Day 4: From Words to Deeds

Johann-Mattis List (DLCE, MPI-EVA Leipzig)

1 Overview

On this last day of our small lecture series, we want to look into the practical consequences of what has been discussed so far. We will focus on two basic tools which I use in my own research on computer-assisted language comparison, LingPy (List and Forkel 2022b) and EDICTOR (List 2021a). Both tools are being actively developed and are freely accessible online. While LingPy is a software library for quantitative tasks in historical linguistics which requires basic programming skills, EDICTOR can in theory be used and applied without any deeper knowledge of programming. However, in order to get started with EDICTOR, it is still important to have a good account of certain data formats that are commonly used, and the major ideas behind the tool, which may at times not be exactly what historical linguists expect, specifically those linguists who have been used to working out their etymologies in very free formats or in prose form. Before we discuss these two tools in detail, however, we will have a look at general principles and recommendations for the handling of data in historical linguistics.

2 Data Preparation

Linguists who prepare their own data often come up with their own, seemingly convenient, formats for data representation which often have huge disadvantages in terms of transparancy and interoperability. It is therefore generally recommended to pay close attention to the basic formats underlying LingPy and EDICTOR on the one hand and the Cross-Linguistic Data Formats (CLDF) initiative on the other hand (Forkel et al. 2018), which are by now largely compatible, with tools that allow for a facilitated conversion from one format to the other (Forkel and List 2020, List 2021b).

2.1 General Rules

Most data annotation in linguistics, even the annotation of texts, can be conveniently represented in a table, and scholars tend to make extensive use of tables – using different kinds of spreadsheet software – when annotating their data. The most obvious failure when preparing data in tables is to include multiple different types of information into one cell. Thus, if a word has a variant, scholars often place it into one cell in their tables and separate the entries by a comma, a colon, a tilde, a dash, or at times even by a back-slash, often even using all of these separators inconsistently for the same dataset. A first and general rule for data creation is therefore (1).

(1) Only one type of information should be put into one cell in a spreadsheet.**

This rule is extremely important and should not be negotiated.In our experience resulting from working with a large number of differently coded datasets, which we retro-standardized as part of the Lexibank project (List et al. 2022a), annotation errors are usually inevitable, even when scholars try to be consistent. Computers are not like humans, and if one wants to profit from computers to ease one's work, one needs to understand that computers cannot guess whether commas and slashes are used with or without without semantic difference when listing word variants. Even humans often have a hard time to understand the meaning of different separators used in the same dataset, even if they created the data themselves.

A more general rule deriving from this first rule is the rule (2).

(2) All information valid for a given analysis needs to be consistently annotated.**

This means, for example, that, if root alternation is important for one's reconstruction and cognate decisions, one needs to think of ways to represent root alternation in consistent markup. If one's data contains reflexes of an alternating protoform **ka-* vs. **ku-*, for example, it is not sufficient to simply write **ka-* ~ **ku-* and listing the reflexes, assuming that readers will understand which reflex stems from which of the two alternants. Instead, Two proto-forms should be listed, the variants should be assigned to the correct proto-form from which they evolve, and the additional information should be given that **ka-* and **ku-* are variants of the same root. This practice is rarely used in etymological dictionaries, and therefore also often disregarded in databases. It is, however, obvious that it is the only way to transparently list what reflex stems from which proto-variant. In a broader sense, this is not a only matter of a more computational approach to historical linguistics, but rather a matter of improving our common practices in historical linguistics, which have been for too long a time based on very lax guidelines.

What is the problem with lax guidelines in a scientific discipline like historical linguistics?

2.2 Representing Data in Tables

Apart from texts, tables are the most frequently used data structures in linguistics. Specifically in historical linguistics, scholars tend to represent all kinds of data in tables, ranging from wordlists, via cognate sets, up to regular sound correspondence patterns. It is therefore not surprising that computer-assisted tools for historical language comparison also invoke a tabular format. Not all tabular formats used in the field, however, are useful for data analysis.

Concepts	**Languages**				
	English	**German**	**Dutch**	**Danish**	**Swedish**
"hand"	hænd	hant	hɑnt	hʌnˀ	hanːd
"ashes"	æʃ	aʃə	ɑs	asg	asːka
"bark"	bɑːrk	rɪndə	bɑst	bɑːg	barːk
...	...	...	...	...	...

A: Concepts/Languages table format.

Concepts	**Languages**				
	English	**German**	**Dutch**	**Danish**	**Swedish**
"bark"	bɑːrk	rɪndə	bɑst	bɑːg	barːk
"bark"		bɔrkə			

B: Using additional rows for concepts.

Concepts	**Languages**				
	English	**German**	**Dutch**	**Danish**	**Swedish**
"bark"	bɑːrk	rɪndə, bɔrkə	bɑst	bɑːg	barːk

C: Using separators inside cells.

Concepts	**Languages**					
	English	**German**	**German (b)**	**Dutch**	**Danish**	**Swedish**
"bark"	bɑːrk	rɪndə	bɔrkə	bɑst	bɑːg	barːk

D: Additional language column.

As an example, consider the widely used format by which languages are represented in columns and concepts are represented in rows, which is shown in the figure A above. This format lacks flexibility, as there is only one piece of information that we can give for each concept in a given language. When dealing with more languages it becomes more and more impractical, since it is difficult to inspect all languages on a screen, specifically because scrolling horizontally is always more difficult than scrolling vertically.

Despite its shortcomings, the "language-columns-concepts-rows" format is one of the most widely used formats in historical linguistics, and scholars even have often extended it in order to allow for the display of cognacy among the words listed in the individual cells, or to make it possible to present more than one word form as the translatioon of a given concept in a given language. Here, the problems of the format become even more evident, since cognate information is often added in an ad-hoc manner. Thus, for the handling of synonyms, the STARLING format (Starostin 2000) adds additional rows for individual concepts, as shown in figure B above. Other studies use commas or other separators to display more than one entry in the same cell (a format commonly used to import data into the ReFLeX database, Segerer and Flavier 2015, as shown in figure C. We even find cases where additional columns for languages with synonyms are added, as illustrated in figure D.

When it comes to the annotation of cognate sets inside these tables, the formats become even more creative, ranging from color-coding to represent cognate words, via multi-sheet formats, up to cases where scholars even binarize their cognate data manually, instead of having this done automatically (compare examples in Forkel et al. 2018).

Software packages like STARLING try to circumvent the problems resulting from the basic tabular format by allowing for additional columns which add additional information for the same language. As a result, STARLING tables currently have three columns for each language, one for the original word form, one for the cognate judgments, and one for comments. LingPy and EDICTOR, the two packages which provide the major methodology discussed in this study, however, employ a different approach which greatly increases the flexibility of the format.

The major principle of this approach is to reserve *one row* in the spreadsheet for exactly *one word form*. Additional information for each word form is provided in additional columns (which can be flexibly added by the user). The content of each column in these EDICTOR-spreadsheets is given in the header of the file, with the first column being reserved for a numeric ID which should be greater than 0. This column should be called ID. Additional columns can be flexibly ordered, but should provide basic information on the name of the language (usually called DOCULECT), on the concept that is expressed by the word (called CONCEPT), and – in order to be able to compare words – a column providing the word form segmented into individual sound units (called 'TOKENS'). Depending on the analysis one wants to carry out, additional columns need to be supplied, but they can be empty when starting with an analysis. Thus, in order to store cognate sets, a column for full cognates (often called COGID) or a column for partial cognates (often called COGIDS) should be supplied. In order to annotate morphemes with the help of morpheme glosses, a column storing this information should be added (often called MORPHEMES). In order to store phonetic alignments, an alignment column (often called ALIGNMENT) should be added.

ID	DOCULECT	CONCEPT	VALUE	FORM	TOKENS	BORROWING	COGID
3631	East_Futuna	above	à/luga/	luga	l u g a	0	1382
284	Wallisian	above	'o/luga/	luga	l u g a	1	1382
5391	Futuna_Aniwa	above	weihlunga	weihlunga	w e i + ʰl u ŋ a	0	1382
761	Maori	above	i runga	i runga	i _ r u ŋ a	0	1382
3332	North_Marquesan	above	'una	'una	ʔ u n a	0	1382
4214	Mele-Fila	all	euči	euči	e u tʃ i	0	1115
3917	Pukapuka	all	katoa(toa)	katoa	k a + t o a	0	293
560	Proto-Polynesian	yellow	*reŋareŋa, *felo(-felo)	*reŋareŋa	r e ŋ a + r e ŋ a	0	162
560	Proto-Polynesian	yellow	*reŋareŋa, *felo(-felo)	*felo	f e l o	0	230

The figure above shows a sample table for a dataset on Polynesian languages (List et al. 2018), in which an extra column for borrowings (BORROWING) was added, in which information on borrowing is stored in a binary format (if a word is considered to be borrowed, it is given a 1, otherwise a 0). Additionally, a column for the original value in the original data has been added (VALUE), as well as a column showing the intermediate word format extracted from the value without segmentation (FORM).

If one wants to prepare one's data in this format, it is recommend to start with a spreadsheet editor (such as Excel, LibreOffice, or GoogleSheets), where one inserts the values as indicated above. In order to convert this spreadsheet into the required tab-separated value format, the easiest way is to create an empty file with the ending `.tsv`, to open the file with a text editor (a program like Word should not be used), and to copy-paste all columns and rows with values (individual cells can be empty of course, but no rows and no columns should be fully empty and all columns should have a header and all rows should have an ID) into this file. Having done so, the data can be directly accessed with LingPy from within Python scripts, or loaded into EDICTOR, where one can directly manipulate it.

What is so important about tables when discussing data in historical linguistics?

3 EDICTOR

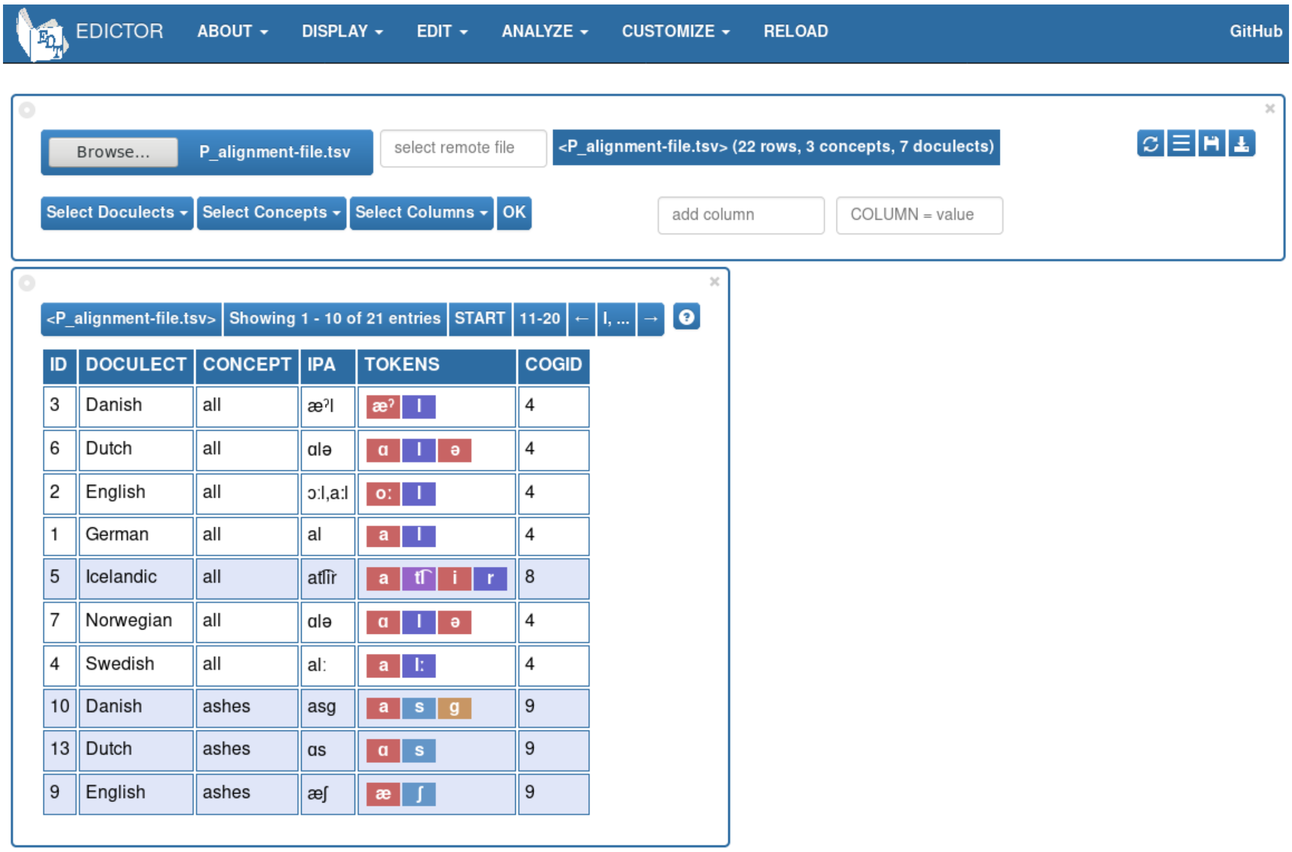
The EDICTOR (`https://digling.org/edictor`, List 2017, List 2021a) is a web-based tool for the curation of etymological data in historical linguistics. The tool has a modular structure which is organized in the form of *panels*. Panels are windows which open once data was loaded into the tool, and users can investigate their data by loading different panels at the same time. The basic panel, the WORDLIST panel, is used to edit data, similar to the way in which this can be done in a spreadsheet editor. Additional panels help to annotate cognates, to align words, or to analyze the data interactively.

3.1 Getting Started with the EDICTOR

What users need in order to use the tool is a text-file encoded in the form in which it was discussed in the previous section, that is, a file in the standard format in which each word is given a row, and a header informs which type of data a certain column contains. In order to use the tool, users need to open the website, located at `https://digling.org/edictor` (for the development version, see `https://lingulist.de/edev` in their browser and drag their file into the BROWSE button which which shows up on the top left of the window.

EDICTOR is written in plain JavaScript. When invoking the tool in this form, the code runs entirely on the system of the user and no data will be send to any servers. As a result, no data can be stolen, users cannot be tracked, and nobody even knows that one is using the tool. Thus, there is no need to be afraid that using the EDICTOR tool will result in data theft in any form. Apart from the fact that it is unlikely that anybody would appropriate somebody else's collected data, it is also not possible, as long as users load a file from their computer into the system and later also export it back from there.

The following figure shows the typical Wordlist panel of the EDICTOR which opens after loading a dataset.



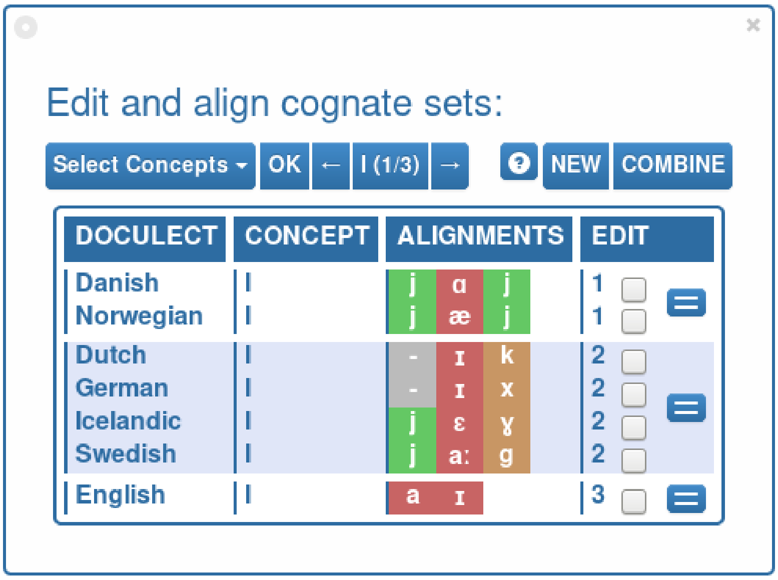
Why is it not possible that the EDICTOR system will save users' data when somebody uses it through the web interface?

3.2 Features of the EDICTOR

The basic structure of the EDICTOR tool is based on *panels*, which allow users to inspect, modify, or analyze their data in specific regards. Panels can also be used to interact together in solving certain tasks. Thus, there is a panel for the annotation of cognate sets, a panel for the annotation of partial cognate sets (which will probably disappear from a future version or frozen and no longer developed actively), and there is a panel to edit the basic data in tabular form and a rather new panel to edit morpheme glosses and partial cognates at the same time.

For the inspection and analysis of the data, there is a panel allowing to check the phonology of a given language variety (along with the possibility to create an IPA chart), there is a panel to inspect various forms of colexifiations (which will be significantly altered in future versions), and there are two important new panels, one showing the cognate set distribution across concepts and languages, allowing also for an export to Nexus format, and one allowing scholars to investigate sound correspondence patterns in their data.

The following figure shows the Cognates panel of the EDICTOR, which can be used to annotate cognate sets in a consistent manner.



What is the difference between annotation and inspection when it comes to the EDICTOR panels?

4 LingPy

LingPy (`https://lingpy.org`, List and Forkel 2022b) is a Python library for quantitative tasks in historical linguistics. With the help of LingPy, several methods which are important for the analysis of wordlists and etymological data can be carried out automatically, including *phonetic alignment analyses* and automated cognate detection (List 2014), basic phylogenetic analyses using the UPGMA (Sokal and Michener 1958) or Neighbor-joining algorithm (Saitou and Nei 1987), and several tasks that help to manipulate word list data in various forms. A couple of years ago, we started to extend LingPy with LingRex (`https://pypi.org/project/lingrex`, List and Forkel 2022c), a Python library dedicated to various tasks related to linguistic reconstruction, which specifically offers access to the new algorithms for correspondence pattern detection (List 2019), and phonological reconstruction (List et al. 2022b). An important feature of LingPy and LingRex is that the basic formats that the libraries read and write are directly compatible with the tab-separated wordlist formats required by the EDICTOR tool. As a result, we now use EDICTOR and LingPy/LingRex in combination, and preprocess a given dataset automatically in order to later refine it manually (Wu et al. 2020).

4.1 Getting Started with LingPy

Installing LingPy and LingRex should by now no longer be a problem for those who have some experience with the installation of Python libraries. Both libraries work without problem on Windows, MacOS, and Linux systems. Numerous tutorials exist, both in the form of articles (List et al. 2018) and in the form of online tutorials (`https://lingpy.org`), and we try to regularly update the basic documentation of our software packages. Additional information can also be found in our blog, where we provide tutorials and howtos on computer-assisted language comparison on a regular basis, with at least one contribution per month (`https://calc.hypotheses.org`). In case of questions it is never wrong to write me an email to inquire, or to file an issue on our GitHub pages (`https://github.com/lingpy/lingpy` and `https://github.com/lingpy/lingrex`).

Where should one start if one does not know anything about programmiing at all?

4.2 Features of LingPy

As mentioned, LingPy and LingRex offer a larger array of methods and implementations for algorithms that were proposed to solve certain problems in historical linguistics, such as the detection of cognates in multilingual wordlists, the alignment of sound sequences, or the reconstruction of phylogenetic trees, as well as initial approaches to phonological reconstruction and borrowing detection (List and Forkel 2022a). It is furthermore important to note that the models for data handling which we have discussed in the course so far are all directly implemented (or represented) in LingPy and thus help us to make sure that our theoretical approaches to data representation and modeling prove useful in practice. In the future we will try to expand LingPy and LingRex further, adding specifically new methods to borrowing detection and developing new methods for the automated segmentation of words into morphemes.

Why would the segmentation of words into morphemes be useful in historical linguistics?

References

Forkel, R. and J.-M. List (2020). "CLDFBench. Give your Cross-Linguistic data a lift." In: *Proceedings of the Twelfth International Conference on Language Resources and Evaluation.* "LREC 2020" (Marseille). Luxembourg: European Language Resources Association (ELRA), 6997–7004.

Forkel, R., J.-M. List, S. J. Greenhill, C. Rzymski, S. Bank, M. Cysouw, H. Hammarström, M. Haspelmath, G. A. Kaiping, and R. D. Gray (2018). "Cross-Linguistic Data Formats, advancing data sharing and re-use in comparative linguistics." *Scientific Data* 5.180205, 1–10.

List, J.-M. (2014). *Sequence comparison in historical linguistics.* Düsseldorf: Düsseldorf University Press.

— (2017). "A web-based interactive tool for creating, inspecting, editing, and publishing etymological datasets." In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics. System Demonstrations.* Valencia: Association for Computational Linguistics, 9–12.

— (2019). "Automatic inference of sound correspondence patterns across multiple languages." *Computational Linguistics* 45.1, 137–161.

— (2021a). *EDICTOR. A web-based tool for creating, editing, and publishing etymological datasets. Version 2.0.0.* Leipzig: Max Planck Institute for Evolutionary Anthropology. URL: `https://digling.org/edictor`.

— (2021b). "PyEDICTOR [Python library, Version 0.3.0]."

List, J.-M. and R. Forkel (2022a). "Automated identification of borrowings in multilingual wordlists [version 3; peer review: 4 approved]." *Open Research Europe* 1.79, 1–11.

— (2022b). *LingPy. A Python library for quantitative tasks in historical linguistics [Software Library, Version 2.6.9].* Leipzig: Max Planck Institute for Evolutionary Anthropology.

— (2022c). *LingRex: Linguistic reconstruction with LingPy.* Leipzig: Max Planck Institute for Evolutionary Anthropology.

List, J.-M., R. Forkel, S. J. Greenhill, C. Rzymski, J. Englisch, and R. D. Gray (2022a). "Lexibank, A public repository of standardized wordlists with computed phonological and lexical features." *Scientific Data* 9.316, 1–31.

List, J.-M., N. W. Hill, and R. Forkel (2022b). "A new framework for fast automated phonological reconstruction using trimmed alignments and sound correspondence patterns." In: *Proceedings of the 3rd Workshop on Computational Approaches to Historical Language Change.* (Dublin, 26–05/27/2022). Association for Computational Linguistics. Dublin, 89–96.

List, J.-M., M. Walworth, S. J. Greenhill, T. Tresoldi, and R. Forkel (2018). "Sequence comparison in computational historical linguistics." *Journal of Language Evolution* 3.2, 130–144.

Saitou, N. and M. Nei (1987). "The neighbor-joining method: A new method for reconstructing phylogenetic trees." *Molecular Biology and Evolution* 4.4, 406–425.

Segerer, G. and S. Flavier (2015). *RefLex: Reference Lexicon of Africa.* Version 1.1. URL: `http://reflex.cnrs.fr`.

Sokal, R. R. and C. D. Michener (1958). "A statistical method for evaluating systematic relationships." *University of Kansas Scientific Bulletin* 28, 1409–1438.

Starostin, S. A. (2000). *The STARLING database program.* Moscow: RGGU.

Wu, M.-S., N. E. Schweikhard, T. A. Bodt, N. W. Hill, and J.-M. List (2020). "Computer-Assisted Language Comparison. State of the Art." *Journal of Open Humanities Data* 6.2, 1–14.