EXAMPLE 6:
WORKING WITH WEIGHTS AND COMPLEX SURVEY DESIGN

EXAMPLE RESEARCH QUESTION(S):

- How does the average pay vary across different countries, sex and ethnic groups in the UK?
- How does remittance behaviour vary by socio-demographic characteristics?

DESCRIPTION: In this example we show how to use weights and sampling design information provided with the data to obtain appropriate population (mean, regression coefficient) estimates and their confidence intervals.

FILES: a_indresp

WAVES: In this example we will use information from multiple waves

NEW COMMANDS:

```
mean, test, tabstat, pweight
svyset, svydes, svy:, estat, proportion
```

6.1 OVERVIEW

[*All Tables and Appendices referred to in text are at the end of this worksheet*]

Stata, like most statistical software, assumes that the sample is a simple random sample and that each sample unit is selected with equal probability and independently of each other. However, most surveys, including Understanding Society, do not fall into this category.

- If sample units are selected with unequal probability by design or if not every selected sample unit responds to the survey, and those included in the sample are systematically different from those not in the sample then population estimates based on this sample will be biased. In such cases weights are to be used to produce unbiased estimates of population parameters.
- If the sample design is not a simple random sample as it is in the case of Understanding Society, then these sampling design features (such as whether this is a clustered and/or stratified sample) need to be considered to produce unbiased estimates of standard errors of the population estimates.

Variables representing weights and sample design are available in the data. Stata's `svy` suite of commands is very convenient for producing unbiased estimates of population parameters. We will discuss this in detail in the analysis section.

During the lecture we have discussed the sample design of Understanding Society and the weights provided with the data. For a quick recap of the sample design and the key sample design variables including primary sampling unit (PSU) and strata see Table 1 and Appendix A. For further details see the Understanding Society User Manual (https://www.understandingsociety.ac.uk/documentation/mainstage).

To choose the correct weights for your analysis, the Weights sections in the User Manual provides detailed tables of options when analysing

- households or enumerated individuals
- adult respondents including proxy
- adult respondents excluding proxy
- the "Extra 5 minutes" sample
- adult respondents who completed the self-completion questionnaire
- youth respondents

There also is information on design weights (for advanced users).

Note that all longitudinal (but not cross-sectional) weights are zero for Temporary Sample Members (TSMs) except for household weights. Also weights which are computed for analysis with a specific sample will be zero for anyone who does not qualify for that sample. For example, **b_indinus_xw** is designed for analysis with adult respondents in Wave 2 of the GPS and EMBS and so will be zero for the BHPS sample and also for proxy respondents in the GPS and EMBS. Similarly, the self-completion longitudinal weights will be zero for TSMs, Proxy respondents and anyone who did not complete the self-completion questionnaire. But additionally, **b_indscbh_xw**, the self-completion weights designed for the BHPS sample will also be zero for GPS and EMBS members and **b_indscus_xw**, designed for the GPS and EMBS will be zero for the BHPS sample members.

6.2 DATA PREPARATION

Examining the data

In this example we will use **a_indresp.dta** file. First open the file and examine the data by looking at the data, variables of interest and their distribution. We have come across different commands that are useful for this purpose: **`summarize, describe, tab`**

As we outlined at the beginning we will be discussing two analyses: one about monthly pay and the other about remittance behaviour. See Table 2 for a complete list of variables useful for this analysis.

This is a good opportunity to see that you understand why some variables have missing values. As you are aware Understanding Society data is provided without any system missing (.) but all missing values are replaced by negative values representing the reason for missing information.

For a distribution of the different types of missing cases for the imputed gross monthly pay variable, **a_paygu_dv**:

```
tab a_paygu_dv if a_paygu_dv < 0
```

See the "Example6.do" file for the syntax to use to identify the reasons for missing values on this variable.

In Understanding Society, the question that asks about remittances is,

> Many people make gifts or send money to people in another country. Did you send or give money to anyone in a country outside the UK in the past 12 months for any of the following reasons?
>
> 1 Repayment of a loan
> 2 Support for family members or friends
> 3 Support for a local community. Please do not include donations to large charities such as Oxfam or Save the Children
> 4 Personal investment or savings, including property
> 5 No money sent/given

The responses were recorded in **a_remit1, a_remit2, a_remit3, a_remit4, a_remit5**. So, the variable that records whether any remittance was made is **a_remit5**:

```
tab a_remit5
```

We see that the remittance question was inapplicable (**a_remit** = -8), that is not asked, of a lot of respondents. Why? As this question is part of the extra 5 minute questions, it should only have been asked of the "Extra 5 minutes" sample (see **Appendix A**).

We can use the flag variable **a_xtra5min_dv** which identifies this sample to check whether those with **a_xtra5min_dv=0** are the ones for whom **a_remit5=-8**.

```
tab a_remit5 a_xtra5min_dv
```

The output of this cross-tabulation shows that only those were in the "Extra 5 minutes" sample were asked this question.

Next we will check whether the distribution of weights is as we would have expected. For that we can use another useful Stata command, **`tabstat`**. It is particularly useful to look at the summary statistics of some variables across different groups. For example, cross-sectional respondent weights, by definition, should be zero for proxy respondents. We should be able to check that by looking at the mean of cross-sectional respondent by interview outcome:

```
tabstat a_indinus_xw a_indpxus_xw a_ind5mus_xw a_indscus_xw, ///
stat(mean min max sd) by(a_ivfio) longstub nototal
```

Similarly **a_ind5mus_xw** should zero for those who were not eligible for the "Extra 5 minutes" questions.

```
tabstat a_ind5mus_xw, stat(mean min max sd) ///
by(a_xtra5min_dv) longstub nototal
```

Which weights to use for our analyses?

By looking at the tables in the Weights section of the Understanding Society User Manual we can conclude that (i) for the analysis of monthly pay, which is available for all adult respondent but not proxy respondents, we should use **a_indinus_xw** and (ii) for the analysis of remittance behaviour, which is an extra five minutes question, we should use **a_ind5mus_xw.**

Next, let us examine some of the sample design features. Remember that the GPS-NI sample is a simple random sample. But Stata cannot estimate standard errors if there are single PSU strata. By design this is the case for GPS-NI sample. You can think of every household in this sample as a PSU. Using that logic, each household in the GPS-NI sample has been assigned a separate pseudo-PSU number to allow computations using Stata. This is not the case for the other countries. You can check that by looking at the mean and standard deviation of the psu and strata variables for each UK country.

```
tabstat a_psu a_strata, stat(mean min max sd) ///
by(a_country) longstub nototal
```

Before we begin our analysis, we need to recode these negative values to system missing as Stata only recognizes system missing as missing values.

```
mvdecode _all, mv(-9/-1)
```

6.3 NOTES ON USING WEIGHTS AND SVY SUITE OF COMMANDS IN STATA

If this is a simple random sample but not all sample units are selected with equal probability, then only weights need to be specified:

```
Stata command [pweight = weight]
```

However, if sample design is more complex like that of Understanding Society then Stata's **svy** suite of commands needs to be used:

```
svyset psu [pweight = weight], strata(strata)
```

where *weight* is the variable representing probability weight, *psu* is the variable representing the primary sampling unit and *strata* is the variable representing the stratification variable. In Understanding Society, these variables are **w_psu** and **w_strata**. See Stata Manual for more explanation. It is a good idea to clear Stata's memory of any existing **svyset** specifications before specifying the sampling design.

```
svyset, clear
svyset psu [pweight = weight], strata(strata)
```

6.4 ANALYSIS: ESTIMATING AVERAGE MONTHLY PAY IN UK

To estimate unweighted mean of gross monthly pay and its standard error without correcting for complex survey design:

```
mean a_paygu_dv
```

To estimate weighted mean of gross monthly pay and its standard error, *without* correcting for complex survey design:

```
mean a_paygu_dv [pweight = a_indinus_xw]
```

Note that if those who are over or under-represented in the sample or those selected with higher or lower selection probabilities are different in terms of gross monthly pay then the weighted estimates will be different from un-weighted estimates.

To estimate weighted mean of gross monthly pay and its standard errors, after correcting for the complex survey design:

```
svyset a_psu [pweight = a_indinus_xw], strata(a_strata)
svy: mean a_paygu_dv
```

You will notice that there are no standard errors! Why? Stata gives the answer in the note below: "Note: missing standard errors because of stratum with single sampling unit." This is not a problem of the sample design but could happen with any data based on a clustered and stratified design. In this case this happens because the analysis uses non-missing values of pay, which results in a sample such that some of the observations belong to a stratum with a single PSU. If we were analysing a different variable this problem may not arise. For example, if we were interested in estimates of mean age, there would be no problem. You can check that.

```
svy: mean a_dvage
```

In this next sub-section we will discuss how to deal with the problem of single PSU strata. We will get back to the discussion of estimating wages across different UK countries and testing their differences after this.

6.5. PROBLEM WITH STRATA WITH A SINGLE PRIMARY SAMPLING UNIT

Pitblado (2009) discusses this problem with Stata and the different methods that could be used to get estimates of standard errors.

Method 1: One method is to simply drop these cases. You can use **`svydes`** to identify any strata with single sampling units. But in some cases, as in our sample, there are no strata with single PSUs but this happens for a specific estimation. To identify such cases just using **`svydes`** will not help. You will need to type another command after the estimation command,

```
svy: mean a_paygu_dv

svydes if e(sample), single
```

This command will list out all such strata. Then copy and list out these strata to drop them. Now if you estimate mean pay you will see that Stata does estimate standard errors.

Method 2: is to combine adjacent strata (see Appendix C on how to do this with our data).

Method 3: Stata provides an adhoc measures to deal with this. For that you will have to first specify the **`svyset`** with its **`singleunit()`** option. There are four different specifications for **`singleunit()`** option: **`missing (default), certainty, scaled, centred`**. The "missing" option simply asks Stata not to produce any standard errors which is also the

default option. That is what we saw above. Read Stata Help Manual for a description of the other options to decide which one is appropriate. So, if you use the following syntax, you will see that the standard errors are computed,

```
svyset a_psu [pweight = a_indinus_xw], ///
strata(a_strata) singleunit(scaled)
svy: mean a_paygu_dv
```

You will see this does not change the estimates for gross monthly pay (in comparison to the weighted estimates without correcting for sample design) but changes the estimates of standard errors.

6.6 ANALYSIS: ESTIMATING AVERAGE MONTHLY PAY ACROSS THE FOUR COUNTRIES/REGIONS WITHIN UK

In this sub-section we will estimate mean pay in the four countries of UK and check if these are different.

```
mean a_paygu_dv, over(a_country)
mean a_paygu_dv [pweight = a_indinus_xw], over(a_country)
svy: mean a_paygu_dv, over(a_country)
```

You will notice that Stata has changed the value label for **a_country** = 4 category from **Northern Ireland** to **_sub_pop_4** for to its internal purposes because the value label includes a space. Had we labelled the Northern Ireland category without a space and then Stata would not have changed it.

We will next test differences in pay across the different countries. For reasons unknown, in this command we will need to use the value labels instead of values of the variable. Remember Stata has transformed the value label of **a_country** = 4 from **Northern Ireland** to **_subpop_4**, so we will have to use the new label.

```
test [a_paygu_dv]England = [a_paygu_dv]Wales = ///
     [a_paygu_dv]Scotland = [a_paygu_dv]_subpop_4

// Note: the syntax for this has changed from Stata 16 onwards
test  a_paygu_dv#1.a_country = ///
      a_paygu_dv#2.a_country = ///
      a_paygu_dv#3.a_country = ///
      a_paygu_dv#4.a_country
```

The result shows that these differences are statistically significant.

If you want to run estimations on a sub-population as a Stata user, your first reaction will be to use "if" statements to select a sub-sample for the analysis. However, if you read Stata's manual for **svy** set of commands, you will find that use of "if" statements to select a sub-sample is not recommended for any analysis using **svy** set of commands. Rather Stata recommends using the **subpop** option. We will now use this method to estimate the average gross monthly pay for each of the four UK countries.

```
svy, subpop(if a_country ==1): mean a_ paygu_dv
svy, subpop(if a_country ==2): mean a_ paygu_dv
svy, subpop(if a_country ==3): mean a_ paygu_dv
svy, subpop(if a_country ==4): mean a_ paygu_dv
```

6.7 ANALYSIS: ESTIMATING DESIGN

A clustered sample generally leads to higher standard errors (of some estimated value) compared to a simple random sample of equal size. The opposite is generally the case for a stratified sample. As standard error is a measure of the precision of an estimate, it is good to know how much precision you gain or lose by using a particular sample design. One way to measure this is by using the **design effect (deff).** It is the ratio of the variance of a statistic based on the actual sample design to the variance of this statistic had the sample design been a SRS (simple random sample) of the same size. In other words, it indicates by how much the variance is inflated or deflated due to the sampling design. **deft** is the square root of **deff**, i.e., it is the ratio of the two standard errors.

```
svy: mean a_paygu_dv
estat effects, deff deft
```

6.8 ANALYSIS: HOW DOES REMITTANCE BEHAVIOUR VARY BY SOCIO-DEMOGRAPHIC CHARACTERISTICS?

In this section we will use weights in multivariate analysis. To illustrate this we will use a specific research question: **How does remittance behaviour vary by socio-demographic characteristics?** Whether a person sends remittance or not is a binary variable and so we will need to use a logit or probit model to estimate how remittance behaviour varies by socio-demographic characteristics. So, we will need to create a 0-1 indicator variable (or dummy variable) that takes on the value 1 if a person sends money and 0 otherwise,

```
generat remit=1 if a_remit5==0
replace remit=0 if a_remit5==1
```

The different socio-demographic characteristics that we want to include in this model are: **age, gender, education, marital status, ethnic group, and UK country of residence.** You may want to add other variables such as number of own children, household income, years since arrival to the UK as these could also influence remittance behaviour. In the following model specifications we use factor variables (i.). If you want to know about the use of factor variables see **Appendix C.**

Unweighted estimates, without accounting for complex survey design

```
logit remit a_age_dv i.a_sex_dv i.a_hiqual_dv i.a_country ///
i.a_mastat_dv i.a_racel_dv
```

Weighted estimates, without accounting for complex survey design

```
logit remit a_age_dv i.a_sex_dv i.a_hiqual_dv i.a_country ///
i.a_mastat_dv i.a_racel_dv [pweight = a_ind5mus_xw]
```

Weighted estimates, accounting for complex survey design

```
svyset, clear
svyset a_psu [pweight = a_ind5mus_xw], strata(a_strata)

svy:  logit remit a_age_dv i.a_sex_dv i.a_hiqual_dv ///
i.a_country i.a_mastat_dv i.a_racel_dv
```

Note while using factor variables directly on the existing categorical variables is quite convenient you may find it more useful to convert these variables into fewer categories that are more sensible. For example, some categories may just have a few cases. As in the above analysis, we found that some categories did not include any cases and were dropped from the analysis. Also, we may only be interested in knowing whether a person is living with a

spouse or partner, so we should convert the marital status variable into a 0-1 indicator variable which takes on a value of one if the person is married, in a civil partnership or in living with someone as a cohabiting couple.

6.9 USING LONGITUDINAL WEIGHTS

In this section we will show you how to use longitudinal weights for producing estimates in longitudinal analysis. In this exercise we will estimate the average gross monthly wage of employed individuals across 3 years, i.e., across Waves 1, 2 and 3. As this information is asked of adult respondents we will be using the files **a_indresp b_indresp** and **c_indresp.**

Which weights should we use? We will use information provided by adult respondents but excluding proxies. We will not be using any of their responses to the self-completion questionnaire. The purpose of longitudinal weights is to make the estimates representative of the core sample selected in the first Wave and so the longitudinal weights for each wave is a product of the initial weight and all wave-on-wave non-response adjustment weights (for further details see the User Manual). So, the longitudinal weights are non-zero for respondents who have continually responded from the first wave. If you are conducting longitudinal analyses using N waves of data then use the longitudinal weight from the N^{th} wave. So, the appropriate longitudinal weight for this exercise is **c_indinus_lw**.

Create a long format file using **a_indresp b_indresp** and **c_indresp** (as shown in Example 3), keeping the variables which we will need for this exercise, that is, pidp, w_paygu_dv,, w_strata, w_psu and w_indinus_lw. Even though the strata and psu variables are provided with each wave specific file with a wave prefix, as these are sample design variables, their values do not change across the waves for any person.

```
foreach w in a b c {
   use pidp `w'_paygu_dv `w'_strata `w'_psu `w'_indinus_*w ///
   using "$inpath/ukhls/`w'_indresp", clear
   rename `w'_* *
   generat wave= strops("abc","`w'")
   save `w', replace
}
use a, clear
foreach w in b c {
   append using `w'
}
sort pidp wave
mvdecode _all, mv(-9/-1)
save long.dta, replace
```

Below is an example to illustrate longitudinal weights, psu and strata across three waves for individuals with different response patterns. As you can see from this table,

- **psu** and **strata** values remain the same across all three waves for everyone
- Person 1 responded in all 3 waves and so has a positive longitudinal weight in both Waves 2 and 3.
- Person 2 has missed the second wave and so the value of this weight for Wave 3 is zero
- Person 3 joined the household in the second wave and so has zero longitudinal weights

- Person 4 responded in the first two waves but not in the third wave and so has a longitudinal weight for Wave 2

Pidp	wave	paygu_dv	indinus_lw	psu	strata
1	1	1000	.	67	14
1	2	1200	0.9934	67	14
1	3	1200	0.9092	67	14
2	1	500	.	2002	2555
2	3	600	0	2002	2555
3	2	1650	0	1555	1445
3	3	1700	0	1555	1445
4	1	2000	.	560	3003
4	2	2500	0.8745	560	3003

The weight from the 3rd wave can be added to previous wave rows for each person, but a simpler method is to simply merge the longitudinal weight variable from Wave 3, **c_indinus_lw**, to the long-format file created with three waves of data, **long.dta**. See below on how to do this.

```
use long, clear
merge m:1 pidp using "$inpath/ukhls/c_indresp", ///
  keepusing(c_indinus_lw)
```

This is how the data will now look.

pidp	wave	paygu_dv	indinus_lw	psu	strata	c_indinus_lw
1	1	1000	.	67	14	0.9092
1	2	1200	0.9934	67	14	0.9092
1	3	1200	0.9092	67	14	0.9092
2	1	500	.	2002	2555	0
2	3	600	0	2002	2555	0
3	2	1650	0	1555	1445	0
3	3	1700	0	1555	1445	0
4	1	2000	.	560	3003	.
4	2	2500	0.8745	560	3003	.

This newly attached weight variables, **c_indinus_lw**, can be used just like any weight variable in the `svyset` command or when using the `pweight` option.

For this exercise we want to estimate 3-year average wage for adults in UK. First, we will create the 3-year average wage.

```
bys pidp: egen avg_wage=mean(paygu_dv)
```

After this is created only need to keep observations for the 3rd wave.

```
keep if wave==3
```

The unweighted estimate of the average 3-year average wage for the UK can be computed,

```
mean avg_wage
```

To produce the unbiased population estimate of the 3-year average wage with correct standard errors, specify `svyset` with **`c_indinus_lw`**, **`psu`** and **`strata`**

```
svyset, clear
svyset psu [pweight = c_indinus_lw], strata(strata) ///
singleunit(scaled)

svy: mean avg_wage
```

This exercise can be repeated for a wide-format file. First create the file, then the 3-year average wage, then specify the svyset command and finally estimate the mean 3-year average wage. You can compare and see that the results from the two approaches are identical.

```
foreach w in a b c {
  use pidp `w'_paygu_dv `w'_strata `w'_psu `w'_indinus_*w ///
  using "$inpath/ukhls/`w'_indresp", clear
  save `w', replace
}
use a, clear
foreach w in b c {
  merge 1:1 pidp using `w'
}
mvdecode _all, mv(-9/-1)
save wide.dta, replace
```

pidp	a_paygu_dv	b_paygu_dv	c_paygu_dv	a_indinus_lw	b_indinus_lw	c_indinus_lw	a_psu	b_psu	c_psu	a_strata	b_strata	c_strata
1	1000	1200	1200	.	0.9934	0.9092	67	67	67	14	14	14
2	500		600	.		0	2002		2002	2555		2555
3		1650	1700		0	0		1555	1555		1445	1445
4	2000	2500		.	0.8745		560	560		3003	3003	

Create the 3-year average wage

```
egen avg_wage=rowmean(a_paygu_dv b_paygu_dv c_paygu_dv)
```

After this is created only keep observations for the 3rd wave.

```
keep if c_indinus_lw<.
```

The unweighted estimate of the average 3-year average wage for the UK can be computed,

```
mean avg_wage
```

To produce the unbiased population estimate of the 3-year average wage with correct standard errors, specify `svyset` with **`c_indinus_lw`**, **`psu`** and **`strata`**

```
svyset, clear
svyset c_psu [pweight = c_indinus_lw], strata(c_strata) singleunit(scaled)
svy: mean avg_wage
```

References

Pitblado, J. 2009. Survey Data Analysis in Stata. *2009 Canadian Stata Users Group Meeting*. http://www.stata.com/meeting/canada09/ca09_pitblado_handout.pdf

Wooldridge, J., Haider, S., Solon, G. 2013. What are we weighting for? *NBER working paper* No. 18859

Table 1: Description of key survey design variables in Understanding Society

Variable	Description	Data file available in
w_psu	Primary sampling unit	All files
w_strata	Strata	All files
w_hhorig	sample indicator	All files
w_lda	Low ethnic minority concentration area indicator	All files
a_month	Monthly sample indicator	All files
a_ivfio	Individual interview outcome	All individual level files

Table 2: Variables to be used in the analyses

Variable description	Variable name
Sex	**a_sex_dv**
Age	**a_age_dv**
De facto marital status	**a_mastat_dv**
Ethnic group	**a_racel_dv**
Region of residence	**a_gor_dv**
Educational qualification	**a_hiqual_dv**
Usual gross monthly pay	**a_paygu_dv**
Reasons for sending or giving money to people in another country (remittance)	
For repayment of loan	**a_remit1**
To support family members or friends	**a_remit2**
To support a local community	**a_remit3**
For personal investments or savings including property	**a_remit4**
No money sent/given	**a_remit5**[#]

[#] This is the relevant variable for our analysis as it is an indicator of remittance (i.e., whether any money was sent or given to anyone in another country)

Appendix A

Understanding Society sample design

- General Population Sample (GPS) has two components: GPS-GB and GPS-NI
  - GPS-GB: A clustered and stratified sample drawn from Great Britain where each unit had an equal selection probability.
  - GPS-NI: A simple random sample from Northern Ireland where sampling units had approximately twice the selection probability as the units in GPS-GB.
- The Ethnic Minority Boost Sample (EMBS): A clustered, stratified sample drawn from high ethnic minority concentration areas in Great Britain. Households at selected addresses were screened in to include households where at least one person was from an ethnic minority group, or their parents or grandparents were.
- The British Household Panel Survey (BHPS) sample became part of the Understanding Society sample from the second wave of the Study.

"Extra 5 minutes" questions

Part of the sample, often referred to simply as the "Extra 5 minutes" sample, are asked some extra questions (approximately 5 minutes' worth) in addition to all the questions the rest of the sample are asked. These questions are generally those of particular relevance to ethnicity related research. For example, in Wave 1 this included questions on remittances, harassment, discrimination, and a detailed migration history.

The "Extra 5 minutes" sample comprises of

- OSMs in the EMBS
- OSMs selected to be part of the General Population Comparison Sample (GPCS). The GPCS consists of approximately 1000 households randomly selected from the GPS-GB (one of every 18 selected addresses in 40% of the selected PSUs). The achieved sample size was approximately 500 households in Wave 1.
- Ethnic minority OSMs in the GPS-GB living in low ethnic minority concentration areas. This status was frozen in Wave 1 and from Wave 2 onwards, all household members of these individuals were included in the "Extra 5 minutes" sample.

Note all TSMs co-resident with the "Extra 5 minutes" sample members are also asked the "Extra 5 minutes" questions.

Appendix B

Merging adjacent strata

In Understanding Society data, the values of the strata reflect the original ordering. In other words, two strata with consecutive numbers are contiguous. So, we can simply merge in single PSU strata with the strata right before or after these.

Let's start by revisiting the problem.

```
use pidp a_paygu_dv a_psu a_strata a_indinus_xw a_gor_dv ///
using "$inpath/ukhls_w1/a_indresp", clear

keep if a_paygu_dv>0
```

Remember to create `a_country`

```
svyset a_psu [pweight = a_indinus_xw], strata(a_strata)
svy: mean a_paygu_dv, over(a_country)
```

Next let us save the dataset.

```
save temp, replace
```

In these next few steps we will identify single PSU strata. So, we should keep one observation per stratum-PSU combination.

```
bys a_strata a_psu: keep if _n==1
```

Now we can count the number of PSU per stratum

```
bys a_strata: gen numpsu=_N
fre numpsu
```

We see there are 35 single PSU strata, list out the cases

```
li pidp a_strata a_psu a_country if numpsu==1
```

The next few steps are to merge these 35 strata with their adjacent stratum. First keep one observation per strata.

```
bys a_strata: keep if _n==1
```

Next, sort the dataset by strata and then generate a new variable that takes on the value of the adjacent stratum for single PSU strata

```
sort a_strata
gen newstrata =a_strata[_n+1] if numpsu==1
fre newstrata
li pidp a_strata a_psu a_country newstrata if numpsu==1
```

Next, we only keep the single PSU strata

```
keep if numpsu==1
keep a_strata newstrata numpsu
```

Next, we merge this data with the original saved data

```
merge 1:m a_strata using temp
fre numpsu
```

Finally, we replace the single PSU strata with the new values

```
replace a_strata = newstrata if numpsu==1
```

Let us now re-run the code for estimating average age of men living in UK and see if it works,

```
svyset a_psu [pweight = a_indinus_xw], strata(a_strata)
svy: mean a_paygu_dv, over(a_country)
```

It does not work. So, we repeat the process.

```
save temp, replace

bys a_strata a_psu: keep if _n==1
bys a_strata: gen numpsu2=_N
fre numpsu2
li pidp a_strata a_psu a_country if numpsu2==1

bys a_strata: keep if _n==1
sort a_strata
gen newstrata2 =a_strata[_n+1] if numpsu2==1
fre newstrata2
li pidp a_strata a_psu a_country newstrata2 if numpsu2==1
keep if numpsu2==1
keep a_strata newstrata2 numpsu2
merge 1:m a_strata using temp
drop _merge
fre numpsu2
replace a_strata = newstrata2 if numpsu2==1

svyset a_psu [pweight = a_indinus_xw], strata(a_strata)
svy: mean a_paygu_dv, over(a_country)
```

```
. svyset a_psu [pweight = a_indinus_xw], strata(a_strata)

      pweight: a_indinus_xw
          VCE: linearized
  Single unit: missing
     Strata 1: a_strata
         SU 1: a_psu
        FPC 1: <zero>

. svy: mean a_paygu_dv, over(a_country)
(running mean on estimation sample)

Survey: Mean estimation

Number of strata =    1736          Number of obs    =   22902
Number of PSUs   =    5450          Population size  = 23334.3
                                    Design df        =    3714

      england: a_country = england
        wales: a_country = wales
     scotland: a_country = scotland
    _subpop_4: a_country = northern ireland
```

Over	Mean	Linearized Std. Err.	[95% Conf.	Interval]
a_paygu_dv				
england	1911.865	13.76962	1884.868	1938.862
wales	1517.589	36.46704	1446.092	1589.086
scotland	1807.227	36.21151	1736.23	1878.223
_subpop_4	1609.037	41.72962	1527.222	1690.853

```
Note: 2 strata omitted because they contain no subpopulation
      members.
```

Appendix C

Dealing with Categorical Variables: Advantages and disadvantages of using factor variables in Stata

When the model you want to estimate includes categorical explanatory variables then you will first need to transform these variables into 0-1 dummy variables before you can use them as in the analysis. If there is a **n**-category variable then it needs to be transformed into **n** 0-1 dummy variables and only **n-1** of these dummy variables should be included in the model as explanatory variables. This will take a lot of time and effort. For example, if we want to include sex and educational achievement in the model, we will have to create dummy variables,

```
recode a_sex_dv 1=0 2=1, gen(female)

generat OtherHigher = 0 if a_hiqual_dv>0 & a_hiqual_dv <.
replace OtherHigher = 1 if a_hiqual_dv==2

generat ALevel = 0 if a_hiqual_dv>0 & a_hiqual_dv <.
replace ALevel = 1 if a_hiqual_dv==3

generat GCSE = 0 if a_hiqual_dv>0 & a_hiqual_dv <.
replace GCSE = 1 if a_hiqual_dv==4

generat OtherQual = 0 if a_hiqual_dv>0 & a_hiqual_dv <.
replace OtherQual = 1 if a_hiqual_dv==5

generat None = 0 if a_hiqual_dv>0 & a_hiqual_dv <.
replace None = 1 if a_hiqual_dv==9
```

Here we have chosen to make male and degree as the reference/omitted categories.

```
logit remit a_age_dv female OtherHigher ALevel GCSE OtherQual None
```

Factor variable method

An easy way to do this in Stata is to use factor variables. In this framework, once Stata knows that a variable is a categorical variable it will transform the **n**-category variables into **n-1** dummy variables; by default it will omit the lowest value category. For example, if you type

```
logit remit c.a_age_dv i.a_sex_dv i.a_hiqual_dv
```

Stata knows that **a_age_dv** is a continuous variable and **a_sex_dv a_hiqual_dv** are categorical variables. By default Stata will omit male and degree categories as they have the lowest values in those variables. As Stata results will be in terms of the value labels of the categorical variable, if you want to use this method it is a good idea to label the categorical variables sensibly.

Note you can specify which one to omit by typing `ib#` where # represents the numerical value of the category you want to omit. So, to make women the omitted category (we know sex takes on the value 1 for men and 2 for women and so by default men are the omitted category) type **`ib2.a_sex_dv`** instead of **`i.a_sex_dv`**

This method has another advantage. It allows you to easily deal with interactions and produce marginal effects of interacted variables quite easily. So, if we wanted to include age and age-squared term instead of creating a variable for age square we could have simply typed, **`c.a_age_dv##c.a_age_dv`**

Compare the two results and you will see the estimation results are exactly the same.