

Analyses of Sequences using Stata

The SQ-Ados 2.0

Ulrich Kohler

`ulrich.kohler@uni-potsdam.de`



Faculty of Economics and Social Sciences
University of Potsdam

<http://www.uni-potsdam.de/pcqr>

2016 Italian Stata Users Group Meeting
Rome, November 17th 2016

Contents

- 1 Introduction
- 2 Tables
- 3 Graphs
- 4 Sequence statistics
- 5 Sequence Similarity Statistics
- 6 Applications

Inhalt

- 1 Introduction
- 2 Tables
- 3 Graphs
- 4 Sequence statistics
- 5 Sequence Similarity Statistics
- 6 Applications

Definition of Sequences

Sequences are entities carrying a certain characteristic. They are build from elements organized in a specific order. The order of the elements defines the characteristic of the sequence.

☞ Examples

● ♣B ♠B ♣A ♣10 ♣9 ♣8 ♣7 ♥A ♠9 ♦7



● | G | A | A | T | T | C |

● | I | N | F | I | N | I | T | Y |

Analysis of Sequences

Sequence analysis aims to find similarities between sequences, or to detect typical sequences.

Similarities between sequences may arise from common causes (common ancestors), or due to causal relationships between the sequences.

👉 Examples

- Spelling Checker
- Detection of family relationships
- Transition from school to work (description of societies)
- Record Linkage (cf. Schnell et al., 2004)

Sequence analysis does **not** deal with relationships of the elements **within** the sequences. It is a description of the characteristics of the entire sequences.

Techniques for the analysis of sequences

Sequences can be analyzed with various devices:

Tables Tables of sequence data

Graphs Graphical displays of some, all, or typical sequences

Sequence statistics Descriptive measures of various characteristics of sequences

Sequence similarity statistics Measures of similarity or dissimilarity between sequences

Sequence statistics and similarity statistics might be used in subsequent analyses – such as regression models, cluster analysis or multidimensional scaling.

The SQ-Ados

- The SQ-Ados are a collection of user written programs to calculate sequence statistics and similarity statistics, and to provide graphical displays.
- Available since 2006 (Brzinsky-Fay et al., 2006; Kohler et al., 2006),
- New developments:
 - Various new sequence statistics
 - Interface to SADI (Halpin, 2014)
 - Similarity statistics for strings (see also Reiff, 2010; Barker, 2014; Provalis Research, 2016)
 - New graphical displays
 - A tool for record linkage

This talk presents the entire package, with an emphasis on the new developments.

SQ-Setting

- The SQ-Ados assume that the data set is long and was sq-setted by the user
- The Stata command `reshape` should be applied for making wide data long
- The command `sqset` is used to SQ-set the data

```
sqset elemvar idvar ordervar [ , trim rtrim ltrim  
keeplongest ]
```

👉 Example

```
. sqset st id order, trim  
    element variable:  st, 1 to 5  
    identifier variable: id, 1 to 500  
    order variable:   order, 1 to 36
```


Inhalt

- 1 Introduction
- 2 Tables**
- 3 Graphs
- 4 Sequence statistics
- 5 Sequence Similarity Statistics
- 6 Applications

Description of the data set

`sqdes [if][in][, so se gapinclude subsequence(a,b)]`

👉 Example

```
. sqdes
  # of observed sequences: 500
overall # of obs. elements: 5
      max sequence length: 36
  # of producible sequences: 1.455e+25
```

Observations	Sequences	% of observed	Cum.
1	309	61.8	61.8
2	22	4.4	66.2
3	5	1	67.2
4	3	.6	67.8
5	2	.4	68.2
7	1	.2	68.4
9	1	.2	68.6
10	1	.2	68.8
18	1	.2	69
28	1	.2	69.2
38	1	.2	69.4
Total	347	69.4	

Frequency table of sequences

`sqtab [if][in][, ranks(numlist) so se gapinclude
subsequence(a,b)]`

👉 Example

```
. sqtab, ranks(1/5)
```

Sequence-Pa ttern	Freq.	Percent	Cum.
4:36	38	36.89	36.89
3:36	28	27.18	64.08
5:36	18	17.48	81.55
4:12 1:24	10	9.71	91.26
5:12 1:24	9	8.74	100.00
Total	103	100.00	

Same order and same elements similarity

The standard options `so` and `se` group sequences using a simple definition of similarity

☞ Examples

```
. sqtab, ranks(1/5) so
```

Sequence-Order	Freq.	Percent	Cum.
4	38	27.14	27.14
3	28	20.00	47.14
4 3	28	20.00	67.14
5 1	28	20.00	87.14
5	18	12.86	100.00
Total	140	100.00	

```
. sqtab, ranks(1/5) se
```

Sequence-Elements	Freq.	Percent	Cum.
3 4	68	30.63	30.63
1 3 4	42	18.92	49.55
3 4 5	38	17.12	66.67
4	38	17.12	83.78

Inhalt

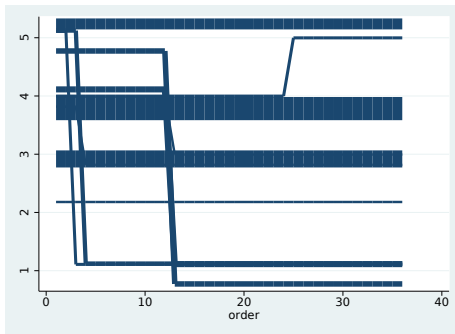
- 1 Introduction
- 2 Tables
- 3 Graphs**
- 4 Sequence statistics
- 5 Sequence Similarity Statistics
- 6 Applications

Parallel-Coordinates-Plot

```
sqparcoord [if][in][ , ranks(numlist) so offset(##)
wlines(##) gapinclude twoway_options]
```

👉 Example

```
. sqset st id order, trim
. sqparcoord, ranks(1/10) offset(.5) wlines(7)
```



Sequence-Index-Plots

```
sqindexplot [if][in][ , ranks(numlist) se so  
order(varname) by(varlist) color(colorstyle) gapinclude  
twoway_options]
```

👉 Example

```
. sqindexplot, rbar order(sqdim) by(cluster, rows(1)) legend(pos(6) r
```

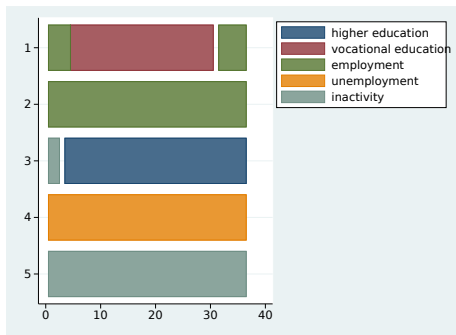


Sequence-Modal-Plots (New)

```
sqmodalplot [ if ][ in ][ , over(varname) so order(varname)  
by(varname) color(colorstyle) gapinclude subsequence(a,b)  
tie(keyword) twoway_options ]
```

👉 Example

```
. sqmodalplot, over(cluster)
```

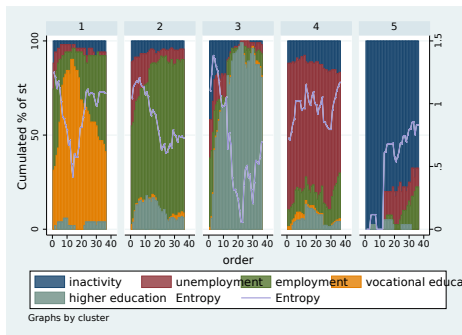


Sequence-Percentage-Plot (New)

```
sqpercentageplot [if][in][ , entropy nosecond  
baropts(barlook options) lopts(connect options)  
l2opts(connect options) twoway_options]
```

👉 Example

```
. sqpercentageplot, entropy by(cluster, rows(1)) legend(pos(6) rows(2))
```



Inhalt

- 1 Introduction
- 2 Tables
- 3 Graphs
- 4 Sequence statistics**
- 5 Sequence Similarity Statistics
- 6 Applications

SQ-Egen functions

Sequence statistics are calculated using a suite of function for `egen`

```
egen [type] newvar = sqfcn() [ , options]
```

`sqallpos()` Number of sub-sequences with a specified pattern within a sequence (new)

`squelemcount()` Number of elements in a sequence

`sqepicount()` Number of episodes in a sequence

`sqfirstpos()` Position, where a specified pattern is first found (new)

`sqfreq()` Frequency of a sequence of this type (new)

`sqgapcount()` Number of „gaps“ in a sequence

`sqgaplength()` Overall length of all episodes with gaps

`sqlength()` Length of a sequence

`sqranks()` Position of the sequence in a rank table (new)

`sqsuccesss()` „Success“ of a sequence (new; see Manzoni, 2016)

`sqtstring()` String-representation of a sequence (new)

Common options

The SQ-egen-commands share a set of common options:

`gapinclude` Calculate the statistic including “gaps” (i.e. positions within the sequence wherer the element is missing)

`subsequence(a,b)` Calculate the statistic for a subsequence between positions a and b

`pattern(spec)` is used in some function to specify a specific kind of sequence:

Examples

Sequenz	Pattern
1-2-1	<code>pattern(1 2 1)</code>
1-5-5-1	<code>pattern(1 5:2 1)</code>
1-4-4-4-2-2-1-3-3-3-3-	<code>pattern(1 4:3 2:2 1 3:4)</code>

Inhalt

- 1 Introduction
- 2 Tables
- 3 Graphs
- 4 Sequence statistics
- 5 Sequence Similarity Statistics**
- 6 Applications

A primer on sequence similarity

Consider the following sequences of latin letters:

r	e	g	r	e	s	s	i	o	n	
p	r	o	g	r	e	s	s	i	o	n

Note that the two words seem similar despite the fact that there is only one position with identical elements.

Levensthein-distance (Levenshtein, 1966)

The Levensthein-distance is the minimum number of substutions and “indels” necessary to make a pair of sequences identical.

- Substitution (S)

$$\begin{array}{cccccccccccc} r & e & g & r & e & s & s & i & o & n & x \\ p & r & o & g & r & e & s & s & i & o & n \\ \hline S & S & S & S & S & S & 0 & S & S & S & S \end{array} = 10 \cdot S$$

- Insertion/Deletion (indel) (D)

$$\begin{array}{cccccccccccc} p & r & o & e & g & r & e & s & s & i & o & n \\ p & r & o & e & g & r & e & s & s & i & o & n \\ \hline I & 0 & I & I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} = 3 \cdot I$$

- $\sum_{k=1}^K s_k + d_k \stackrel{!}{=} \min$

$$\begin{array}{cccccccccccc} p & r & e & g & r & e & s & s & i & o & n \\ p & r & o & g & r & e & s & s & i & o & n \\ \hline I & 0 & S & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} = 1 \cdot I + 1 \cdot S$$

Variants

- Hamming Distance (Hamming, 1950)
- Dynamic Hamming Distance (Lesnard, 2010)
- Time Warp Edit Distance (Marteau, 2009)
- Elzinga's Combinatorial Measures (Elzinga, 2003, 2005, 2007)

👉 **Note** The Hamming Distance is a special case of the Levenshtein Distance. The Levenshtein-Distance is the standard distance measure for „Optimal Matching“ (Abbott and Tsay, 2000). The SQ-Ados use an implementation of the „Needleman-Wusch-Algorithm“ (Needleman and Wunsch, 1970) to compute the Levenshtein Distance.

sqom

```
sqom [ if ][ in ][ , common_options name(varname) full  
idealttype(pattern) refseqid(spec) sadi(sadicmd) ]
```

New:

```
sqstrlev [ if ][ in ][ , common_options ]
```

Common options:

```
indelcost( # ) subcost( # | rawdistance | matexp | matname ) k( # )
```

Examples (numeric sequences)

```
. sqom
Distance Variable saved as _SQdist

. matrix sub = 0,8,7,3,2\8,0,8,7,3\7,8,0,8,7\3,7,8,0,7\2,3,7,7,0

. sqom, indelcost(3) subcost(sub) idealtypes(3:10 4:10)
Distance Variable saved as _SQdist

. sqom, full
Perform 60031 Comparisons with Needleman-Wunsch Algorithm
Running mata function
Distance matrix saved as SQdist

. sqom, full k(2)
Perform 60031 Comparisons with Needleman-Wunsch Algorithm
Running mata function
Distance matrix saved as SQdist

. sqom, sadi(oma)
Running plugin; Please cite Brandan Halpin's work
Normalising distances with respect to length
(0 observations deleted)
347 unique observations
Distance matrix saved as SQdist

. sqom, sadi(hollister) timecost(3) localcost(1)
Running plugin; Please cite Brandan Halpin's work
Normalising distances with respect to length
347 unique observations
Distance matrix saved as SQdist
```

Examples (strings)

```
. use mdbV2, replace
. sqstrlev prename
. sqstrlev prename, indelcost(1) subcost(1.5) ignorecase asciiletters
```

Inhalt

- 1 Introduction
- 2 Tables
- 3 Graphs
- 4 Sequence statistics
- 5 Sequence Similarity Statistics
- 6 Applications**

Grouping

Sequences can be grouped according to their similarity by applying cluster analysis on the distance matrix created by `sqom`, `full` or `sqom`, `sadi()`. `sqclusterdat` assists to add the cluster results to the (long) sequence dataset.

Example

```
. sqom, sadi(oma)
. sqclusterdat
. clustermat wardslinkage SQdist, name(myname) add
. cluster generate cluster = groups(5)
. sqclusterdat, return keep(cluster myname*)
```

Sacling (new)

Sequences can be scaled along one (or more) dimensions according to their similarities by applying multidimensional scaling on the distance matrix created by `sqom, full` or `sqom, sadi()`. `sqmdsadd` assists to add the MDS results to the (long) sequence dataset.

👉 Example

```
. sqom, sadi(oma)
. mdsmat SQdist
. predict sqdim, saving(om1)
. sqmdsadd using om1
```

Identification of nearest neighbours (new)

The egen function `sqstrnn()` creates a new variable holding the string(s) that (are) most similar to a string-value.

```
egen [type] newvar = sqstrnn() [ , max(#) ignorecase  
asciilettersonly soundex sqom-options]
```

Example

```
. egen nn = sqstrnn(prename), max(3) standard(none)  
(153 missing values generated)  
. list prename nn in 1/7 if !mi(nn)
```

	prename	nn
1.	Achim	Jochim
2.	Adalbert	Albert
5.	Adolf	Adolph; Ludolf; Rolf; Rudolf; Wolf
6.	Adolph	Adolf
7.	Aenne	Anne

Record Linkage (new)

The idea of a string's nearest neighbour can be used to merge records from two files using strings that are similar. This idea is implemented in the new command `sqstrmerge`:

```
sqstrmerge mergetype varlist, max(maxlist)  
[sqstrlev-options merge-options]
```

The syntax mirrors official Stata's `merge`-command. The required option `max()` controls the maximal acceptable distance for the approximative merge. The higher the values, the higher the risk to merge a wrong match.

`sqstrmerge` creates the variables `_var_using` and `_var_distance` to control the results of the merge.

Example of sqstrmerge

```
. sqstrmerge m:1 county year using ../srlt_population ///  
> , max(6 0) standard(none)
```

_county_distance	_merge			Total
	master on	using onl	matched (
0	0	0	2,462	2,462
1	0	0	173	173
2	0	0	305	305
3	0	0	132	132
4	0	0	418	418
5	0	0	90	90
6	0	0	67	67
.	274	1,502	0	1,776
Total	274	1,502	3,647	5,423

```
. bysort county (year): keep if _n==1  
(5,128 observations deleted)
```

```
. list county _county_using if _county_distance==6
```

	county	_county_using
210.	Schwedt Oder, Stadtkreis	Schwedt/ Oder, Stadt
247.	Weimar, Land	Weimar
273.	k Angermuende	Angermünde

- Abbott, A. and A. Tsay. 2000. Sequence Analysis and Optimal Matching Methods in Sociology. Review and Prospect. *Sociological Methods and Research* 29(1): 3–33. <http://smr.sagepub.com/content/29/1/3.full.pdf+html>.
URL <http://smr.sagepub.com/content/29/1/3.full.pdf+html>
- Barker, M. 2014. STRDIST: Stata module to calculate the Levenshtein distance, or edit distance, between strings. Statistical Software Components, Boston College Department of Economics.
URL <https://ideas.repec.org/c/boc/bocode/s457547.html>
- Brzinsky-Fay, C., U. Kohler, and M. Luniak. 2006. Sequence Analysis With Stata. *Stata Journal* 6(4): 435–460.
- Elzinga, C. H. 2003. Sequence Similarity. A Nonaligning Technique. *Sociological Methods and Research* 32(1): 3–29.
<http://smr.sagepub.com/content/32/1/3.full.pdf>.
URL <http://smr.sagepub.com/content/32/1/3.full.pdf>
- . 2005. Combinatorial Representations of Token Sequences. *Journal of Classification* 22: 87–118. Kostenpflichtig.
- . 2007. Sequence Analysis: Metric Representations of Categorical Time Series. Department of Social Science Research Methods, Vrije Universiteit Amsterdam.
- Halpin, B. 2014. SADI: Stata module to compute Sequence Analysis Distance Measures. Statistical Software Components, Boston College Department of Economics.
URL <http://EconPapers.repec.org/RePEc:boc:bocode/s458056>.
- Hamming, R. W. 1950. Error-detecting and error-correcting codes. *Bell System Technical Journal* 29: 147–160.
URL <http://guest.engelschall.com/~sb/hamming/?page=3>
- Kohler, U., M. Luniak, and C. Brzinsky-Fay. 2006. SQ: Stata module for sequence analysis. Statistical Software Components, Boston College Department of Economics.
- Lesnard, L. 2010. Setting Cost in Optimal Matching to Uncover Contemporaneous Socio-Temporal Patterns. *Sociological Methods & Research* 38(3): 389–419.
URL <http://smr.sagepub.com/content/38/3/389.abstract>
- Levenshtein, V. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady* 10(4): 707–710.
- Manzoni, A. 2016. Binary Sequence Dynamics applied to Career Success. Accepted presentation for International Conference on Sequence Analysis and Related Methods, Lausanne, June 8–10.
- Marteau, P.-F. 2009. Time Warp Edit Distance with Stiffness Adjustment for Time Series Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 31(2): 306–318.
- Needleman, S. and C. Wunsch. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal for Molecular Biology* 48(3): 443–453.
- Provalis Research. 2016. WordStat for Stata. Software.
URL <http://provalisresearch.com/products/content-analysis-software/wordstat-for-stata/>
- Reiff, M. 2010. STRGROUP: module to match strings based on their Levenshtein edit distance. Statistical Software Components, Boston College Department of Economics.
URL <https://ideas.repec.org/c/boc/bocode/s457151.html>
- Schnell, R., T. Bachteler, and S. Bender. 2004. A Toolbox for Record Linkage. *Austrian Journal of Statistics* 33: 125–133.