timeuse.org

CENTRE FOR
TIME USE RESEARCH

'timeuse'

Introducing user-written Stata commands for time diary data analysis

October 2024

Prepared by

Juana Lamote de Grignon Pérez and Elena Mylona

UCL

Contact details

Juana Lamote de Grignon Pérez
juana.lamote@gmail.com

Elena Mylona
e.mylona@ucl.ac.uk

About this report

Time use diaries keep track of what respondents do through the day, where they do it and who they do it with, among other things. Diaries are often collected for entire households allowing to investigate household dynamics. As a result, time use surveys are an incredibly rich source of data that can inform many disciplines. Yet, most of the potential of time use data remain untapped because of the difficulties associated with the manipulation of these complex data files. This article introduces a new **Stata Package timeuse**, that facilitates significantly the manipulation of time use diary data. The main programs are ***timeuse*** and ***timeusex***. Timeuse transforms the dataset to provide durations and instances of participation for several activities, while timeusex offers more detailed information, but for a single activity. In short, these programs transform the complex episode-level files into ready-to-use files containing the time use information desired by the user. The new files can be used by researchers with no prior experience with time use diary files.

To install, type:

ssc instal timeuse

Introduction

Time use data are in principle, any data containing information about how people use their time. There are two main ways in which time use data are collected: stylized questions and time use diaries. An example of a 'stylised question' is: *"How much time did you spend working last week?"*. On the other hand, a time use diary is a method where respondents provide a sequential account of their activities. Typically, the recording starts at some point during the night and ends at the same time the next day.

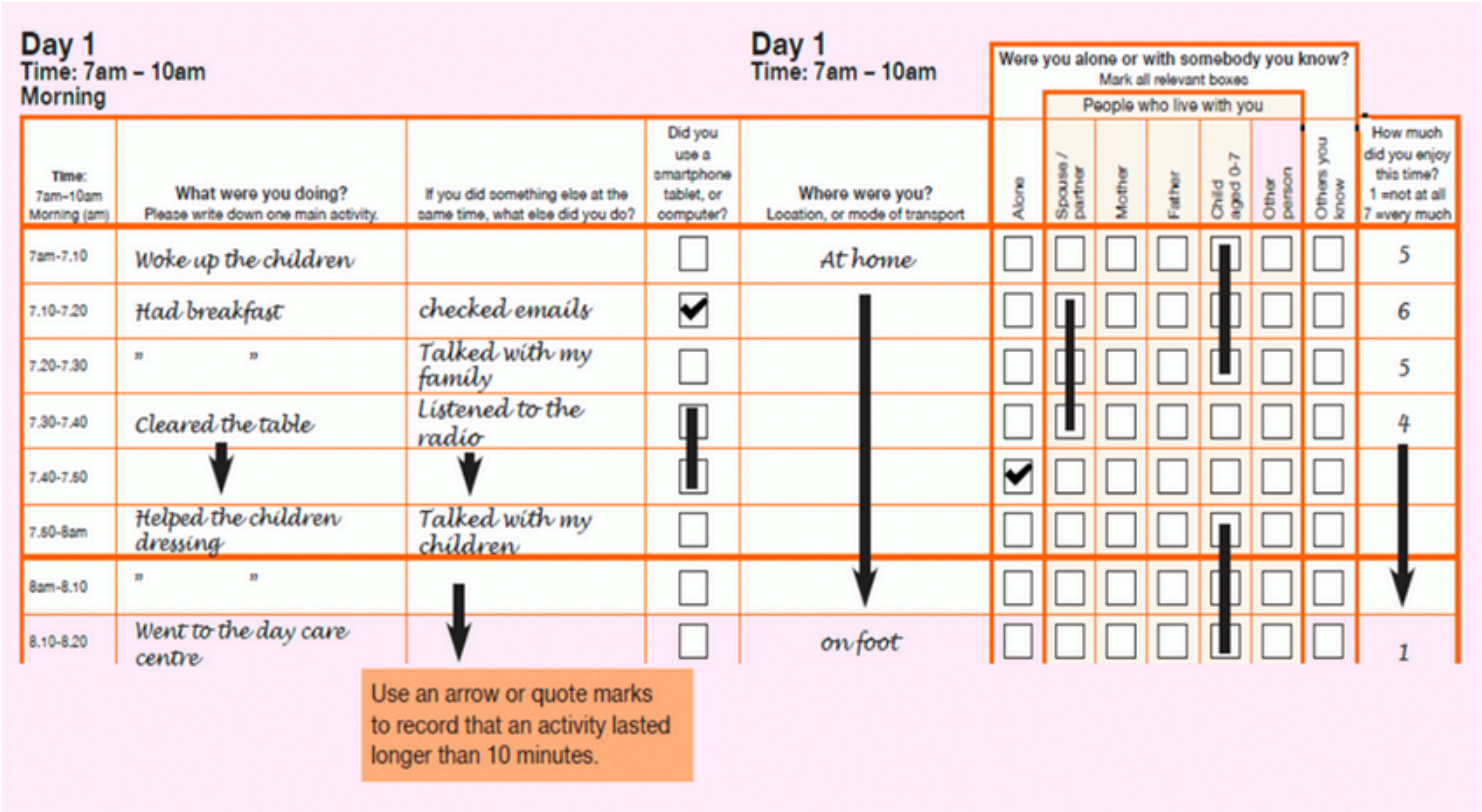
Time diaries are a reliable method to measure the time use of individuals. They have been validated through experimental studies comparing camera recordings against time diaries (Gershuny et al. 2020; Kelly et al 2015). Moreover, they provide much more comprehensive information than merely the time spent on different activities — which is all that stylized questions can offer. Time use diaries meticulously record what respondents do throughout the day, where they do it, who they are with, among other details.

Time diary data allow for investigating not only what people do and for how long they do it, but also allows for analysing the timing of those activities, interconnections between activities, and the associations of activities with other diary fields such as location, co-presence, subjective well-being, or the use of technological devices.

Often, diaries are collected for entire households, enabling the investigation of household dynamics as well. There are many time use surveys collected over several decades for many countries, with harmonised versions available through the Multinational Time Use Study (MTUS) and IPUMS, allowing for cross-country comparison

This report introduces a new Stata Package - timeuse - designed to enhance accessibility to time use data. The package consists of several programs, that will be introduced subsequently. However, before delving into the details of the package, we will first introduce the three main types of time use datasets.

Figure 1. The diary instrument used in the UK Time Use Survey of 2014.



The figure shows the diary instrument used in the UK for the Time Use Survey of 2014 (Office for National Statistics 2019). The figure only shows a part of the diary covering the period between 7am and 8:20 am. The full instrument is several pages long since the day is divided in 144 10-minute slots. For each of these 10-min slots, the respondent is expected to report the main activity, a simultaneous activity (if there was any), the use of ITs, the location, the presence of other individuals etc.

Time use file types

The information-rich source: the episode and the calendar files

When a respondent completes a diary, the recorded information is transcribed into a dataset. This dataset stores information on activities and the contextual variables for each moment of the day. Usually, each data row will represent 10 minutes in the life of the respondent. The dataset will have 144 observations for each day. This format is known as calendar format and is shown below:

Figure 2. An example of a calendar file.

ID	tslot	primary	secondary	location	ww_alone	ww_child	ww_partner	ww_other	ww_nonhh
1	1	sleep		at own home					
1	2	sleep		at own home					
1	3	sleep		at own home					
...	...	...	...	...	...	...	...	...	...
1	19	sleep		at own home					
1	20	self-care	self-care	at own home					
1	21	eat & drink		at own home	No	No	No	No	No
1	22	self-care		at own home					
1	23	self-care		at own home					
1	24	commute		travelling					
1	25	commute		travelling					
1	26	commute	other leisure	travelling	No	No	No	No	Yes
1	27	commute	other leisure	travelling	No	No	No	No	Yes
1	28	commute	other leisure	travelling	No	No	No	No	Yes
1	29	commute	other leisure	travelling	No	No	No	No	Yes
1	30	commute	other leisure	travelling	No	No	No	No	Yes
...	...	...	...	...	...	...	...	...	...
1	144	sleep		at own home	No	No	No	No	No

This figure shows a snapshot of a calendar file. Each row represents 10 minutes in the day of the respondent. The figure shows the diary identifiers and all the variables that define the episodes -the activity and the other activity features such as location, copresence, enjoyment etc. We have used different colours to show which time slots represent a single episode of activity -when all episode features are used to define the episode of activity. Every time there is a change in either of the episode level variables, a new episode of activity is defined -and a new colour is used.

This format is very intuitive because it looks exactly like the diary instrument that the respondents fill out, however it is a somewhat inefficient way of storing information. Simply by collapsing all time slots that are the same while including some information about duration, we have a much shorter and smaller file known as the episode or sequence file. This episode-level file provides the exact same information as the calendar file but takes up much less space. Besides, we could, at any time, produce the calendar file from the episode file, and vice versa.

Figure 3. An example of an episode file.

ID	epnum	primary	secondary	location	...	start	end	time	closkst
1	1	sleep		at own home		0	200	200	7.2
1	2	self-care	self-care	at own home		200	210	10	7.3
1	3	eat & drink		at own home		210	230	10	7.4
1	4	self-care		at own home		230	250	20	8
1	5	commute		travelling		250	270	20	8.2
1	6	commute	other leisure	travelling		270	320	50	8.4
...	...	...	...	...		...	...	...	...
1	25	sleep		at own home		1110	1440	330	22.3

The figure shows the same dataset as Fig 2 but this time the data is organised in episodes. Note that in the new format, each row represents an episode of activity regardless of its duration. The different episodes are coloured using the same colours as in the figure above.

The episode file and the calendar file can be presented in long or wide format. The examples presented in Figures 2 and 3 are in long format – long because the day detail is displayed vertically, making the file longer – with more observations. There is another possibility to store that information and that is using a wide format. Although long formats are somewhat more common, as their structure is like panel data, the wide formats are also used. Using the Stata command ‘reshape’ one can go from one long to wide, and vice versa. The programs introduced in this article start off from an episode file in long format.

A shortcut for users: aggregated diary-level files

Aware of the difficulties of dealing with the rich-episode file, institutions in charge of collecting and producing time use data, often make a further processing of the richer data file to make things easier for the user, and they convert a file like the episode file described before into a diary-level file where instead of seeing the full detail of the episodes, each row of data represents a full day and different variables summarise some aspect of time use, often, the amount of time devoted to different activities. Figure 4 shows an example of an aggregated file from the Multinational Time Use Study (MTUS) (Lamote de Grignon Pérez et al. 2024).

Figure 4. The aggregated file produced by the Multinational Time Use Study that shows time spent across activities in each diary.

sex	age	day	sleep	eatdrink	selfcare	paidwork	educatn	foodprep	cleanetc
Woman	48	Thursday	290	150	10	0	0	150	80
Woman	48	Sunday	460	90	70	0	0	180	190
Man	13	Thursday	540	120	50	0	400	10	0
Man	13	Sunday	690	0	10	0	20	10	30
Man	75	Sunday	470	110	70	0	0	70	10
Man	75	Tuesday	400	60	140	40	0	140	50
Woman	68	Wednesday	580	70	40	0	0	100	190
Woman	68	Saturday	590	50	50	0	0	90	60
Man	74	Wednesday	370	80	150	0	0	30	150
Man	74	Saturday	410	60	150	0	0	10	80

The figure shows a snapshot of the aggregated file that the Multinational Time Use Study shares with their users. On the rows we have different days and on the columns, some demographic information followed by the variables that measure time devoted to a series of activities on the day.

Aggregated files are notably more user-friendly than the episode or calendar files as they follow similar structure to plenty of social science datasets, where rows represent individuals, and columns include variables revealing information about those individuals. Researchers using aggregated files can go straight to doing analyses. Unfortunately, the information included in aggregated files is limited, rarely going beyond the durations of the primary activities.

If we want to know how much time we spend eating alone versus in eating in the company of others, or how much we are multitasking – any reporting of a secondary activity -, or even the time at which people are more likely to do sports, accessing the episode file becomes necessary. The latter information is unlikely to appear in an aggregated file because the amount of possible aggregated files that will need to be produced are endless. The Stata package timeuse helps in the manipulation of data files by for example, creating aggregated file, customised to the researcher's needs.

The example datasets that will be used throughout the report

All the examples and illustrations provided in this report use the following example datasets: “diary.dta”, “calendar.dta” and “atusdiary.dta”. These datasets contain small fictitious time-use data, and they are available to all users who have installed timeuse, after typing “net get timeuse”. “diary.dta” is an episode file, “calendar.dta” is a file in time slots, and “atusdiary.dta” is an episode file resembling those in the American Time Use Survey (ATUS).

The file “diary.dta” contains 3,309 observations – episodes of activity - corresponding to 100 diaries. Each diary can be uniquely identified with the variables personid and diaryid. These diaries belong to 50 different individuals (identified by the variable personid). The number of episodes in each diary is determined by the number of times that there is a change in either the activity (primary or secondary) or any of the activity features: location, co-presence and enjoyment.

Variables which must exist for many of the commands to work: 'start' and 'end'.

Two key variables in the episode file are those defining the time at which the episode of an activity begins and ends. These variables are called 'start' and 'end' in "diary.dta" file, and measure the timing of the activity expressed as the minute of the day. Each day or diary has 1440 minutes (24 hours), and the variables 'start' and 'end' give you the timing of the activity expressed as the number of minutes that have elapsed since the diary begun. In all diaries, the first activity reported starts at minute 0 and the last one ends at minute 1440.

What may change from one diary to another is the clock-time that such minute will represent because that will depend on the time at which the diary stars. For example, in the dataset "diary.dta" that starts at 4:00 am, an activity that begins at 6:00, will begin at minute 120, whereas in a diary starting at 5:00 am, it will begin at minute 60. Note, however, that when one browses the 'diary.dat' file, they will not see the minute-of-the-day, but the clock-like format attached to the variable. This label is attached to facilitate inspection of the data, but it is important to keep in mind what the variable values are like (i.e., that the start and end range from 0 to 1440).

The programs included in timeuse require the existence of the timing variables with this exact format and with the exact names: ‘start’ and ‘end’. If a dataset contains these variables with the right format but a different name, renaming is required to start using the programs. For example, “rename begin start”.

If the timing is expressed using a different format, you will need some additional recoding before you can use the programs. The program clocktomin can be used to help users with such specific transformations, and is the one needed for the American Time Use Survey files.

start: start time of the episode expressed as minute of day. ‘start’ can take values from 0 to 1439. The first episode of the diary should start at minute 0.

end: end time of the episode expressed as minute of day. ‘end’ can take values from 1 to 1440. The last episode of the diary should end at minute 1440.

Some datasets provide the start and end time of activities, but others only provide the start time only. In those cases, the end time will need to be generated, by assuming that each activity ends by the time the next activity starts (see box below).

```
ssc describe timeuse
net get timeuse
use diary, clear
drop end
egen udid=group(personid diaryid)
bysort udid: egen lastep=max(epnum)
xtset udid epnum
gen end=f1.start
replace end=1440 if epnum==lastep
```

Options that are not optional: the diary identifier and the start time of the diary

The diary identifier

All the commands included in this package require that the user indicates the diary identifier. In other words, the variable, or combination of variables that when jointly considered uniquely identify each diary.

Often, time use surveys have several identifiers that need to be used in combination to uniquely identify each diary: a household identifier, a personal identifier for each household member, and a third one that identifies each diary. If this is the case, all three identifiers need to be specified in the 'diaryid' option, which is used in all commands.

The start time of the diary

Some of the commands in the 'timeuse' package must be informed of the time the diary day started to assign the correct clock-like value labels. Therefore, some of the commands will include an option called 'diaryst(n)' where the user needs to provide a number that indicates the start time of the diary in a 24-hour system clock, with "0" representing midnight, "4" is 4am, and "18" would be 6pm. See the examples of the timeusex command to see some examples.

Extracting basic time use information across a range of activities:
timeuse.ado

The program timeuse transforms an episode file into a diary-level file containing total time spent on the different categories of activity of 'var' (var_1 to var_n), as well as the number of episodes for each activity (var_1_n to var_N_n).

The syntax:

```
timeuse var, diaryid(string)
```

Although no part of the syntax the program needs the variables 'start' and 'end' to exist and to have those exact names and format as explained in the previous section.

The categories of 'var' do not need to be labelled but it is recommended that they are so that the output file produced is much easier to interpret. 'var' can have missing values but no negative values are allowed.

'timeuse'

Example 1: time spent across the different categories of the primary activity

Using the example dataset "diary.dta", we produce a diary-level file that will give us the amount of time devoted to each of the categories of 'primary'.

```
ssc describe timeuse
net get timeuse
use diary, clear
timeuse primary, diaryid(personid diaryid)
describe
sum primary_1-primary_20
```

After running timeuse a small report will be displayed in the results window that tells the user whether the variables var_1 to var_n containing the time spent on each category of activity adds up to 1440 or not. In this case they do not because 'primary' has some missing values. The activity variable often has missing values, and, in those cases, it is expected that the sum of the different activities does not add up to 1440 minutes.

If you preferred to have variables that add up to 1440, simply recode the missing values before running timeuse. This is what we do in the following example. The resulting file has 40 new variables, 20 containing the time spent across the 20 categories of primary and 20 that give the number of times that the individual took part on that activity on that day. We explore average time across the different activities with the last line of code.

Example 2: Number of times people go outdoors in a day

This time we want to know about the number of times people go outdoors and how much time they spend outdoors every time they do so. The episode variable containing this information is the variable 'inout' that takes three possible values: "1" means the person is indoors, "2" outdoors, "3" in a vehicle and then there are some missing values. This time we want to have categories of activity adding up to 1440 minutes per day, so we will start by recoding the missing values by some valid number.

```
ssc describe timeuse
net get timeuse
use diary, clear
recode inout (.=4)
lab define inout 4 unknown, modify
lab value inout inout
timeuse inout, diaryid(personid diaryid)
desc inout*
sum inout_2_n
sum inout_2
```

This time we get the message saying that indeed “The activities created, inout_1 to inout_n, add up to 1440 minutes.” By describing the dataset, we will know which of the created variables in the one we are interested in (see Fig #N). The variable of interest are ‘inout_2’ and ‘inout_2_n’.

Figure 5. Example of the data output produced by timeuse.ado

Variable name	Storage type	Display format	Value label	Variable label
personid	float	%3.0f		person identifier
diaryid	double	%3.0f	diaid	diary identifier
inout_1	float	%9.0g		mpd on: inside
inout_2	float	%9.0g		mpd on: outside
inout_3	float	%9.0g		mpd on: in vehicle
inout_4	float	%9.0g		mpd on: unknown
inout_1_n	float	%9.0g		n episodes: inside
inout_2_n	float	%9.0g		n episodes: outside
inout_3_n	float	%9.0g		n episodes: in vehicle
inout_4_n	float	%9.0g		n episodes: unknown

This figure shows the variables in the dataset after running timeuse1.ado in the diary.dta. The program has created 8 new variables: 4 that measure the minutes per day across the 4 activity groups, and 4 that show number of instances of participation for each of the 4 categories of activity. To facilitate exploration, these variables will be put at the beginning of the file right after the identifiers. After the variables created by the program, all the other variables that were initially in the dataset will still be there after the newly created variables. While this may be confusing for the episode level variables that will still be there -because these variables no longer make sense once the file is has been converted to a diary level file-, it is convenient to have the any other individual and diary level variables (eg. gender, age and day of the week).

‘timeusex’

Extracting detailed time use information for a given activity:
timeusex.ado

The program timeusex transforms an episode file in long format into a diary-level file that summarizes the time use information for the selected activity. The syntax is similar to that of timeuse, but ‘var’ is no longer a categorical variable where each value represents one activity, but binary (dummy) variable taking the value 1 whenever the episode of activity is equal to the activity of interest, and zero any other value otherwise.

The syntax:

```
timeusex var, diaryid(string) diaryst(string)
```

The user should fill in the diaryid and the diaryst options. The option diaryst it is just a number that indicates the start time of the diary in a 24-hour system clock, with “0” representing midnight, “4” is 4am, and “18” would be 6pm. The variables start and end need to be there.

The new file will contain variables with the following information: total time spent on the activity (total), number of times the diarist engages in the activity on that day (episodes), and timing and duration for the different episodes of activity (start1 to startN, end1 to endN, duration1 to durationN). The variables start1 to startN, and the variables end1 to endN capture the beginning and end of each episode of activity. Finally, the program produces the variables: start_last, end_last, and duration_last, showing the timing and duration for the last episode of activity.

In some way the variables referring to the last episode of activity are a duplicity; however, we think it will make the users life easier to have these variables there. If for instance the users what to know the time at which people have dinner, looking at the last eating episode in a diary is probably better than taking the third eating episode because not all diaries include the same number of eating episodes.

For now, the program allows a single variable in the argument. If the user wanted to produce the time use outputs for a second activity, the command would need to be run a second time after reopening the data again because timeusex transforms the dataset.

Example 3: Exploring eating behaviour

We want to know how much time people spend eating, how many times people eat, and at what time they do so. The command timeusex can give us all this information. We start by creating a dummy that flags eating episodes, and then we can run timeusex.

```
ssc describe timeuse
net get timeuse
use diary, clear
gen eating=0
replace eating=1 if primary==2|secondary==2
timeusex eating, diaryid(personid diaryid) diaryst(4)
```

After running the syntax above, the data editor will look like Figure 6. To make the figure more visible, we do not show the identifiers, but it will suffice to know that each row of data represents one day in the life of a person. Note how some diaries have three or four episodes of eating corresponding to breakfast, lunch, dinner and some snack somewhere. But there are also diaries with no eating at all (or at least no reported eating), like the 4th diary in the figure. This may be a mistake from the diarist that has forgotten to report eating or it may be the case that the person in question did not eat at all in that day.

Figure #6. The diary level file produced by timeusex – replace.

	total	episodes	start1	end1	duration1	start2	end2	duration2	start3	end3	duration3	start_last	end_last
1	150	4	06:40	06:50	10	07:00	07:30	30	13:00	14:00	60	18:00	18:50
2	140	4	07:20	08:00	40	10:00	10:10	10	13:10	14:00	50	18:20	19:00
3	120	4	07:20	07:30	10	13:20	14:30	70	19:00	19:30	30	20:40	20:50
4	0	0	.	.	0	.	.	0	.	.	0	.	.
5	180	7	08:20	08:30	10	09:00	09:10	10	12:00	12:50	50	20:10	20:20
6	190	2	07:30	08:30	60	12:30	14:40	130	.	.	.	12:30	14:40
7	70	4	09:10	09:20	10	12:40	13:00	20	15:40	16:00	20	17:30	17:50
8	50	2	13:00	13:20	20	18:30	19:00	30	.	.	.	18:30	19:00
9	80	3	09:00	09:40	40	12:40	12:50	10	18:00	18:30	30	18:00	18:30
10	60	3	10:10	10:20	10	13:00	13:20	20	19:00	19:30	30	19:00	19:30
11	60	5	05:50	06:00	10	15:40	15:50	10	18:20	18:40	20	20:50	21:00
12	80	4	05:50	06:20	30	09:20	09:30	10	16:30	16:40	10	19:00	19:30
13	40	1	18:00	18:40	40	.	.	.	.	.	.	18:00	18:40
14	70	3	05:50	06:20	30	16:30	16:40	10	19:00	19:30	30	19:00	19:30
15	50	2	12:00	12:30	30	17:00	17:20	20	.	.	.	17:00	17:20
16	80	2	11:40	12:30	50	19:50	20:20	30	.	.	.	19:50	20:20

The figure shows the output file produced by timeusex to explore eating. The new file has only 100 observation or rows corresponding to the 100 diaries in the dataset. For each of them, we know the total number of minutes spent eating (total), the number of times we eat and/or drink during the day (episodes), as well as the start, end and duration of each of those episodes of eating and drinking. For example, the diary number 2 devotes 140 minutes to eating and that eating is broken down into three episodes of activity. Breakfast starts at 9:40 for that person and day and ends 10:00. Lunch takes place between 14:00 and 15:00 and dinner starts at 18:30 and ends at 19:30.

With regards to the timing of activities, the program will produce as many start and end variables as number of eating episodes there are in the diary that has the largest number of episodes. Then, in many diaries, many of those episodes will be ".". In this example there are 7 start variables, 7 end variables, and 7 duration variables, this is so because there is there is one or more diaries with 7 eating episodes. For that diary in question this seven variables will contain valid values, for all the other variables that have fewer eating episodes, some of those variables will have a missing value.

Note that the first eating episode of the diary is breakfast, so simply by exploring 'start_1', 'end_1' and 'duration_1' we can learn a lot about breakfast. If, however, you are interested in knowing when and for how long people were having dinner, we should probably look at the last eating episode in each diary: start_last, end_last and duration_last. If we wanted to explore lunch time, a good strategy could be to look for eating episodes in a given time frame and choose one of them. For example, we could define lunch as the longest eating episode taking place between 12 and 2 pm.

Example 4: Solitary screen use

The activity of interest doesn't need to be just an activity (in fact it doesn't need to be an activity at all), but it could also be a combination of an activity and some other feature of the activity, e.g. activity and copresence. If we were interested in looking at solitary screen use, all we would need to do is to create a dummy capturing that behavior and then running timeusex on that variable. But in this example, we will do something more, to run the program a second time on a dummy looking at screen time with family to be able to compare some aspects of solitary screen time and family screen time. This requires saving the variables of interest with a different name and saving a file before running timeuse the second time.

```
ssc describe timeuse
net get timeuse
use diary, clear
gen tvalone=0
replace tvalone=1 if primary==14 & ww_alone==1
timeusex tvalone, diaryid(personid diaryid) diaryst(4)
keep personid diaryid episodes duration_last
rename episodes episodes_alone
rename duration_last duration_last_alone
sort personid diaryid
tempfile tvalone
save "`tvalone'"
use diary, clear
gen familytv=0
replace familytv=1 if primary==14 & ww_alone==0
timeusex familytv, diaryid(personid diaryid) diaryst(4)
keep personid diaryid episodes duration_last
rename episodes episodes_family
rename duration_last duration_last_family
sort personid diaryid
tempfile familytv
save "`familytv'"
use "`tvalone'", clear
merge 1:1 personid diaryid using "`familytv'"
```

About CTUR

The Centre for Time Use Research (CTUR) is an ESRC-designated Legacy Research Centre specialising in the collection and analysis of time-use diary data, disseminating both skills and datasets to a wide international audience, and conducting a world-leading research programme.

CTUR originated, produces, and disseminates the Multinational Time Use Study (MTUS), which currently provides 1.5 million days of randomly sampled time diary evidence for 25 countries over 65 years, via its website (timeuse.org) and IPUMS USA (ipums.org).

We work with time-use data to investigate issues of social life, paid work patterns, work-life balance, family, gender, and socio-economic structure. Our centre also specialises in finding innovative applications for population time-use evidence in fields such as public health, transport and energy research.

CTUR is an Economic and Social Research Council (ESRC) designated research centre and it has been based in the UCL Institute of Education since April 2019.

CENTRE FOR
TIME USE RESEARCH

Report created by

Elena Mylona
e.mylona@ucl.ac.uk