Statistics/Data analysis

```
use "${simulationData}\01_${countryName}_${simulationName}_${dem_inc_SY}.dta" , clea
> r
merge 1:1 hhID memberID using "${simulationData}\06_${countryName}_${simulationName}
> _${mkt_inc_PY}.dta", nogen update replace


* Here the order is opposite to grossing up!

*================================================================================
        * 1. Social Contributions (SIC): SIC rate in baseline is 20%
*================================================================================
gen double SIC = -1 * wage * ${SIC_rate}
gen double wage_PIT = wage + SIC

* dropping variables
drop wage wage_net              // dealing with wage separately since wage_PIT is al
> ready generated above


local PIT_list = subinstr("${market_income}", "wage", "", .) // dropping "wage" sinc
> e wage_PIT is already defined above

* renaming vars to *_PIT and dropping *_net vars
foreach z in `PIT_list' {
        rename `z'  `z'_PIT             // making parallel change to entr_inc etc as
>  wage above to make the coding below consistent,
*       drop  `z'_net   // dropping *_net to avoid confusion since I'll later rename
>  var_PIT to var (as in "rename wage_PIT wage") when I generate back wage_net from
> wage_PIT (after converting the var name from wage_PIT to wage first)
}



*================================================================================
        * 2. Personal Income Tax (PIT)
*================================================================================

/* Adding all incomes that are subjected to PIT */
* putting the list of PIT incomes (wage_PIT etc) into a macro to automate the comman
> d right below it
local pit_inc_list
foreach z in $market_income {
        local pit_inc_list  `pit_inc_list' `z'_PIT
}

global PIT_income_list `pit_inc_list'           // Use var_PIT instead of var

gen double PIT_income_gross = 0
        foreach var in $PIT_income_list {
                replace PIT_income_gross = PIT_income_gross +  `var'  // adding all
> income components that are subjected to PIT
        }


gen double PIT_income_gross_afterExemption = max(0, PIT_income_gross - ${PIT_deducti
> on})  // income before PIT personal exemption
        gen PIT_exempted = PIT_income_gross - PIT_income_gross_afterExemption
        assert PIT_exempted >= 0 & PIT_exempted <= ${PIT_deduction}  // if !missing(
> PIT_exempted)
```

```
/* Recover net wage from gross wage  - this part is automoated by global options in
> dirtax.ado */
dirtax PIT_income_gross_afterExemption, grossinput rates(0 ${PIT_rate_lists}) tholds
> (0 ${PIT_cutoff_lists}) gen(PIT_income_net_afterExemption)  // Recover new wage fr
> om gross wage
        assert PIT_income_net_afterExemption <= PIT_income_gross_afterExemption

gen double PIT = -1 * (PIT_income_gross_afterExemption - PIT_income_net_afterExempti
> on)
gen double PIT_income_net = PIT_income_net_afterExemption + PIT_exempted
        assert (PIT_income_gross_afterExemption - PIT_income_net_afterExemption)>=0


/* We restore net (plus PIT) incomes from net proportionally to their contribution t
> o gross PIT */
foreach z in $PIT_income_list {
        local z = subinstr("`z'", "_PIT", "" , .)       // dropping the letter "_PIT
> " in the var_PIT (eg wage_PIT)
        gen double `z' = `z'_PIT *  PIT_income_net/PIT_income_gross
        drop `z'_PIT            // now we recovered new wage (named "wage") so we do
> n't need gross wage named wage_PIT, so dropping var_PIT
}

des PIT         PIT_income_net  PIT_income_gross, full
sum PIT         PIT_income_net  PIT_income_gross



*================================================================================
        * 3. Consistency check: baseline, original (survey based) income vs simulate
> d net market incomes
*================================================================================

/*
global income_consistency_check = 1
if $income_consistency_check == 1 {
        egen double net_market_income = rowtotal(${market_income} ${SSC} ${direct_ta
> xes}), missing
        assert abs(net_market_income_orig - net_market_income) < 10^(-10)
}
*/

*================================================================================
        * 4. Save it
*================================================================================
keep hhID memberID ${SSC} ${direct_taxes}
order hhID memberID ${SSC} ${direct_taxes}

mvencode ${SSC} ${direct_taxes}, mv(0) override                 // changing missing
> values to zero
isid hhID memberID

des, full
sum
save "${simulationData}\07_${countryName}_${simulationName}_${ssc_dir_tax_PY}.dta",
```