data

**Philippe Van Kerm**

University of Luxembourg and Luxembourg Institute of Socio-Economic  
Research

2018 Nordic and Baltic Stata User Group meeting  
September 12 2018, Oslo

# Stata for income distribution analysis

Many “community-contributed” commands

`findit inequality` or `net search inequality` returns a long list of packages and programs (see `[R] inequality`), such as

- ▶ Percentile shares and Lorenz curves and Gini: `lorenz`, `svylorenz`, `glcurve`, `pshare`, `ldtest`
- ▶ Generalized entropy, Atkinson, Gini indices: `ineqdeco`, `inequal7`, `svygei`, `svyatk`, `sgini`

(Ditto for *poverty* or other distributional features.)

- ▶ DASP: Stata implementation of DAD, very rich collection of functionality on its own (need to register to obtain download key)

# Inference with user-written commands

- ▶ The majority of those commands provide point estimates but no variance calculation
  - ▶ bootstrap or jackknife possible
- ▶ Some commands provide variance calculations and have built-in support for survey design (respect `svyset`)
  - ▶ 'non standard' interaction with the `svy` prefix
  - ▶ compatibility with various generic post-estimation commands (`test`, `nlcom`, etc.)

⇒ Benchmark: Think of how you'd use `mean`

yadap

Inference

Implementation

Illustration

# Yet another distribution analysis package (yadap)

Three estimation commands for *quantiles* and indices of *inequality* and *poverty* from (household- or individual-level) data

- ▶ “estimation commands” means ‘e-class’ commands with complete inference functionality ( $e(b)$  and  $e(V)$ ) to allow interaction with post-estimation `test`, `nlcom`, etc.
- ▶ ... also means integrating with `svy` suite—an essential feature given nature of data on income distribution
- ▶ ... and with the `mi` suite for handling multiply imputed data—an increasingly critical feature

# percentils Estimation of quantiles

`percentils varname [if] [in] [weight] [, percentiles(numlist)]`

- ▶ Hyndman and Fan (1996) definition 4 (and optionally definition 1):

$$P_4(\theta) = y_{k-1} + (y_k - y_{k-1}) \left( \frac{\theta N - N_{k-1}}{w_k} \right)$$

where  $N_k = \sum_{j=1}^n w_j \mathbf{1}(y_j \leq y_k)$  and  $k$  is the smallest integer such that  $N_k \geq \theta N$

# inequally Estimation of (relative) inequality indices

`inequally varname [if] [in] [weight] [,  
selection-of-indices keepnonpositive]`

- ▶ Calculates summary statistics capturing the ‘relative’ dispersion in a vector of incomes/expenditure/wealth (see, e.g., Cowell, 2000, Jenkins and Van Kerm, 2009)
  - ▶ ‘Relative’:  $I(y_1, \dots, y_N) = I(\lambda y_1, \dots, \lambda y_N)$
- ▶ 12 different (families of) measures: (generalized) Gini coefficients, percentile ratios, quantile share ratios, (generalized) entropy measures, coeff of variation, Atkinson measures, SD of logs (and Pietra ratios, de Vergottini, Piesch, Bonferroni, Kakwani indices)

# poverty Estimation of poverty indices

```
poverty varname [if] [in] [weight] [,  
    fracmedian(#) fracmean(#) line(#|varname)  
    selection-of-indices]
```

- ▶ Calculates summary statistics capturing the ‘concentration’ below a certain threshold in a vector of incomes/expenditure/wealth (see, e.g., Zheng, 1997)
- ▶ 7 different (families of) measures: Foster-Greer-Thorbecke, Watts, (generalized) Sen-Shorrocks-Thon index, Chakravarty, Clark-Hemming-Ulph 2 families, and median gaps
- ▶ see Van Kerm (2009) for early version



yadap

**Inference**

Implementation

Illustration

# Two main approaches to inference

Two main approaches for variance estimation, construction of confidence intervals, tests

- ▶ analytic, linearization approaches
- ▶ empirical, resampling-based approaches (jackknife and bootstrap)

# Variance estimation by linearization

## general principles

- ▶  $\theta$  is the statistic of interest, estimated by  $\hat{\theta}$
- ▶ A linearization variable  $Z$  for  $\theta$ , is a linear variable ( $\hat{Z} = \sum_i w_i z_i$ ) such that

$$\text{Var}(\hat{Z}) \approx \text{Var}(\hat{\theta})$$

- ▶ Once we know  $z_k$ , it is easy to estimate  $\text{Var}(\hat{Z})$  and therefore  $\text{Var}(\hat{\theta})$  (it is the variance of a total – methods are well-known to estimate this with various complex survey design)

## Variance estimation with the influence function

An asymptotic approximation of the variance of  $\theta$  is given by (Hampel, 1974)

$$V(\theta) \approx \int IF(y; \theta, F)^2 dF(y)$$

where IF is the influence function.

Practically boils down to estimation of a total (Deville, 1999):

$$V(\hat{\theta}, F) \approx V\left(\sum_{i=1}^n w_i \hat{F}(y_i; \theta, \hat{F})\right)$$

... and formula for the variance of a total known even with complex survey design, e.g.  $V(T) = \frac{1}{n-1} \sum_{i=1}^n (\hat{t}_i - \bar{t})^2$

# The influence function

Expressions for  $IF(y; \theta, F)$  exist (or can be derived) for a wide range of statistics  $\theta$  <sup>1</sup>:

... simple (linear) statistics, e.g., the mean

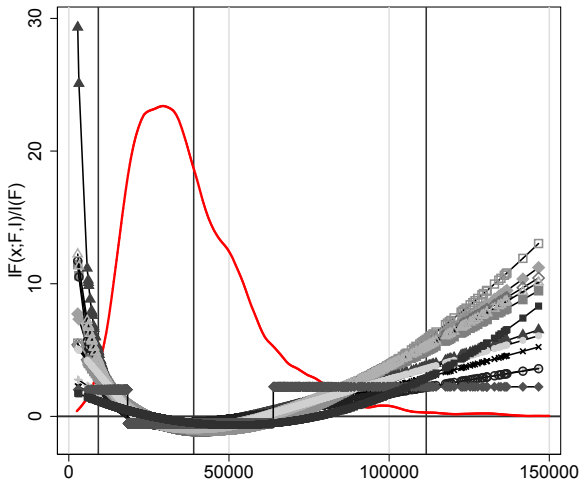
$$IF(y; \mu, F) = y - \mu(F)$$

... and more complex (non linear) statistics, e.g., a quantile

$$IF(y; Q_\theta, F) = \frac{1}{f(Q_\theta(F))} (\theta - I(y \leq Q_\theta(F)))$$

(Estimators of IFs obtained by plugging estimates of the unknowns, e.g.,  $\mu(\hat{F})$ ,  $Q_\theta(\hat{F})$ ,  $\hat{f}$ , etc.)

# Influence functions for inequality measures



yadap

Inference

**Implementation**

Illustration

# Aim

- ▶ calculate point estimates  $e(b)$ : easy
- ▶ calculate linearized standard errors for  $e(V)$
- ▶ work 'seamlessly' with
  - ▶ `test` (and `testnl`, `lincom`, `nlcom`)
  - ▶ `bootstrap`: (and `jackknife`:) prefixes for alternative standard errors and CIs
  - ▶ `svyset` and `svy`: prefix for complex survey data
  - ▶ `mi estimate`: prefix for multiply imputed data
  - ▶ ... and other goodies such as `esttab`, `regsave`, etc.



# Implementation

Easy! Just follow the instructions in [SVY] Variance Estimation and [P] `_robust`...  
(but I must also thank Tech Support and Jeff Pitblado!)

Write two programmes:

- ▶ main program (say `inequally.ado`)
  - ▶ calculates point estimates
  - ▶ does some housekeeping (e.g., initiates unit diagonal covariance matrix—prepare the bread)
  - ▶ calls `_robust` to calculate and fill  $e(V)$
- ▶ secondary ‘prediction’ command (say `inequally_p.ado`)
  - ▶ predict’s value of influence function at the data points used for estimation

(The prediction command is called by `_robust` to fill  $e(V)$ —fills in the sandwich.)

yadap

Inference

Implementation

**Illustration**

# Data

- ▶ The **Luxembourg Income Study**
  - ▶ Household survey data in 48 countries (mostly on income and employment) from early 1980s (most data for 2000s)
- ▶ The **Luxembourg Wealth Study**
  - ▶ Household survey data on household *wealth and assets* in 10 countries
- ▶ Household- and individual-level representative survey data
- ▶ *ex post* harmonization by LIS staff
- ▶ Remote access only (using (slightly restricted) Stata)
- ▶ Illustrations on small training datasets for US



# test and nlcom

```
. test [Perc_50]_cons = 45000
( 1) [Perc_50]_cons = 45000
      F( 1, 1082) = 0.72
      Prob > F = 0.3960

. test [Perc_10]_cons = [Perc_90]_cons
( 1) [Perc_10]_cons - [Perc_90]_cons = 0
      F( 1, 1082) = 962.26
      Prob > F = 0.0000

. nlcom [Perc_90]_cons/[Perc_10]_cons
      _nl_1: [Perc_90]_cons/[Perc_10]_cons
```

dpi	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
_nl_1	7.05641	.3308968	21.33	0.000	6.407864	7.704956

# Weights and clustered standard errors

```
. percentils dpi [pw=ppopwgt] , percentiles(10 50 90)  
Estimation of percentiles                               Number of obs   =   1,083
```

dpi	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
Perc_10 _cons	12972.69	695.7431	18.65	0.000	11607.53	14337.85
Perc_50 _cons	39179.59	1501.47	26.09	0.000	36233.46	42125.71
Perc_90 _cons	93502	3023.02	30.93	0.000	87570.35	99433.65

```
. percentils dpi [pw=ppopwgt] , vce(cluster hid) percentiles(10 50 90)  
Estimation of percentiles                               Number of obs   =   1,083  
                                                       (Std. Err. adjusted for 500 clusters in hid)
```

dpi	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
Perc_10 _cons	12972.69	881.3513	14.72	0.000	11241.07	14704.3
Perc_50 _cons	39179.59	2349.247	16.68	0.000	34563.95	43795.22
Perc_90 _cons	93502	5302.941	17.63	0.000	83083.16	103920.8

# svy prefix

```
. svyset [pw=ppopwgt] , psu(hid) strata(region_c)
```

```
    pweight: ppopwgt
      VCE: linearized
Single unit: missing
  Strata 1: region_c
    SU 1: hid
    FPC 1: <zero>
```

```
. svy: percentiles dpi , percentiles(10 50 90)
(running percentiles on estimation sample)
```

Survey: Estimation of percentiles

```
Number of strata =      14          Number of obs   =      1,083
Number of PSUs   =     500          Population size = 295,516,599
                                          Design df      =       486
                                          F( 0, 486)     =       .
                                          Prob > F       =       .
```

dpi	Linearized			P> t	[95% Conf. Interval]	
	Coef.	Std. Err.	t			
Perc_10						
_cons	12972.69	879.2173	14.75	0.000	11245.15	14700.22
Perc_50						
_cons	39179.59	2348.275	16.68	0.000	34565.56	43793.61
Perc_90						
_cons	93502	5278.47	17.71	0.000	83130.56	103873.4

```
. test [Perc_50]_cons = 45000
```

Adjusted Wald test

```
( 1) [Perc_50]_cons = 45000
      F( 1, 486) =    6.14
      Prob > F =    0.0135
```

# Best practices for testing over subpopulations



# svyest for testing over subpopulations

```
. svy , subpop(if inrange(age,40,60)) : percentils dpi , percentiles(10 50 90)
>
(running percentils on estimation sample)
Survey: Estimation of percentiles
Number of strata = 14
Number of PSUs = 500
Number of obs = 1,083
Population size = 295,516,599
Subpop. no. obs = 340
Subpop. size = 94,092,430.1
Design df = 486
F( 0, 486) = .
Prob > F = .
```

dpi	Linearized		t	P> t	[95% Conf. Interval]	
	Coef.	Std. Err.				
Perc_10 _cons	14798.19	2097.141	7.06	0.000	10677.61	18918.77
Perc_50 _cons	52389	4021.873	13.03	0.000	44486.59	60291.41
Perc_90 _cons	105170	7891.038	13.33	0.000	89665.25	120674.8

```
. estimates store a1
. qui svy , subpop(if !inrange(age,40,60)) : percentils dpi , percentiles(10 50
> 90)
. estimates store a2
```

# suest for testing over subpopulations

```
. suest a1 a2
```

```
Simultaneous survey results for a1, a2
```

```
Number of strata = 14          Number of obs = 1,083
Number of PSUs = 500         Population size = 295,516,599
Design df = 486
```

	Linearized		t	P> t	[95% Conf. Interval]	
	Coef.	Std. Err.				
a1_Perc_10 _cons	14798.19	2097.141	7.06	0.000	10677.61	18918.77
a1_Perc_50 _cons	52389	4021.873	13.03	0.000	44486.59	60291.41
a1_Perc_90 _cons	105170	7891.038	13.33	0.000	89665.25	120674.8
a2_Perc_10 _cons	12837.11	887.3294	14.47	0.000	11093.63	14580.58
a2_Perc_50 _cons	35681	2243.504	15.90	0.000	31272.84	40089.16
a2_Perc_90 _cons	88345.95	5881.93	15.02	0.000	76788.8	99903.11

```
. test [a1_Perc_50]_cons = [a2_Perc_50]_cons
```

```
Adjusted Wald test
```

```
( 1) [a1_Perc_50]_cons - [a2_Perc_50]_cons = 0
F( 1, 486) = 15.47
Prob > F = 0.0001
```

# inequally now...

```
. svy: inequally dpi
(running inequally on estimation sample)
Survey: Estimation of inequality indices
```

```
Number of strata =      14      Number of obs   =      1,069
Number of PSUs   =     491      Population size  = 291,258,463
                                   Design df         =       477
                                   F( 0, 477)          =       .
                                   Prob > F           =       .
```

dpi	Coef.	Linearized Std. Err.	t	P> t	[95% Conf. Interval]	
SGini_2 _cons	.3949245	.0208743	18.92	0.000	.3539076	.4359414
GE_0 _cons	.3057455	.0364963	8.38	0.000	.2340322	.3774589
GE_1 _cons	.2800711	.0454891	6.16	0.000	.1906874	.3694549
A_1 _cons	.263426	.0268822	9.80	0.000	.2106038	.3162482
A_2 _cons	.9733214	.0253033	38.47	0.000	.9236016	1.023041

# inequally again

```
. svy: inequally dpi , cov sdlogs sgini(2 4) ge(-2 1) atk(0.5 2) piesch qsr(9010) perratio(9010) schutz
> mean bonferroni vergottini kakvani
(running inequally on estimation sample)
```

Survey: Estimation of inequality indices

```
Number of strata = 14          Number of obs = 1,069
Number of PSUs  = 491        Population size = 291,258,463
                                Design df = 477
                                F( 0, 477) = .
                                Prob > F = .
```

dpi	Linearized		t	P> t	[95% Conf. Interval]	
	Coef.	Std. Err.				
SGini_2						
_cons	.3949245	.0208743	18.92	0.000	.3539076	.4359414
SGini_4						
_cons	.6091519	.0183418	33.21	0.000	.5731112	.6451926
GE_m2						
_cons	74830.83	74649.12	1.00	0.317	-71850.93	221512.6
GE_1						
_cons	.2800711	.0454891	6.16	0.000	.1906874	.3694549
A_p5						
_cons	.1315336	.0151921	8.66	0.000	.1016818	.1613853
A_2						
_cons	.9733214	.0253033	38.47	0.000	.9236016	1.023041
hRMD						
_cons	.2851081	.0141072	20.21	0.000	.2573882	.312828
SDLog						
_cons	.9138877	.1304969	7.00	0.000	.6574679	1.170308
CoV						
_cons	.9356967	.1734633	5.39	0.000	.59485	1.276543

# inequally ctd.

QSR_9010 _cons	15.16305	2.285513	6.63	0.000	10.67213	19.65396
PeR_9010 _cons	6.624843	.4944219	13.40	0.000	5.653329	7.596357
Kakwani _cons	.1368142	.0133028	10.28	0.000	.1106749	.1629535
Piesch _cons	.3253333	.0218912	14.86	0.000	.2823182	.3683484
Vergo _cons	.8338568	.0981316	8.50	0.000	.6410331	1.02668
Bonfe _cons	.5265866	.0182917	28.79	0.000	.4906444	.5625288
Mean _cons	50381.32	2632.795	19.14	0.000	45208.01	55554.62

# Another nlcom example

The 'Equally-distributed-equivalent'

```
. nlcom (1-[A_p5]_cons)*[Mean]_cons  
      _nl_1: (1-[A_p5]_cons)*[Mean]_cons
```

dp1	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
_nl_1	43754.48	1953.915	22.39	0.000	39924.88	47584.08

# Bootstrap inference

## A studentized bootstrap

Define a small wrapper program to be able to use bootstrap: with sample weights

```
. pr def inequalycalc , rclass
1.     gettoken var therest : 0
2.     qui inequaly 'var' [pw=hpopwgt] 'therest' , gini
3.     return scalar gini = _b[SGini_2:_cons]
4.     mat V = e(V)
5.     return scalar segini = sqrt(V[1,1])
6. end

. inequalycalc dpi if dpi>0
. loc gini = r(gini)
. loc segini = r(segini)
. tempfile bootstat
. bootstrap gini=r(gini) tstat=((r(gini)-'gini')/r(segini)) , ///
>     reps(999) cluster(hid) saving('"'bootstat'"' , replace) : ///
>     inequalycalc dpi if dpi>0
```





# Illustration 2

Now move to data on wealth to illustrate survey bootstrap replications and multiple imputation

```
. use http://www.lisdatacenter.org/wp-content/uploads/us10wh.dta
(us10: version 7.0 23 Mar 2016 10:05)
. keep if inum==1
(400 observations deleted)
. merge 1:1 hid using http://www.lisdatacenter.org/wp-content/uploads/us10wr.dt
> a
```

Result	# of obs.
not matched	0
matched	100 (_merge==3)

# Survey bootstrap weights?

```
. svyset [pw=hpopwgt] , bsrweight(hrwgt1-hrwgt99)
      pweight: hpopwgt
      VCE: linearized
      bsrweight: hrwgt1 hrwgt2 hrwgt3 hrwgt4 hrwgt5 hrwgt6 hrwgt7 hrwgt8 hrwgt9
                hrwgt10 hrwgt11 hrwgt12 hrwgt13 hrwgt14 hrwgt15 hrwgt16
                hrwgt17 hrwgt18 hrwgt19 hrwgt20 hrwgt21 hrwgt22 hrwgt23
                hrwgt24 hrwgt25 hrwgt26 hrwgt27 hrwgt28 hrwgt29 hrwgt30
                hrwgt31 hrwgt32 hrwgt33 hrwgt34 hrwgt35 hrwgt36 hrwgt37
                hrwgt38 hrwgt39 hrwgt40 hrwgt41 hrwgt42 hrwgt43 hrwgt44
                hrwgt45 hrwgt46 hrwgt47 hrwgt48 hrwgt49 hrwgt50 hrwgt51
                hrwgt52 hrwgt53 hrwgt54 hrwgt55 hrwgt56 hrwgt57 hrwgt58
                hrwgt59 hrwgt60 hrwgt61 hrwgt62 hrwgt63 hrwgt64 hrwgt65
                hrwgt66 hrwgt67 hrwgt68 hrwgt69 hrwgt70 hrwgt71 hrwgt72
                hrwgt73 hrwgt74 hrwgt75 hrwgt76 hrwgt77 hrwgt78 hrwgt79
                hrwgt80 hrwgt81 hrwgt82 hrwgt83 hrwgt84 hrwgt85 hrwgt86
                hrwgt87 hrwgt88 hrwgt89 hrwgt90 hrwgt91 hrwgt92 hrwgt93
                hrwgt94 hrwgt95 hrwgt96 hrwgt97 hrwgt98 hrwgt99
Single unit: missing
Strata 1: <one>
SU 1: <observations>
FPC 1: <zero>
```

# Estimation equally easy!

```
. svy : inequaly anw , gini ge(2)
(running inequaly on estimation sample)
Survey: Estimation of inequality indices
```

```
Number of strata = 1
Number of PSUs = 85
Number of obs = 85
Population size = 1,612,590
Design df = 84
F( 0, 84) = .
Prob > F = .
```

anw	Coef.	Linearized Std. Err.	t	P> t	[95% Conf. Interval]	
SGini_2 _cons	.7251663	.0334275	21.69	0.000	.6586921	.7916405
GE_2 _cons	8.759125	4.482467	1.95	0.054	-.154752	17.673

```
. svy bootstrap , nodots : inequaly anw , gini ge(2)
```

```
Survey: Estimation of inequality indices
Number of obs = 85
Population size = 1,612,590
Replications = 99
Wald chi2(0) = .
Prob > chi2 = .
```

anw	Observed Coef.	Bootstrap Std. Err.	z	P> z	Normal-based [95% Conf. Interval]	
SGini_2 _cons	.7251663	.0283444	25.58	0.000	.6696123	.7807203
GE_2 _cons	8.759125	6.912019	1.27	0.205	-4.788183	22.30643



# Multiple imputation: the mi suite

## First some manipulation needed to convert to valid mi data format

```
. use http://www.lisdatacenter.org/wp-content/uploads/us10wh.dta
. * First we need to generate the unimputed version that stata can work with(m=0)
. qui bys hid (anw) : gen imputed = (anw[1] != anw[_N])
. // set as imputed if there are differences in the value of anw across inum
. expand 2 if inum==1 , gen(original)
(100 observations created)
. qui replace inum=0 if inum==1 & original==1
. // set inum to 0 for the newly created obs
. qui replace anw = . if inum==0 & imputed==1
. // set 'unimputed' anw to missing in inum==0
. * Now we can 'import' the data into Stata mi flong format
. mi import flong , m(inum) id(hid) imputed(anw) clear
(89 m=0 obs. now marked as incomplete)
. drop inum original imputed
. mi convert wide , clear
. * and merge replication weights
. qui merge m:1 hid using http://www.lisdatacenter.org/wp-content/uploads/us10wr.dta
```

# Multiple imputation: the mi suite

Then `mi svyset` is specified (note use of both multiple imputation and bootstrap replication weights)

```
. * Now -mi svyset- instead of -svyset-
. mi svyset [pw=hpopwgt] , psu(hid) bsrweight(hrwgt1-hrwgt99) vce(linearized)

      pweight: hpopwgt
             VCE: linearized
      bsrweight: hrwgt1 hrwgt2 hrwgt3 hrwgt4 hrwgt5 hrwgt6 hrwgt7 hrwgt8 hrwgt9
                hrwgt10 hrwgt11 hrwgt12 hrwgt13 hrwgt14 hrwgt15 hrwgt16
                hrwgt17 hrwgt18 hrwgt19 hrwgt20 hrwgt21 hrwgt22 hrwgt23
                hrwgt24 hrwgt25 hrwgt26 hrwgt27 hrwgt28 hrwgt29 hrwgt30
                hrwgt31 hrwgt32 hrwgt33 hrwgt34 hrwgt35 hrwgt36 hrwgt37
                hrwgt38 hrwgt39 hrwgt40 hrwgt41 hrwgt42 hrwgt43 hrwgt44
                hrwgt45 hrwgt46 hrwgt47 hrwgt48 hrwgt49 hrwgt50 hrwgt51
                hrwgt52 hrwgt53 hrwgt54 hrwgt55 hrwgt56 hrwgt57 hrwgt58
                hrwgt59 hrwgt60 hrwgt61 hrwgt62 hrwgt63 hrwgt64 hrwgt65
                hrwgt66 hrwgt67 hrwgt68 hrwgt69 hrwgt70 hrwgt71 hrwgt72
                hrwgt73 hrwgt74 hrwgt75 hrwgt76 hrwgt77 hrwgt78 hrwgt79
                hrwgt80 hrwgt81 hrwgt82 hrwgt83 hrwgt84 hrwgt85 hrwgt86
                hrwgt87 hrwgt88 hrwgt89 hrwgt90 hrwgt91 hrwgt92 hrwgt93
                hrwgt94 hrwgt95 hrwgt96 hrwgt97 hrwgt98 hrwgt99

Single unit: missing
Strata 1: <one>
        SU 1: hid
        FPC 1: <zero>

. // declare boot weights , but set linearized as default vce
```

# Multiple imputation: the mi estimate prefix

The combined mi estimate:svy: is all we need to obtain combined estimates across multiply imputed data

```
. mi estimate , nosmall: ///  
>       svy : inequaly anw , gini keepnonpositive  
Multiple-imputation estimates           Imputations       =           5  
Survey: Estimation of inequality indices Number of obs      =          100  
Number of strata =                      1                 Population size   = 1,876,013  
Number of PSUs  =                      100  
  
Average RVI       =          0.0147  
Largest FMI       =          0.0146  
Complete DF       =           99  
DF:      min      = 18,958.15  
         avg       = 18,958.15  
         max       = 18,958.15  
DF adjustment:   Large sample  
F( 0, .)         = .  
Within VCE type: Linearized             Prob > F           = .
```

anw	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]
_cons	.7752888	.031739	24.43	0.000	.7130775 .8375001

# Multiple imputation: the mi estimate prefix

```
. * finally estimate, NB: I use -noisily- so we can see what is going on:
. mi estimate , vartable noisily nosmall: ///
>      svy : inequaly anw , gini keepnonpositive

(running svy:inequaly on m-1)
(running inequaly on estimation sample)
Survey: Estimation of inequality indices

Number of strata =      1      Number of obs =      100
Number of PSUs  =     100      Population size = 1,876,013
                                   Design df      =      99
                                   F(      0,      99) =      .
                                   Prob > F         =      .
```

	anw	Coef.	Linearized Std. Err.	t	P> t	[95% Conf. Interval]
	_cons	.7704014	.0313707	24.56	0.000	.7081552 .8326476

```
(running svy:inequaly on m-2)
(running inequaly on estimation sample)
Survey: Estimation of inequality indices

Number of strata =      1      Number of obs =      100
Number of PSUs  =     100      Population size = 1,876,013
                                   Design df      =      99
                                   F(      0,      99) =      .
                                   Prob > F         =      .
```

	anw	Coef.	Linearized Std. Err.	t	P> t	[95% Conf. Interval]
	_cons	.7748665	.0314854	24.61	0.000	.7123926 .8373403

```
(running svy:inequaly on m-3)
(running inequaly on estimation sample)
Survey: Estimation of inequality indices

Number of strata =      1      Number of obs =      100
Number of PSUs  =     100      Population size = 1,876,013
                                   Design df      =      99
                                   F(      0,      99) =      .
                                   Prob > F         =      .
```

	anw	Coef.	Linearized Std. Err.	t	P> t	[95% Conf. Interval]
	_cons	.7789197	.0317733	24.51	0.000	.7158745 .8419648

```
(running svy:inequaly on m-4)
(running inequaly on estimation sample)
Survey: Estimation of inequality indices

Number of strata =      1      Number of obs =      100
Number of PSUs  =     100      Population size = 1,876,013
```



# Multiple imputation: the mi estimate prefix

... and works with bootstrap replication weights too!

```
. * works also with bootstrap , but we need to give the vceok option
. mi estimate , vceok nosmall: ///
>       svy bootstrap : inequaly anw , gini keepnonpositive
Multiple-imputation estimates      Imputations      =          5
Survey: Estimation of inequality indices  Number of obs   =         100
                                          Population size = 1,876,013
                                          Average RVI     =    0.0215
                                          Largest FMI     =    0.0212
DF:      min      =  9,064.50
          avg      =  9,064.50
          max      =  9,064.50
DF adjustment:  Large sample
Within VCE type:  Bootstrap      F(  0,      .) = .
                                          Prob > F      = .
```

anw	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
_cons	.7752888	.0263925	29.38	0.000	.7235536	.827024



# Conclusion

- ▶ Ensuring the compatibility of one's estimation command with Stata's internal engines can bring huge benefits both in terms of 'functionality' and 'useability'
- ▶ (and is easier to code!)
- ▶ Illustration with a new set of commands for estimation of percentiles and of inequality and poverty indices
  - ▶ The possibility to account for multiple imputation and complex survey design is a key and important feature!

# References I

- Cowell, F. A. (2000), Measurement of inequality, *in* A. B. Atkinson and F. Bourguignon, eds, 'Handbook of Income Distribution', North Holland, Amsterdam, The Netherlands, pp. 87–166.
- Deville, J.-C. (1999), 'Variance estimation for complex statistics and estimators: Linearization and residual techniques', *Survey Methodology* **25**(2), 193–203.
- Essama-Nssah, B. and Lambert, P. J. (2012), Influence functions for policy impact analysis, *in* J. A. Bishop and R. Salas, eds, 'Inequality, Mobility and Segregation: Essays in Honor of Jacques Silber', Vol. 20 of *Research on Economic Inequality*, Emerald Group Publishing, chapter 6, pp. 135–159.
- Hampel, F. R. (1974), 'The influence curve and its role in robust estimation', *Journal of the American Statistical Association* **69**(346), 383–393.
- Hyndman, R. J. and Fan, Y. (1996), 'Sample quantiles in statistical packages', *The American Statistician* **50**(4), 361–365.
- Jenkins, S. P. and Van Kerm, P. (2009), The measurement of economic inequality, *in* W. Salverda, B. Nolan and T. M. Smeeding, eds, 'Oxford Handbook of Economic Inequality', Oxford University Press, Oxford, chapter 3, pp. 40–67.
- Van Kerm, P. (2009), Computing poverty measures with survey data, German Stata Users' Group Meetings 2009 03, Stata Users Group.
- URL: <https://ideas.repec.org/p/boc/dsug09/03.html>

# References II

Zheng, B. (1997), 'Aggregate poverty measures', *Journal of Economic Surveys* **11**(2), 123–62.