The Stata Journal

The *Stata Journal* publishes reviewed papers together with shorter notes or comments, regular columns, book reviews, and other material of interest to Stata users. Examples of the types of papers include 1) expository papers that link the use of Stata commands or programs to associated principles, such as those that will serve as tutorials for users first encountering a new field of statistics or a major new technique; 2) papers that go "beyond the Stata manual" in explaining key features or uses of Stata that are of interest to intermediate or advanced users of Stata; 3) papers that discuss new commands or Stata programs of interest either to a wide spectrum of users (e.g., in data management or graphics) or to some large segment of Stata users (e.g., in survey statistics, survival analysis, panel analysis, or limited dependent variable modeling); 4) papers analyzing the statistical properties of new or existing estimators and tests in Stata; 5) papers that could be of interest or usefulness to researchers, especially in fields that are of practical importance but are not often included in texts or other journals, such as the use of Stata in managing datasets, especially large datasets, with advice from hard-won experience; and 6) papers of interest to those who teach, including Stata with topics such as extended examples of techniques and interpretation of results, simulations of statistical concepts, and overviews of subject areas.

The *Stata Journal* is indexed and abstracted by *CompuMath Citation Index*, *Current Contents/Social and Behavioral Sciences*, *RePEc: Research Papers in Economics*, *Science Citation Index Expanded* (also known as *SciSearch*, *Scopus*, and *Social Sciences Citation Index*.

For more information on the *Stata Journal*, including information for authors, see the webpage

http://www.stata-journal.com

Subscriptions are available from StataCorp, 4905 Lakeway Drive, College Station, Texas 77845, telephone 979-696-4600 or 800-STATA-PC, fax 979-696-4601, or online at

http://www.stata.com/bookstore/sj.html

Subscription rates listed below include both a printed and an electronic copy unless otherwise mentioned.

U.S. and Canada		Elsewhere	
Printed & electronic		**Printed & electronic**	
1-year subscription	$ 98	1-year subscription	$138
2-year subscription	$165	2-year subscription	$245
3-year subscription	$225	3-year subscription	$345
1-year student subscription	$ 75	1-year student subscription	$ 99
1-year university library subscription	$125	1-year university library subscription	$165
2-year university library subscription	$215	2-year university library subscription	$295
3-year university library subscription	$315	3-year university library subscription	$435
1-year institutional subscription	$245	1-year institutional subscription	$285
2-year institutional subscription	$445	2-year institutional subscription	$525
3-year institutional subscription	$645	3-year institutional subscription	$765
Electronic only		**Electronic only**	
1-year subscription	$ 75	1-year subscription	$ 75
2-year subscription	$125	2-year subscription	$125
3-year subscription	$165	3-year subscription	$165
1-year student subscription	$ 45	1-year student subscription	$ 45

Back issues of the *Stata Journal* may be ordered online at

http://www.stata.com/bookstore/sjj.html

Individual articles three or more years old may be accessed online without charge. More recent articles may be ordered online.

http://www.stata-journal.com/archives.html

The *Stata Journal* is published quarterly by the Stata Press, College Station, Texas, USA.

Address changes should be sent to the *Stata Journal*, StataCorp, 4905 Lakeway Drive, College Station, TX 77845, USA, or emailed to sj@stata.com.

The Stata Journal (2013)
13, Number 3, pp. 618–624

Allocation of ordered exclusive choices

Marc Sangnier
Sciences Po and Paris School of Economics
Paris, France
marc.sangnier@sciences-po.org

Abstract. In this article, I describe the `alloch` command, which helps to allocate exclusive choices among individuals who have ordered preferences over available alternatives.

Keywords: st0311, alloch, random allocation, choice criterion

1 Introduction

Scarcity makes exclusive choices a frequent feature of the real world: everyone cannot get his or her preferred outcome when someone's choice prevents an object from being allocated to someone else. Such situations arise when students are allocated to schools or when guests are seated around a dining table. Similarly, teachers often face situations where they need to allocate students to different tasks, for example, presentations or book reviews. In this article, I present a simple command, `alloch`, that handles situations where each choice can be attributed to only one individual. This command has initially been designed to allocate students to classroom presentations but can be used to allocate workers to individual tasks or employees to desktops located in different places. The command performs this task in flexible ways by taking care of conflicting individual preferences and allowing different criteria to look for the "best" allocation.

Using a dataset that contains ordered choices made by individuals, the `alloch` command enables the user to quickly allocate those individuals to the choices they made. Its basic principle is to randomly draw an individual and to give that individual his or her preferred choice. It then draws a second individual, who receives his or her first choice if it is still available; otherwise, the individual receives his or her second choice. And so on and so forth. A property of the resulting allocation of choices among individuals is that it is renegotiation proof: once individuals see the resulting allocation, no pair of individuals is interested in a bilateral exchange of allocated choices.

Because the `alloch` command relies on random draws, many different allocations may be obtained using this process. Thus the user may be tempted to look for the "best" one. This requires a criterion to discriminate between feasible allocations. Should we prefer that individual A or individual B receives his or her lower-ranked choice? The `alloch` command proposes several criteria to choose among allocations. To achieve this, the command first creates a given number of different allocations and then chooses the most efficient one according to the chosen criterion.[1]

2 Required structure of the data

The data used by the `alloch` command must be structured as follows:

- Each line must contain the ordered choices of a separate individual over a list of alternatives.
- Each individual must be uniquely identified by a list of variables.
- The chosen objects must be identified by consecutive numbers ranging from 1 to N, where N is the total number of alternatives available to individuals.
- The most preferred choice of each individual must be contained by a variable that ends with 1 (for example, `pref_1`).
- The second preferred choice must be contained by a variable that ends with 2 and starts with the same name as the most preferred choice (for example, `pref_2`).

In practice, you can construct the data used by the command in two steps. First, provide individuals with a list of numbered alternatives and ask them to order alternatives by using the alternatives' identification numbers. Note that the number of available alternatives must be at least as large as the number of individuals.[2] Second, append these ordered preferences in a single dataset where each line corresponds to the preferences of a single individual and where each individual has a unique identifier.[3]

The command is designed to deal with choices badly filled by individuals. They may rank the same alternative in two different places (for example, rank the same object as first and fourth preferred choices). They may also leave a blank position (that is, not provide an object for the second choice but fill the first and the third ones). Finally, they may rank only an insufficient number of objects (for example, rank only two objects out of a list of four, while objects will be allocated among three individuals). By default, the `alloch` command does not treat such individuals in any particular way. That is, an individual is allocated his or her preferred choice if it is still available when that individual is drawn; otherwise, he or she receives a random choice once the process is

1. In such a framework, algorithms to determine the "best" allocation could be used ex ante. However, such processes become very demanding once the numbers of choices and individuals become large. This is why this part of the `alloch` command relies on a series of random draws.
2. Otherwise, it would be impossible to allocate each individual a different choice.
3. Online forms or spreadsheets are convenient tools to use to achieve these preliminary steps.

over for all individuals who did not make any errors when expressing their preferences. Under the `sanction` option (see below), such individuals are removed from the pool at the beginning of the process and randomly allocated to one of the remaining objects at the very end of the process, even if their preferred choices are still available.

3 The alloch command

3.1 Syntax

The `alloch` command has been developed under Stata 11.2. Its syntax is the following:

`alloch` [`using` *filename*] `,` `choice(`*var*`)` `alter(`#`)` `ident(`*varlist*`)` [`iter(`#`)`
`linear quadratic unique sanction detail plot`]

The command uses the active dataset if `using` *filename* is not specified. Required and optional options are described below.

3.2 Options

`choice(`*var*`)` is required. *var* must be the common part of names of variables that contain the ordered choices. For example, if the most preferred choice is entered in variable `pref_1` and the second preferred choice is entered in variable `pref_2`, then *var* must be set to `pref_`. Note that choice variables must be numbered consecutively.

`alter(`#`)` is required. # must be set to the total number of different alternatives available to individuals. For example, if individuals have to choose among 30 different items, then # must be set to 30, regardless of the number of choices that have been expressed. This option is necessary to deal with individuals for whom the set of expressed choices is incomplete or badly filled.[4]

`ident(`*varlist*`)` is required. *varlist* must be the variables that uniquely identify observations.

`iter(`#`)` specifies the number of iterations. The default is `iter(100)`.

`linear` and `quadratic` specify the criterion used to choose among several distributions. The default criterion is a Rawlsian one, where the chosen distribution is the one that minimizes the number of individuals with low-ranked choices.[5] `linear` minimizes the sum of ranks. `quadratic` minimizes the sum of squared ranks. `linear` and `quadratic` may not be used simultaneously.

4. See above for more details about the required data structure.
5. See Rawls (1974). This criterion is also known as “maximin”. Technically, this criterion compares any pair of allocations and selects the one that allows the worst-off individuals to receive lower-ranked choices. The `alloch` command implements a lexicographic Rawlsian approach as it tries to improve the outcome of the second worst-off individual if it is not possible to improve the outcome of the worst-off individual, and so on.

`unique` specifies that only one random allocation be run and that it be the final one. `unique` offsets `linear`, `quadratic`, and `iter(#)`.

`sanction` specifies that all individuals for whom at least one choice is missing or who put the same choices in different positions be excluded from iterations. These individuals will not be used when choosing among distributions and will be randomly allocated to available choices when the process is over. By default, such individuals are kept in the process and allocated to one of their choices if possible.

`detail` specifies that precise information about individuals with missing or redundant choices be returned.

`plot` produces a simple histogram of the chosen allocation of choices.

3.3 Output

`alloch` preserves the used dataset and returns a dataset that contains the unique identifiers of observations and three variables: `outcome` displays the identification number of the allocated choice; `draw` displays the position at which the individual has been drawn; and `choice_rank` presents the rank of the choice allocated to the individual.

4 Example

Table 1 presents the general structure of data used by the `alloch` command. Here 44 individuals identified by their first and last names have been asked to choose among 50 different choices. To ensure consistency, the planner required them to express 47 choices.[6] These data are original choices expressed by students about presentations to be allocated.[7]

Table 1. Structure of data used by the command `alloch`

first_name	name	pref_1	pref_2	pref_3	...	pref_45	pref_46	pref_47
Kent	Brockman	37	19	38		47	2	39
Bart	Simpson	41	26	47		24	28	45
Homer	Simpson	33	19	31				
⋮	⋮							
Milhouse	Van Houten	4	9	15		36	20	32
Rainier	Wolfcastle	48	17	25		45	12	19

6. Note that choices ranked from 45 to 47 are useless. Such a situation, however, may occur when the planner overestimated the number of respondents when gathering preferences.
7. Students' names have been replaced by names of characters from *The Simpsons* TV show.

Let us type

```
. use alloch_example
. set seed 148
. alloch, alter(50) ident(first_name name) choice(pref_) iter(10)
Progress (10)
..........
--------------------------------------------------------------------------------
Random allocation of ordered exclusive choices
--------------------------------------------------------------------------------
Warning: Observations with the same choice in different positions.
Observations #: 12, 43
Warning: Observations with missing choices.
Observations #: 26, 30
--------------------------------------------------------------------------------
Number of iterations: 10
Choice criterion: Rawls
--------------------------------------------------------------------------------
Number of choices: 47
Number of alternatives: 50
Number of observations: 44
Number of post-allocated: 0
--------------------------------------------------------------------------------
```

The first block of information in the output warns the user that individuals 12, 26, 30, and 43 either did not submit enough choices or put the same choice in more than one position. The second block of information recalls the criterion used and the number of iterations required by the user. Finally, the last block of output displays basic information about the process. The initial dataset contains the 47 ordered choices (from a list of objects numbered from 1 to 50) expressed by 44 individuals. In the resulting allocation, none of the four individuals with badly filled preferences have been randomly assigned to choices.[8]

8. If one of them had been drawn at a point where the list of still-available objects had not contained one of his or her expressed choices, then the number of postallocated individuals would be different from 0.

Because option iter(#) is set to 10, one would obtain the "best" out of 10 allocations from the Rawlsian point of view because the linear and quadratic options are not specified. That is, the resulting allocation is such that the nine other allocations do not provide the worst-off individual with lower-ranked choices.[9] Resulting data are presented in table 2. This table means that O. Simpson was the first to be drawn and received his first choice, that is, object number 26. On the other hand, R. Wolfcastle was the last to be drawn and received his ninth choice.

Table 2. Structure of data produced by the command alloch

first_name	name	outcome	draw	choice_rank
Orville	Simpson	26	1	1
Gros	Tony	29	2	1
⋮	⋮			
Cyrus	Simpson	10	43	1
Rainier	Wolfcastle	28	44	9

By adding the option plot, the user would produce figure 1. This figure means that the resulting allocation managed to provide 29 individuals with their most preferred choice, 11 with their second preferred choice, etc.

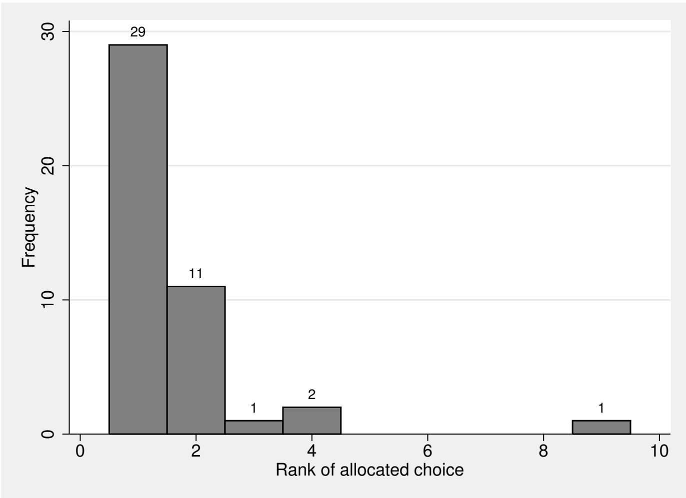

Figure 1. Distribution of allocated choices

9. See the description of the linear and quadratic options for additional insight about the Rawlsian criterion.

Assume that the planner wants to impose sanctions on the individuals who made errors in the expression of their preferences. Let us type

```
. use alloch_example
. set seed 148
. alloch, alter(50) ident(first_name name) choice(pref_) iter(10) sanction
> detail
Progress (10)
..........
--------------------------------------------------------------------------------
Random allocation of ordered exclusive choices
--------------------------------------------------------------------------------
Warning: Observations with the same choice in different positions.
Leonard Lenny (12), pref_15=pref_18
Van Houten Milhouse (43), pref_5=pref_6, pref_1=pref_9
Warning: Observations with missing choices.
Wiggum Ralph (26), pref_9
Simpson Homer (30), pref_47, pref_46, pref_45, pref_44, pref_43, pref_42,
> pref_41, pref_40, pref_39, pref_38, pref_37, pref_36, pref_35, pref_34,
> pref_33, pref_32, pref_31, pref_30, pref_29, pref_28, pref_27, pref_26,
> pref_25, pref_24, pref_23,
--------------------------------------------------------------------------------
Number of iterations: 10
Choice criterion: Rawls
--------------------------------------------------------------------------------
Number of choices: 47
Number of alternatives: 50
Number of observations: 44
Number of sanctions: 4
--------------------------------------------------------------------------------
```

By adding the `detail` option, the user is informed about the identity and the precise errors of the four individuals who made errors. As you can see, L. Leonard ranked the same object in positions 15 and 18. M. Van Houten also made similar errors. R. Wiggum forgot to fill his 9th choice, and H. Simpson simply stopped ranking objects after his 22nd preferred choice. By specifying the `sanction` option, the user requests that these four individuals be removed from the process and allocated random choices after all other individuals have been drawn.

5 Acknowledgment

This program uses suggestions from Stata tip 41 (Harrison 2007).

6 References

Harrison, D. A. 2007. Stata tip 41: Monitoring loop iterations. *Stata Journal* 7: 140.

Rawls, J. 1974. Some reasons for the maximin criterion. *American Economic Review* 64: 141–146.

About the author

Marc Sangnier is a postdoctoral fellow at Aix-Marseille School of Economics.