Computer-Assisted Language Comparison: State of the Art

By comparing the languages of the world, we gain invaluable insights into human prehistory, predating the appearance of written records by thousands of years. The traditional methods for language comparison are based on manual data inspection. With more and more data available, they reach their practical limits. Computer applications, however, are not capable of replacing experts' experience and intuition. In a situation where computers cannot replace experts and experts do not have enough time to analyse the massive amounts of data, a new framework, neither completely computer-driven, nor ignorant of the help computers provide, becomes urgent. Such frameworks are well-established in biology and translation, where computational tools cannot provide the accuracy needed to arrive at convincing results, but do assist humans to digest large data sets. In this talk, we will illustrate what we consider the current state of the art of computer-assisted language comparison, by presenting a workflow that starts from raw data and leads up to a stage where sound correspondence patterns across multiple languages have been identified and can be readily presented, inspected, and discussed. We illustrate this workflow with help of a dataset on Hmong-Mien languages, which has so far not yet been analyzed in this way. Our illustration is furthermore accompanied by Python code and instructions on how to make use of additional web-based tools we developed, so that users can replicate our workflow or apply it for their own purposes.

1 Introduction

1.1 The Gap between Computational and Traditional Historical Linguistics

The proposal of new, fancy, and shiny quantitative methods applied to handle problems in historical linguistics has created a gap between what one could call "classical" approaches to historical language comparison and the "new and innovative" automatic approaches. Classical linguists are often skeptical of the new approaches, partly because the results differ from those achieved by classical methods (Anthony and Ringe 2015, Holm 2007), but also because the majority of the new approaches work in a black box fashion and do not allow inspecting the concrete findings in detail. Computational linguists, on the other hand, complain about classical historical linguists' lack of consistency when applying the classical methods.

1.2 Computer-Assisted Disciplines

The use of computer applications in historical linguistics is steadily increasing. With more and more data available, the classical methods reach their practical limits. At the same time, computer applications are not capable of replacing experts' experience and intuition, especially when data are sparse. If computers cannot replace experts and experts do not have enough time to analyse the massive amounts of data, a new framework is needed, neither completely computer-driven, nor ignorant of the assistance computers afford. Such computer-*assisted* frameworks are well-established in biology and translation. Current machine translation systems, for example, are efficient and consistent, but they are by no means accurate, and no one would use them in place of a trained expert. Trained experts, on the other hand, do not necessarily work consistently and efficiently. In order to enhance both the quality of machine translation and the efficiency and consistency of human translation, a new paradigm of computer-assisted translation has emerged (Barrachina et al. 2008: 3).

1.3 Computer-Assisted Language Comparison

Following the idea of computer-assisted frameworks in translation and biology, a framework for computer-assisted language comparison (CALC) could be the key to reconcile classical and computational ap-

proaches in historical linguistics. Computational approaches may still not be able to compete with human experts, but when used to pre-process the data with human experts systematically correcting the results, they can drastically increase both the efficiency and the consistency of the classical comparative method.

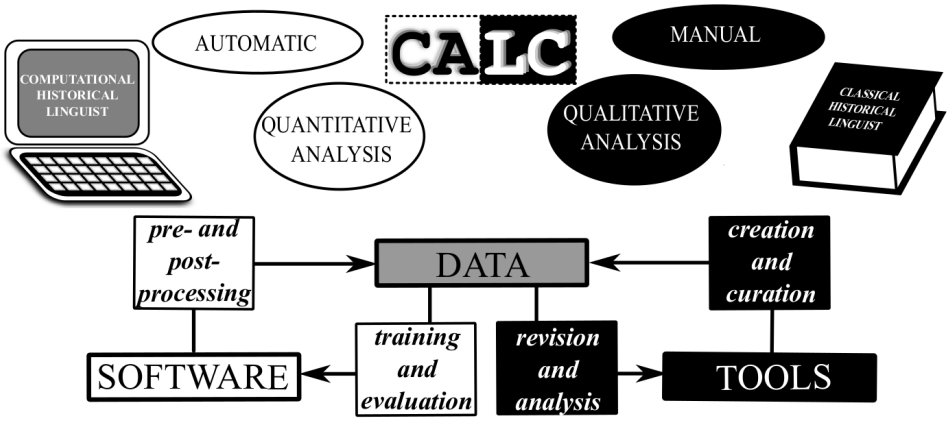

Figure 1: Basic idea of data managment within the CALC framework.

The basic idea behind computer-*assisted* as opposed to computer-*based* language comparison is to allow scholars to do qualitative and quantitative research are done at the same time. In order to allow scholars to do this, **data must always be available in *machine-* and *human-readable* form**. Figure 1 shows a tentative workflow for the CALC framework, in which data is constantly passed back and forth between computational and classical linguists.

Three different aspects are essential for this workflow:

(a) New software allows for the application of transparent methods which increase the accuracy and the application range of current methods and also treat the peculiarities of specific language families (like, e.g., Sino-Tibetan).

(b) Interactive tools provide an interface between human and machine, allowing experts to correct errors and to inspect the automatically produced results in detail.

(c) Specific data is used to test and train the software algorithms.

2 Workflows for Computer-Assisted Language Comparison

2.1 Overview

Our workflows for computer-assisted language comparison have so far been intensively tested on a small set of 8 Burmish languages, which we investigated in collaboration with Nathan W. Hill, who was responsible for the qualitative investigation of the data and for the common discussion of new computer-assisted methods which were then implemented by Johann-Mattis List (see Hill and List 2017 for an exemplary discussion of some of the new approaches). Our experience with the Burmish project by now allows us to set up a first workflow that starts from raw data and leads up to the explicit identification of correspondence patterns across multiple languages. At the moment, List and Hill develop the workflow further to account also for (semi)-automatic reconstructions, but in this talk, only the identification of correspondence patterns will be discussed.

2.2 Details of the Workflow

Our workflow currently comprises 5 different stages, in which we successively lift linguistic data from their raw form in which we can find them in wordlists and tables published in dictionaries and field-work notes, up to a level where correspondence patterns across cognate words have been automatically identified and can be qualitatively inspected by the scholar.

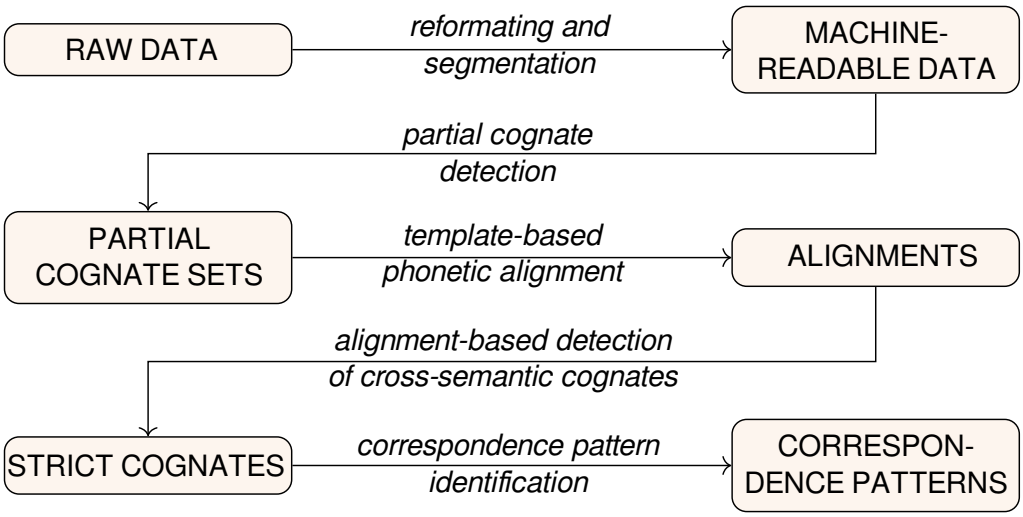

Figure 2: Current state-of-the-art workflow developed in collaboration of different research groups working in computer-assisted frameworks.

Although the workflow can be carried out almost completely without any manual intervention by a linguist, we emphasize that this workflow explicitly *allows* for expert intervention at *any* of the five stages. While, in our experience, specific care is required when lifting the data the first time to machine-readable format, it should further be noted that *all* steps of the workflow profit from human intervention, since none of the automatic methods currently available to us could spot all patterns in linguistic data without over- or underestimating their importance for linguistic reconstruction.

Our workflow starts from *raw data*, including tabular data from fieldwork notes or data published in books and articles, which we re-organize and re-format in such a way that the data can be processed by our tools. Once we have *machine-readable data*, we can use methods for automatic cognate detection (List et al. 2016b) in order to infer *partial cognates* across the languages in our data. Having inferred cognates, we can now also align the data in the cognate sets. While we could use phonetic alignment approaches discussed in the literature (List 2014), we now use a new approach, based on phonotactic templates, which has the advantage of being much faster and accurate when dealing with alignments for South-East-Asian languages. Once having identified the alignments, we start to search automatically for cognates *across* different concepts. Since all automatic methods *need* to start searching for cognates within the same concept slot (otherwise, there would be too many false positives), our new method, which makes used of a systematic comparison of readily aligned cognate sets, systematically searches for cognates independent of their meaning. The improved, cross-semantic cognate sets, which are all readily aligned, have the specific property of being *strict*: no cognate set could compare two morphemes from the same language which would differ in their pronunciation. (List 2018b) calls these cognate sets *regular*, but in discussions with Nathan Hill, we decided that *regular* is probably not the best term, as they can well be wrong, so we call them *strict* now. Once strict cognates have been identified, we use the new algorithm for the automatic inference of sound correspondence patterns across multiple languages by List (2019) to infer the correspondence patterns in the data.

In Section 3, we will provide detailed examples how all steps of the workflow interact, using a rela-

tively recent collection of linguistic data on Hmong-Mien languages (Chén 2012) for this purpose.

2.3 Materials and Methods for the Workflow Illustration

The data we use to illustrate our workflow in the next section was originally collected by Chén (ibid.), and later added in digital form to the Wiktionary project. Chén's collection of *frequent terms* (*chángyòng cíbiǎo* 常用词表, pp. 567-862) comprises 885 different concepts translated into 25 varieties of Hmong-Mien. In Figure 3, we contrast one exemplary page from Chéns book with the data as it has been prepared by the Wiktionary users. We can see that the data is essentially the same, but that the rows and columns of the tabular form have been swapped.

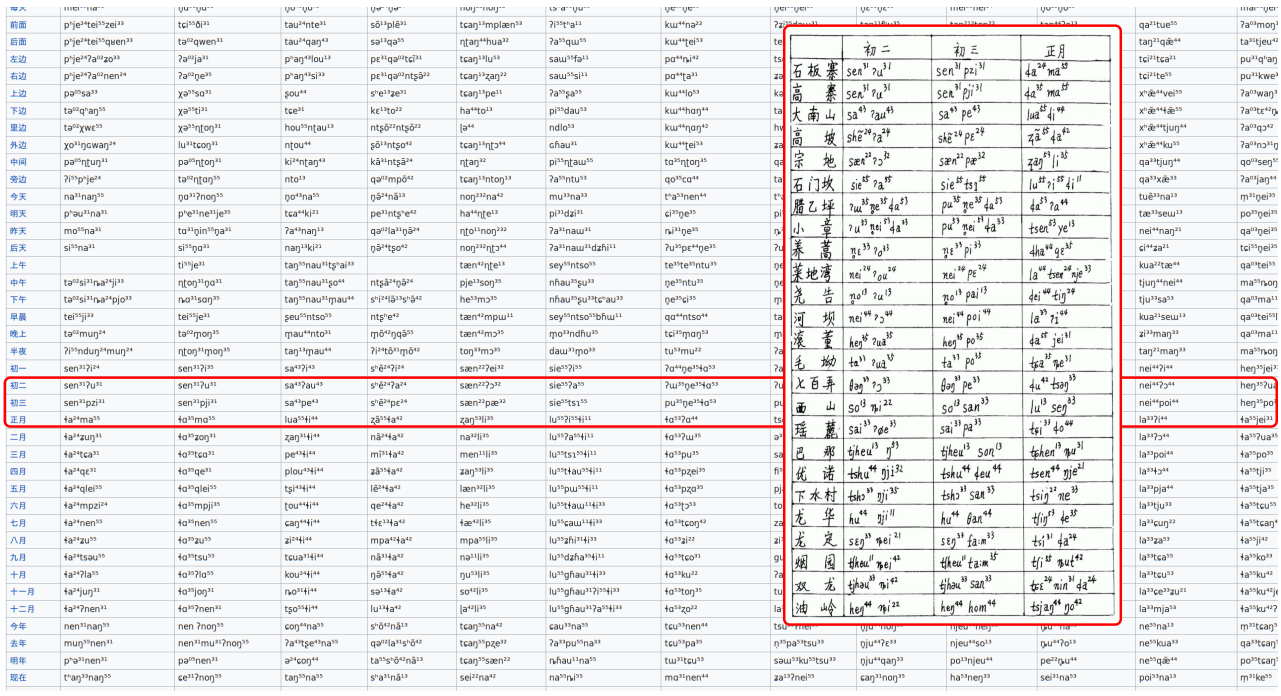

Figure 3: Contrasting Chén's original data with the table in Wiktionary

All methods have either been implemented and published before, or are shared along with the slides and the handout for this talk. Since this is work in progress, however, we warn users that both data and code will be in flux for some time, but we will make sure that both data and code can always be readily analyzed with our tools. All code, the data we use, and installation instructions can be found at `https://github.com/lingpy/calc-workflow`. We ask those interested in testing our methods to use our issue-tracker on GitHub in case they face difficulties of any kind. In this talk, we present the workflow with a subset of 10 varieties of the Hmong-Mien languages in Chén's sample, for which we selected a subset of 313 concepts. The concepts were selected by checking the overlap with the current 504 concept list of the Burmish Etymological Database project (headed by Nathan W. Hill, data online at `https://dighl.github.io/burmish`). The languages were selected for some general reasons, like lexical coverage, geographic distribution, or basic diversity, but not with the specific "eye" of a historical linguist who would select languages to explore the history of a language family. We would be glad about any additional recommendations, if scholars feel competent to give us advice in this context. The geographic locations are shown in the Figure 4.

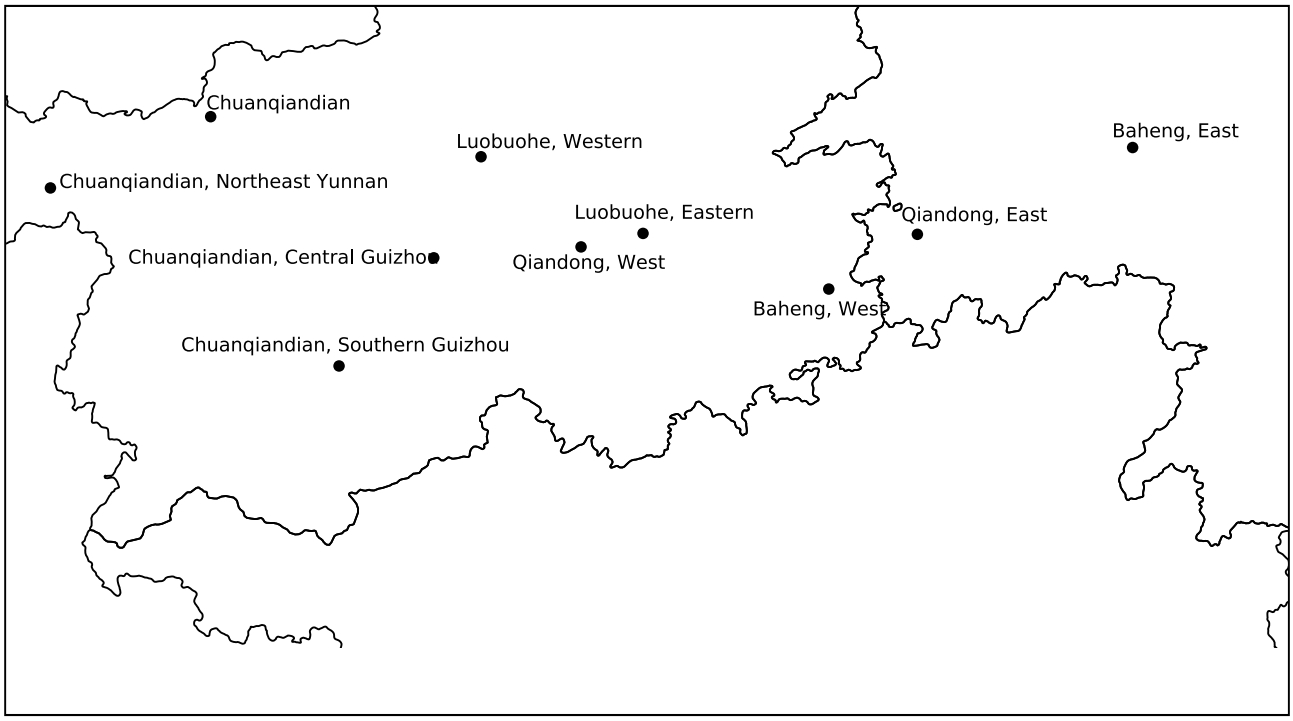

Figure 4: Language geographic locations

3 Illustration of the Workflow

3.1 From Raw Data to Segmented Data

When comparing languages within a computer-assisted framework, with the goal of identifying sound correspondence patterns in the data, we need to make sure that our data is machine-readable at first. If the data is not machine-readable, we can neither use web-based tools like EDICTOR which make it easy to edit the data *manually* (List 2017), nor can we use computational tools, like LingPy (List et al. 2018b), which can help us a great deal in identifying cognate sets and aligning our data.

A first problem for many researchers is to get used to our formats for data representation. In contrast to the typical style used by scholars, we do not use simple tables, with languages in a row and concepts in a column, or vice versa, but instead a so-called long-table format, in which we reserve a *row* in a table for each word, and add a header, which tells us what the cells in each column contain in terms of the data. This long-table format reflects the rule of "One Value per Cell", as stated by the Cross-Linguistic Data Formats initiative (Forkel et al. 2018), reproduced in Figure 5.

As a second rule, we have certain format specifications that make it easier for machines to deal with our input. This includes

- the use of a *segmented* form of IPA transcriptions, in which a space is used to separate distinct sounds from each other, to give the computer direct information on whether symbol combinations are meant to reflect one sound (e.g., affricates, such as [ts, tʃ]), or multiple sounds (compare German *Handschuh* [h a n t ʃ uː] vs. German *Tschüss* [tʃ y s]),
- them use of morpheme segmentation markers (we use a +) to indicate morpheme boundaries, which is straightforward when working with many morpheme-syllabic SEA languages, in which morphemes coincide with syllables,

a ***One Value per Cell***

Many datasets that have been published in the past place multiple values in the same cell of their data. This is most frequently the case with elicitation meanings for which multiple translations could be found. Since scholars are rarely explicit about the separators or the techniques by which they handle these problems, many different ways to address multiple translations per meaning have been used in the past, ranging from additional columns up to secondary characters indicating multiple values in a cell (commas, slashes, pipes), and datasets may even mix the different techniques. To avoid these problems, CLDF specifies to use long tables throughout all applications.

NEITHER:

Meaning	English	German	Dutch
bark	bark	Rinde, Borke	bast

NOR:

Meaning	English	German	Dutch
bark	bark	Rinde	bast
bark	*	Borke	---

BUT:

ID	Meaning	Language	Form
1	*bark*	English	bark
2	*bark*	German	Rinde
3	*bark*	German	Borke
4	*bark*	Dutch	bast

Figure 5: Long-table format instead of condensed formats with multiple values per cell.

- a clear-cut account on the concepts in our data, as they serve as the initial comparanda, so each concept needs to be given a clear-cut definition, and our preferable starting points are concept lists which are translated into the languages to be investigated, as opposed to pre-selected accounts on potential etymological items.

We indicate words in the computer-readable form, by adding a column called `TOKENS` in which data is segmented with a space to distinguish different sounds, and with the plus-symbol to distinguish different morphemes.

Thus, our original data consists of a text-file, separated by tabstop, with the first row serving as a header, and the following rows providing information for one word per language. Our software requires the following columns to be submitted:

- `ID`: numerical identifier, greater than 0,
- `DOCULECT`: name of the language,
- `CONCEPT`: some gloss for the concept,
- `TOKENS`: the morpheme and sound-segmented form of the data.

We recommend also to add a column called `VALUE`, containing the original data, as well as a column `FORM`, which shows the original data but corrected for multiple values per cell. The software usually automatically creates a form `IPA`, which is not necessarily used, but a legacy form that will be replaced by the `FORM` in future updates. Additional values are then consistently added by our workflow and will be discussed later.

Note in general, the data can be prepared with typical spreadsheet programs, such as Excel or LibreOffice or GoogleDocs. In order to create the textfiles, we recommend to simply copy-paste the data from a spreadsheet by opening an empty text file, copying the data, and pasting it into the file. In this way, the tab-separated format required by our applications will always be preserved.

We offer procedures to ease the conversion of the data to the required formats. While the creation of long-table formats is usually done by applying a custom script, we use *orthography profiles* to create morpheme-segmented IPA representations for our `TOKENS` column from the original data (Moran and Cysouw 2018). Orthography profiles are a very straightforward way to convert raw data to space-separated IPA representations. An orthography profile can be thought of as a simple text file with two or more columns in which the first represents the values as you find them in your data (i.e., non-IPA transcriptions, etc.), and the other columns allowing you to convert the sequence of characters that you find in the first column. So in brief, you have a source-pattern and a replacement pattern, for example, the one shown in Table 1. With such a replacement pattern, an input string `čashaa` would on the one

hand be segmented into `č a sh aa` and at the same time, it would be converted to `tʃ a ʃ a:`. We now offer an online demo of orthography profiles at `http://calc.digling.org/profile`, which can be used to test and apply customized orthography profiles.

Grapheme	**IPA**
č	tʃ
ž	dʒ
th	t^h
dh	d̪
sh	ʃ
a	a
aa	aː

Table 1: Very simple orthography profile example.

SUMMARY

- Data must be machine-readable in order to be amenable for computer-assisted analyses.
- Data must specifically be segmented, both with respect to the morpheme boundaries and the boundaries between distinct sounds.
- Data must be provided in form of a *long table* with some specific column headers, providing all relevant information.
- Computer-assisted tools help to prepare the data for computer-assisted processing.

3.2 From Segmented Data to Cognate Sets

Once the data is segmented and provided in the long table format as it is required by the LingPy software package, as described in our tutorial (List et al. 2018a), we can use LingPy's partial cognate detection method to infer partial cognates in our linguistic data. Partial cognates are hereby understood as cognate assessments *per morpheme* in our data, as opposed to cognate assessments *per word*. While it has always been clear to scholars working in the field of South-East Asian linguistics that cognacy should rather be assigned on the level of the morpheme than on the level of full words, given that the high degree of compounding would easily complicate the identification of cognate relations, automatic methods, and specifically phylogenetic reconstruction approaches usually still assume a rather naive one-word-one-cognate relation (List 2016).

In our framework, we explicitly address this problem by adopting a numerical annotation format in which each morpheme instead of each word form is assigned to a specific cognate set (Hill and List 2017). This framework is illustrated in Figure 6, where we contrast word forms for "yesterday" in five Burmish varieties, indicating their detailed "cognate relations". In the first "traditional" style of cognate coding, we would proceed in a *strict* way, only allowing those words which are completely cognate in all their morphemes to be judged as cognates. In the second, *loose* cognate annotation, we judge all words that are in a *connected component* in our shared morpheme network to be cognate, and in the last column, we show our explicit coding of partial cognacy, in which each morpheme is assigned to one cognate set.

Language	Form	Strict	Loose	Exact
Bola	a^{31} $\eta\!\!\!\!\downarrow i^{35}$ $n\varepsilon ʔ^{31}$	1	1	1 2 3
Lashi	a^{31} $\eta\!\!\!\!\downarrow ei^{55}$ nap^{31}	1	1	1 2 3
Rangoon	ma^{53} ne^{53} ka^{53}	2	1	0 3 0
Xiandao	$n̩^{31}$ $m̩an^{35}$	3	1	3 4
Achang	man^{35}	4	1	4

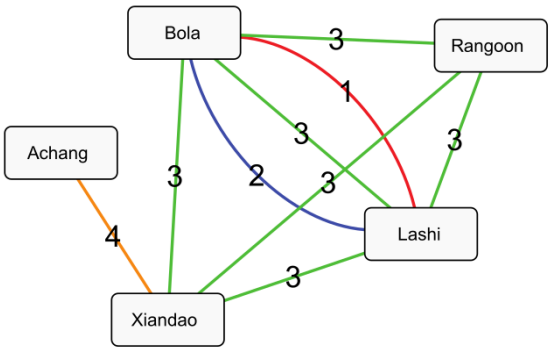

Figure 6: Partial cognacy in Burmish language varieties and different ways of coding (see Hill and List 2017 and further explanations in the main text). coding.

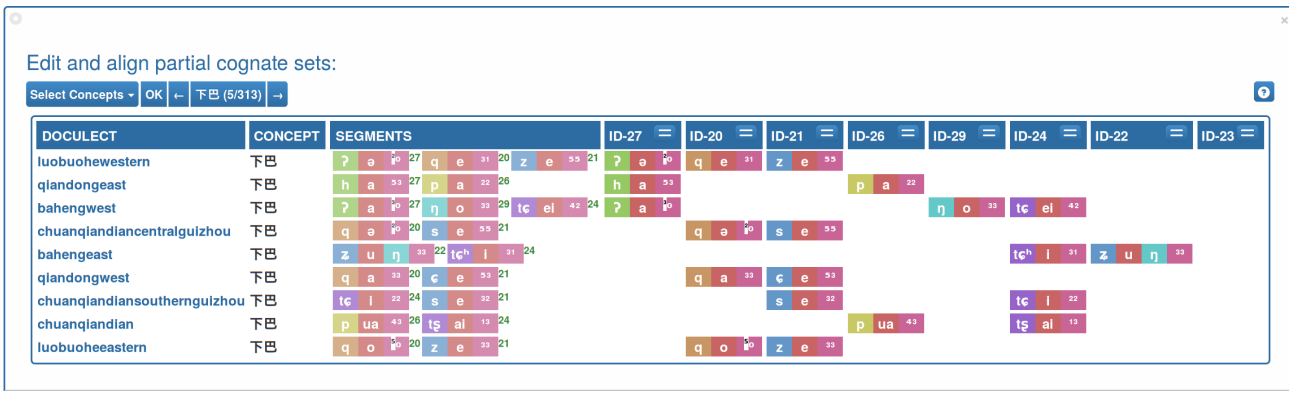

Figure 7: Partial cognate annotation within the EDICTOR tool for the word for “chin” in 10 selected Hmong-Mien varieties.

The software package LingPy offers a straightforward algorithm to detect and annotate partial cognates in datasets formatted as long tables. This algorithm by List et al. (2016b) uses techniques for automatic sequence comparison to create a network of similar morphemes for each meaning slot in a given dataset. It then filters those concepts in consecutive stages, with the goal of avoiding that two or more morphemes in the same word for the same language are assigned to the same cluster. In the end, the algorithm outputs the cognate judgments in the same format as indicated above in Figure 6, namely, but assigning each morpheme to a given number, with the number representing that cognate set.

Note that this algorithm works quite well, although it is, of course, not infallible. It reaches between 88 and 90 percent on a test datasets consisting of Bai dialects, Chinese dialects, and dialects of Tujia. With more challenging datasets, the scores will surely drop, but we can expect that the automatic cognate detection is in any case *helpful*, as is easier to correct cognates than to assign them from scratch.

In addition to the cognate detection algorithm, the EDICTOR web-based tool for computer-assisted language comparison (List 2017), freely available at `http://edictor.digling.org`, can be used to quickly inspect and correct computer-generated cognate sets, by providing a very convenient interface that allows users to quickly assign morphemes to cognate sets. The interface is illustrated in Figure 7.

SUMMARY

- For a realistic annotation of cognate sets, the annotation of partial cognates, by which morphemes are assigned to cognate sets, is the only realistic choice.
- Partial cognates can be automatically identified with help of software, openly available as part of the LingPy software library (`lingpy.org`, List et al. 2018b) and the algorithm by List et al. (2016b).
- Partial cognates can be annotated consistently with help of the EDICTOR tool (List 2017), online available at `http://edictor.digling.org`.
- Partial cognates in these frameworks are assigned to morphemes occurring in words with the same meaning, both for algorithmic and for practical reasons.

3.3 From Cognate Sets to Alignments

Algorithms for phonetic alignments in historical linguistics have been proposed since the 1990s (Covington 1996, Covington 1998). The basic of an alignment is to arrange sequences in such a way in a matrix that corresponding segments are placed in the same column (List et al. 2018c). For the transparent annotation of sound correspondences, alignments are a *sine qua non*, there is no way around them, even if scholars at times think otherwise. Since sound correspondences can only be annotated and detected when comparing sound sequences (words, morphemes) in full, we need alignments to identify them, specifically when working with more than just two languages.

During the beginning of the second millenium, the methods for phonetic alignments have drastically improved. Starting with the work by Kondrak (2000) on pairwise alignments, we have now stable algorithms for multiple alignments that yield accuracy scores almost comparable to the differences we would expect between human annotators (List 2014). With EDICTOR (List 2017), we also have a tool that facilitates to align words across a larger number of languages, and the LingPy software package (List et al. 2018b) offers a very stable implementation of the Sound-Class-Based phonetic Alignment algorithm (List 2012b), which can be considered the current state of the art, as far as multiple phonetic alignments are concerned.

Unfortunately, phonetic alignment algorithms are not perfect, and correcting alignments manually is tedious, specifically when working in computer-assisted workflows, where one runs a computational analysis and then has experts correct the results. If one changes one cognate set assignment, one has to re-do the whole alignment analysis, and if the algorithm constantly gets something wrong, this means that the researcher will need to correct the alignment ever and ever again, even when only small changes to the data are undertaken.

For this reason, we started to develop a new method for multiple phonetic alignments, specifically targeted to SEA languages with restricted syllable structure, which allows us to align words without actually aligning them. This method, which we call *template-based alignments*, starts from the simple observation that many SEA languages don't differ much in their syllable structure, allowing us to capture which sound occurs in which position, and which sound should be compared to which other sound, by simply adding another column to our wordlist file, which contains a layer for the phonotactic structure of each syllable. These *templates* are stored in a column which we call `STRUCTURE`, for convenience, and they are arbitrary in so far as we allow users to represent their template by any symbol sequence, as long as they respect our two-fold segmentation for segmentized IPA-strings, which uses space for the segmentation of sounds, and the plus sign to segment morphemes.

For reasons of simplicity, we started from the well-known structure of Sinitic languages, which – fol-

lowing Wang (1996) – assumes syllable templates consisting of an *initial* (`i`), a *medial* (`m`), a *nucleus* (`n`), a *coda* (`c`), and the *tone* (`t`). In this schema, Chinese *tàiyáng* [tʰ ai ^{51} + j a ŋ ^{35}] would be represented as `i n t + i n c t`. Assuming that the syllable template of the ancestral language we want to investigate did not differ much from this, we can now use the templates to align words automatically, by simply starting from our general template, to which we align all words, and then deleting those columns for the syllable positions which do not occur in the words under comparison. This is illustrate in Figure 8, where four words for "seven" in four Hmong-Mien languages are successively aligned with each other.

Doculect	Concept	Tokens	Structure
East Baheng	seven	tɕ a ^{31}	i n t
West Baheng	seven	tɕ a ŋ ^{44}	i n c t
Chuanqiandian	seven	ɕ a ŋ ^{44}	i n c t
Chuanqiandian (CG)	seven	s ã ^{22}	i n t

→

i	m	n	c	t
tɕ	-	a	-	^{31}
tɕ	-	a	ŋ	^{44}
ɕ	-	a	ŋ	^{44}
s	-	ã	-	^{22}

→

Alignment			
tɕ	a	-	^{31}
tɕ	a	ŋ	^{44}
ɕ	a	ŋ	^{44}
s	ã	-	^{22}

Figure 8: Basic procedure for template-based alignment.

The problem of this procedure is that it requires more input by the user, since templates should ideally be manually assigned and checked for each word in the data. However, by now, we offer different automatic and semi-automatic approaches the further help to create syllable templates automatically. As a first possibility, extended orthography profiles can be used. In these profiles, the data is analyzed in much more detail, offering larger chunks in the first column, which can then be converted to a template at the same time when converting unsegmented strings to segmented strings. Since our procedure also offers to capture the beginning and the end of a sequence, this allows for a rather straightforward handling that is usually sufficient for datasets of moderate size. We illustrate this procedure in Table 2. An alternative possibility is to use the SinoPy library, a Python package for quantitative tasks in Chinese historical linguistics (List 2018c) to create templates from input strings automatically.

Grapheme	Segmented	IPA	Structure
ˆtʰ	tʰ	tʰ	i
ˆv	v	v	i
ɛ	ɛ	ɛ	n
au	au	au	n
iei	i ei	j ei	m n
oŋ	o ŋ	o ŋ	n c
uɑŋ	u ɑ ŋ	w ɑ ŋ	m n c
loŋ	l o ŋ	l o ŋ	m n c
^{31}$	^{31}	^{31}	t

Table 2: An example of an orthography profile that can creates templates along with the conversion to segmented IPA. The second column represents the data as we find it in the source, in segmented form, the third column contains a certain amount of corrections for IPA handling, and the fourth column offers the data in form of the phonotactic structure.

Template-based alignments have not been extensively tested so far, although it is clear for our purpose, that they work at a level of 100%, given that the user virtually provides the alignment without actually aligning sequences. We can think of additional experiments, in which our approach to template-based alignments could be tested, also when dealing with languages with more complex syllable structures, where longer, and more complex templates could be used along with our algorithm. Since morphemes – in contrast to words – tend to be small, the major message of our template-based alignment approach is that we do not need to invest too much time in sophisticated algorithms that try to guess in whatever way how to arrange sound sequences in a matrix, if we can – at least for certain

language families – already determine how to align the strings by simply looking at their phonotactics. Since template-based alignments are essentially linear with respect to their computational complexity, template-based alignments may also provide further help in all those tasks in computational historical linguistics, where alignments are needed, but slow down the algorithms, as for example, when searching for regular sound correspondences with help of randomizing the data (List 2012a).

SUMMARY

- Alignments are indispensable for the detection of sound correspondence patterns.
- Given that existing algorithms are complex, not error-free, and difficult to apply, we use a new method, that essentially uses phonotactic templates, along with a meta-template to align morphemes in our data..
- The advantage of the method is that it is almost error-free (as long as the templates are correct), and extremely fast to apply.
- The disadvantage of the method is that it requires more user-input, since the templates need to be checked manually.
- In order to create templates from segmented data, two methods are available, (a) extended orthography profiles, and (b) a Python function provided along with the SinoPy Python library.

3.4 From Alignments to Cross-Semantic Cognates

As mentioned above in Section 3.2, the partial cognates are only identified for words with the same meaning. This is being done for algorithmic reasons (it would become quite complex to compare all morphemes against each other algorithmically), and for practical reasons, since we believe that it is always better to start from the obvious and save etymologies in historical linguistics, rather than to start from complex ones. Given that semantic shift is a phenomenon for which we dispose of little knowledge with respect to its patterns, we agree explicitly with scholars like Dybo and Starostin (2008) in emphasizing that we should always expect to find clear-cut etymologies within words of the same meaning, even if we know that more etymologies could be find when searching *cross-semantically*, i.e., among words which differ with respect to their meanings.

There are only a few approaches that try to identify cognates across different concepts, and one could say that the task of *cross-semantic cognate detection* is still one of the open problems in computational historical linguistics. Approaches proposed so far include a rather complex workflow by Wahle (2016), who uses *hidden Markov models* for sequence comparison, and proxies on colexifications, drawn from the database by Dellert and Jäger (2017), to infer cognates across different meaning slots. As this task is not completely evaluated, and only described in a short paper, it is difficult to access its usefulness for our purposes. Another approach is presented by Arnaud et al. (2017), who apply Support Vector Machines trained on form and semantic similarities of word pairs along with a flat clustering algorithm to partition words into cognate sets. While this approach is publicly available and seems to yield promising results, we are not sure to which degree it would help us with our very specific goals of lifting an initially “raw” dataset to a level where we can assess sound correspondence patterns across multiple languages, especially since the algorithms the authors use for cognate detection do *not* take regular sound correspondences into account, and they are also *not* sensitive to partial cognates.

Thus, instead of these previously proposed solutions, we propose our own, rather simple approach to search for cross-semantic partial cognate sets in our data. This approach is based on the well-observed fact that the majority of morphemes in South-East Asian languages with a certain preference for compounding and a high degree of word formation, is highly *promiscuous* (List et al. 2016a: 8f), given that they recur within different words, surfacing in the form of *partial colexifications* (Hill and List 2017: 62). The term *partial colexification* hereby serves as a cover term for morphemes recurring across the lexicon of a language, with no specific distinction being made if they are polysemous or homophonous.

Our search for partial colexifications would not allow us directly to identify cross-semantic cognates consistently, given that sound change may yield different morpheme mergers across different languages. As a result, we cannot take the information from one language alone, but have to smartly summarize all the information on recurring morphemes we can find in our data. The solution for this problem is nevertheless straightforward, and it builds on the idea to not only compare single words, as originally proposed in Hill and List (ibid.), but to compare complete *alignments* instead. As our data is already aligned, and we have identified cognates in a first run, potentially even refined by experts, we can compare whole cognate sets that contain *identical words in the same language*.

If two alignments are completely identical with respect to the words they contain, there is no reason to assign them to different cognate sets, and we can directly assign them to the same cognate class. Even if they are simply homophonous, the assumption of regular sound change will allow us to treat them similarly if we reconstruct the words back to the ancestral language.

The problematic cases are those cases, where we have *incomplete data*. And this is usually rather the rule than the exception. We often will encounter cases where we have two alignments which are only filled in parts with data from the different languages, and we will usually have *missing data* for one or more of the languages in our sample in a given alignment. Thus, when comparing two alignments with each other, we need to make sure that we have at least one word in one language in common.

As an example, consider the data on “son” and “daughter” in five language varieties of our illustration data. As can be seen immediately, two languages show striking *partial colexifications* for the two concepts, Chuanqiandian and East Qiandong. In both cases, one morpheme recurs in the words for the two concepts. In the other cases, we find different words, but if we compare the overall cognacy, we can also see that all five languages share one cognate morpheme for “son” (corresponding to the Proto-Hmong-Mien *tu̯ɛn in Ratliff’s reconstruction), and three varieties share one cognate morpheme for “daughter” (corresponding to *mphje^{D} in Ratliff’s 2010 reconstruction), with the morpheme for “son” occurring also in the words for “daughter” in East Qiandong and Chuanqiandian, as mentioned before.

Language	Concept	Form	Cognacy	Cross-Semantic
East Baheng	SON	taŋ^{35}	1	1
East Baheng	DAUGHTER	p^{h}je^{53}	2	2
West Baheng	SON	ʔa$^{3/0}$ + taŋ^{35}	3 1	3 1
West Baheng	DAUGHTER	ta^{55} + qa$^{3/0}$ + t^{h}jei^{53}	4 5 6	4 5 6
Chuanqiandian	SON	to^{43}	1	1
Chuanqiandian	DAUGHTER	^{n}ts^{h}ai^{33}	7	7
Chuanqiandian (Central Guizhou)	SON	tə$^{2/0}$ + tə̃^{24}	8 1	8 1
Chuanqiandian (Central Guizhou)	DAUGHTER	tə̃^{24} + ^{n}p^{h}e^{42}	9 2	1 2
East Qiandong	SON	tei^{24}	1	1
East Qiandong	DAUGHTER	tei^{24} + p^{h}a^{35}	9 2	1 2

Table 3: Terms for “son” and “daughter” across five Hmong-Mien varieties.

Our workflow for automatically identifying these cases of cognacy is a new algorithm for cross-semantic cognate detection, developed first for the work in the Burmish Etymological Dictionary project

lead by Nathan W. Hill. In this workflow, we start from all aligned cognate sets in our data, and then systematically compare all alignments with each other. Whenever two alignments are *compatible*, i.e., they have (1) at least one morpheme in one language occurring in both aligned cognate sets, which is (2) identical, and (3) no shared morphemes in two alignments which are *not* identical, we treat them as belonging to one and the same cognate set. We iterate over all alignments in the data algorithmically, merging the alignments into larger sets in a greedy fashion, and re-assign cognate sets in the data.

The results can be easily inspected with help of the EDICTOR tool, for example, by inspecting cognate set distributions in the data. When inspecting the cross-semantic cognates, which we label `CROSSIDS` in our data, the tool will always show, which cognate sets span more than one concept, and users can directly filter the data and look at the relevant instances. Among the 64 cognate sets reflected in all languages in our sample, we find quite a few cross-semantically recurring morphemes, seven in total (with many more for the whole data). The results are shown in Table 4.

Language	Concept	Form	Morphemes
East Baheng	NOSE	^{n}pjau^{31}	`NOSE`
East Baheng	NASAL MUCUS	qa$^{3/0}$ + ^{n}pjau^{31}	`qa NOSE`
West Luobuohe	TWO	ʔu^{31}	`TWO`
West Luobuohe	TWENTY	ʔu^{31} + ʐo^{31}	`TWO zo`
West Baheng	SON	ʔa$^{3/0}$ + taŋ^{35}	`SON`
West Baheng	SON-IN-LAW	taŋ^{35} + wei^{31}	`SON wei`
West Baheng	GRANDSON	taŋ^{35} + seŋ^{31}	`SON seng`
East Qiandong	SUN	q^{h}aŋ^{33} + nei^{24}	`po SUN`
East Qiandong	DAY (NOT NIGHT)	nei^{24}	`SUN`
West Baheng	FAECES (EXCREMENT)	qa^{31}	`SHIT`
West Baheng	STOMACH	ʔa$^{3/0}$ + tɕ^{h}i^{35} + qa^{31}	`a tci SHIT`
West Qiandong	ANT	kæ̃^{44} + mjɔ^{22}	`INSECT mjo`
West Qiandong	EARTHWORM	kæ̃^{44}+tɕuŋ^{44}	`INSECT tsung`
East Baheng	BIRD	taŋ^{35} + nuŋ^{31}	`BIRD-A BIRD-B`
East Baheng	NEST	ʐo^{11} + taŋ^{35} + nuŋ^{31}	`zo BIRD-A BIRD-B`

Table 4: Partial cognates among stable concepts with reflexes in all languages in our test datasets. We highlight shared cognates by giving a tentative gloss for them in capital letters in the column *Morphemes*.

SUMMARY

- For a realistic analysis, we need to identify cognates not only within the same meaning slot, but across different concepts, specifically when dealing with languages in which compounding and word formation are very productive.
- We employ a new method that makes use of a comparison of the alignments in readily identified and aligned partial cognate sets to identify those morphemes which recur across different concepts in our data.
- The results can be inspected with help of the EDICTOR, but not directly, by now, only indirectly with help of the browser for cognate sets.
- The interpretation of the results cannot be done automatically, but requires expert assessment with respect to the morphology of the data under consideration.

3.5 From cross-semantic cognates to correspondence patterns

Having identified our cross-semantic cognates, or – to be more specific – our *strict* cross-semantic cognates, we can now finally run the analysis required to detect the sound correspondence patterns in our data. Note that by sound correspondence *patterns* we mean essentially sound correspondences for more than just two languages. In most formal or automatic approaches to historical linguistics, starting from Hoenigswald (1960), via (Kondrak 2003), up to the LexStat algorithm, which uses previously identified sound correspondences to identify cognates across multiple languages (List 2012a), sound correspondences are exclusively handled for language *pairs*, also when working with more than two languages. This means, that instead of identifying correspondences across all languages in the data, the methods or descriptions assume that it is enough to look at the correspondences for each of the possible pairs in the data. While it is clear that computationally, this may be the only feasible solution, especially when lacking further information or data about the languages, the pairwise perspective does not help us to further proceed with respect to linguistic reconstruction. If we wish to reconstruct the proto-language, we *need* to switch to a multi-lingual perspective, especially with respect to sound correspondences. For this reason, we adopt the term *correspondence pattern* for those cases where we have more than just two languages in our data, and *correspondence pattern detection* is the task by which we try to identify all patterns of regular sound correspondences in our data.

In our specific, alignment-based interpretation of sound correspondences, sound correspondence patterns are understood as *recurring alignment sites* in a given dataset. An *alignment site* is a column in a given alignment, following the terminology that is usually employed by biologists. While scholars occasionally share correspondence patterns in the literature, the way they share them in almost never exhaustive, but always eclectic, being based on a pre-selection of the data in such a way that they support a given theory or hypothesis. A prototypical example for this representation is given in Figure 9, with reflexes of Proto-Indo-European labial and dental stops (Beekes 1996: 127). That does not essentially mean that proposals on correspondence patterns in the literature are wrong per se, because scholars often have good arguments to exclude specific patterns, which only occur due to specific sound changes which are conditioned by peculiar contexts in some languages. It means, however, that scholars do not always share all of the data, and as long as scholars do not share all of their data, we do not have the possibility to assess whether they really know about all possible variations that their data offers.

PIE	Skt.	Av.	OCS	Lith.	Arm.	Toch.	Hitt.	Gr.	Lat.	OIr.	Goth.
*p	p	p	p	p	h-, w[1]	p	p	p	p	ø	f, b[2]
*b	b	b	b	b	p	p	p	b	b	b	p
*b^h	bh	b	b	b	b-, w[3]	p	p	ph	f-, b[4]	b	b
*t	t	t	t	t	t'	t, c[5]	t[6]	t	t	t	þ, d[2]
*d	d	d	d	d	t	ts, ś[5]	t	d	d	d	t
*d^h	dh	d	d	d	d	t, c[5]	t	th	f-, d, b[7]	d	d

Figure 9: Example for the typical representation of correspondence patterns in the classical literature, taken from Beekes (1996: 127).

An example for a very transparent form of presenting a reconstruction along with correspondence patterns is the Proto-Hmong-Mien reconstruction by Ratliff (2010). Ratliff herself talks about *correspon-*

dence sets, listing a proto-sound following her reconstruction and all *reflexes* in the descendant languages, based on concrete cognate sets. An example for this format, reflexes reflecting the initial *mbl- in Proto-Hmong-Mien, is provided in Figure 10. While this format is much more consistent than the one provided in Figure 9 above, it still has a couple of disadvantages: (1) it does not provide the full words in the examples, but only the morphemes (something we track in our computer-assisted workflow, by handling partial cognates consistently), (2) it does not provide us with information on the degree to which patterns are inconsistent, reflecting secondary variation in the individual languages (we can only find that out by carefully inspecting the data ourselves), and (3) the representation is only to some degree machine-readable, not only because it is not digitized, but also because we do not know completely to which correspondence pattern (or set) each of the other sounds in the data belongs, and we will have problems to check the consistency of entire words when given the data in this form.

2.8 *hn-

PHM *hn-	1	2	3	4	5	6	7	8	9	10	11
1. grain head/bag *hnɔn	n̥haŋ^{1}	n̥hei^{1}	hn̥a^{1}	na^{1b}	n̥en^{A}	n̥en$^{1'}$	n̥ẽ^{1}	-	nɔ^{1}	nan^{1}	-
2. to hear *hnəumX	n̥haŋ^{3}	n̥hɑŋ^{3}	hn̥ɔ^{3}	nɔ^{3b}	n̥u^{B}	n̥ɐŋ^{3}	n̥õ^{3}	nom^{3}	num^{3}	n̥ən^{3}	-
3. to put on/wear (clothes) *(h)naŋX	naŋ^{4}	n̥hei^{3}	hn̥a^{3}	na^{3b}	-	nen^{4}	nẽ^{3}	-	-	-	nɔŋ^{3}
4. to cough *hnɔp	ŋo^{4}	-	hn̥ɔŋ^{7}	naŋ^{7}	n̥o^{D}	-	-	n̥op^{7}	-	n̥ən^{7}	-
PH *hn-	1	2	3	4	5	6	7				
5. sun/day *hnɛŋ^{A}	n̥hɛ^{1}	n̥he^{1}	hn̥u^{1}	noŋ^{1b}	n̥a^{A}	n̥ɔ$^{1'}$	n̥ẽ^{1}				
6. crossbow *hnæn^{B}	n̥hen^{3}	-	hn̥ɛŋ^{3}	nein^{3b}	n̥a^{B}	-	n̥ẽ^{3}				
7. to forget *hnuŋ^{B}	n̥hoŋ^{1}	noŋ^{3}	hn̥ɔ^{3}	na^{3a}	n̥oŋ^{A}	n̥ɐŋ^{3}	nõ^{3}				
8. perilla (*sū má*) *hnaŋ^{B}	n̥haŋ^{3}	n̥hen^{3}	hn̥a^{3}	-	n̥en^{B}	-	-				
PM *hn-								8	9	10	11
9. sun/day *hnu̯ɔi^{A}								n̥ɔi^{1}	nɔ:i$^{1'}$	n̥wai^{1}	nai^{1}
10. to resemble *hnəŋ^{B}								n̥aŋ^{2}	naŋ$^{3'}$	-	-
11. to lift *hniŋ^{C}								niŋ^{5}	niŋ$^{5'}$	-	-
12. crossbow *hnək^{D}								n̥ak^{5}	na$^{7'}$	-	-
13. 泥 mud *hni^{A}								ni^{1}	ni$^{1'}$	n̥i^{1}	nɛi^{1}
14. to think of *hnəm^{B}~*hləm^{B}								-	lam^{3}	-	-

Figure 10: Example from Ratliff (2010: 57), correspondence pattern for Proto-Hmong-Mien *hn-.

The result of an correspondence pattern analysis as we imagine and propose it can be thought of as some kind of a table, in which languages are placed in the columns, and correspondence patterns are placed in rows, with each cell indicating for each individual correspondence pattern, which reflex sound a given language shows for this pattern. If we further assume that our interface is *interactive*, the correspondence pattern analysis should further deliver all information in a way that can be directly traced back to the original data, from the original morphemes, the position in the syllable, up to the potential irregularity of the pattern, and its overall frequency in the data under investigation. This format is essentially offered by the EDICTOR tool by now, when using the correspondence pattern inspection panel of the tool, as illustrated in Figure 11. Here, the user finds the correspondence patterns in the data sorted according to the number of cognate sets in which they are reflected, along with an index indicating where in the alignment the pattern occurs, with which other patterns it is compatible, and the concepts in which the correspondence pattern is reflected. While this would not offer much advantages over any display in books, the tool is interactive, and clicking on the patterns allows to see the full word,

COGNATES	INDEX	PATTERN	CONCEPTS	luo	luo	chu	chu	qia	qia	bah	bah	chu	chu	SIZE
8	1	n̥ / 28	太阳,天	n	n̥	n̥	n̥	n	n	n̥	n̥	n̥	n	3.50 / 6
207	1	n̥ / 28	听见	n	n̥	n̥	n̥	n	n	n̥	Ø	n̥	n	3.50 / 6
300	1	n̥ / 28	咳嗽	n	Ø	Ø	Ø	Ø	n	Ø	Ø	n̥	n	3.50 / 6
847	1	n̥ / 28	麻雀	n	Ø	Ø	Ø	n	n	Ø	Ø	Ø	Ø	3.50 / 6
105	1	n̥ / 28	箭	n	n̥	Ø	Ø	Ø	Ø	n̥	Ø	Ø	Ø	3.50 / 6
93	1	n̥ / 28	弓	Ø	Ø	n̥	n̥	Ø	n	n̥	Ø	n̥	n	3.50 / 6

Figure 11: Interactive display of correspondence patterns within the EDICTOR tool.

the underlying alignment, or to quickly browse back to the original data. All in all, we can lift our data easily to a level similar to the one provided in Ratliff's reconstruction, but without loosing track of all steps that were done before.

Correspondence patterns can be computed in two ways, either by using the correspondence pattern detection algorithm by List (2019), implemented in the LingRex package (List 2018a), which will probably later be included as part of LingPy. An alternative is to use the EDICTOR directly, although this algorithm is based on a simple sorting of alignment sites, and will have difficulties when being confronted with particularly patchy data, with many missing reflexes for a given cognate set.

When applying the algorithm by List (2019) to our data of 10 languages, we find as many as 786 patterns in total for all five different phonotactic positions that we were assuming for this analysis. This may seem to be a lot, but it should be kept in mind that – apart from errors in our identification of cognate sets and alignments – it is quite common to find a lot of exceptional correspondence patterns, and even in Ratliff's data, we can see that essentially all four patterns offered for Proto-Hmong-Mien *hn- are different from each other in at least one reflex. We still expect that a certain amount, especially of the 323 patterns in our analysis which occur only once in the data, may result from the fact that we did only use computational analyses for the data analysis, without any human interaction. In general, however, we are confident, that our approach could identify a quite substantial part of the patterns in the data, which is also confirmed by the comparison with Ratliff's work, which we qualitatively compared against our findings.

No.	Concept	Proto-Form	1	2	3	4	5	6	7	8	9	10	11
	Pattern		n̥h	n̥h	hn̥	n	n̥	n̥	n̥	?	n	n̥	n
1	grain head/bag	*hnɔn	n̥h	n̥h	hn̥	n	n̥	n̥	n̥	Ø	n	n	Ø
2	to hear	*hnəumX	n̥h	n̥h	hn̥	n	Ø	n̥	n̥	n	n	n̥	Ø
3	to put on clothes	*(h)naŋX	n	n̥h	hn̥	n	n̥	n	n	Ø	Ø	Ø	n
4	to cough	*hn ɔp	ŋ	Ø	hn̥	n	n̥	Ø	Ø	n̥	Ø	n̥	Ø

Table 5: Irregularities in the patterns for Proto-Hmong-Mien *hn- in the reconstruction of Ratliff (2010).

As an example why we are confident that our workflows yield in fact very useful results, even when applied in a purely automatic fashion, consider the pattern #189 in our current account of the data, as displayed in Figure 12. The reflexes in this pattern correspond to Ratliff's Proto-Hmong-Mien *pr- initial, and our workflow allowed us to find four examples, including words for "five" and "house", which are also the primary witnesses in Ratliff's reconstruction. Our algorithm for cognate detection essen-

tially succeeded in identifying that the words under question are cognate, despite their highly divergent reflexes, including p-, tʂ-, ts-, and t-. The algorithm failed to identify East Qiandong (qia) ts a ^{24} as cognate with the other words for "five", but this can be easily corrected, specifically when inspecting the correspondence pattern, as we would beyond doubt assume that the East Qiandong initial is ts-. The final two forms, "fruit" and "bear fruit" probably reflect cognate forms, but they could not be captured by our cross-semantic cognate detection algorithm, since the words differ in tone, and can therefore not be considered as being *strictly* cognate. Ratliff assigns the words to another correspondence pattern *pj-, which is justified when assuming that this would yield traces in the reflexes of our data. Indeed, when inspecting the data further, we can see that the algorithm missed to assign Chuanqiandian ts i ^{55} "fruit" to the other cognate sets, but if the algorithm had done so, the "fruit" correspondence pattern would no longer be compatible with the "five" correspondence pattern and therefore no longer be listed here.

COGNATES	INDEX	PATTERN	CONCEPTS	luo	luo	chu	chu	qia	qia	bah	bah	chu	chu	SIZE
419	1	p / 189	五	p	p	tʂ	p	Ø	p	t	p	p	p	2.90 / 4
1412	1	p / 189	房子	p	p	tʂ	p	ts	p	Ø	p	Ø	p	2.90 / 4
1031	1	p / 189	果子	Ø	Ø	Ø	p	Ø	p	t	p	Ø	p	2.90 / 4
246	1	p / 189	结(果)	p	p	Ø	p	Ø	p	t	p	Ø	p	2.90 / 4

Figure 12: Correspondence patterns for "five", "house", and "fruit", as automatically identified from our data for testing.

SUMMARY

- Correspondence patterns are similar to sound correspondences, but they are explicitly thought to be identified across multiple languages, by clustering the alignment sites in a given dataset.
- Correspondence patterns are indispensable for linguistic reconstruction.
- Apart from a tedious and not fully traceable manual identification of sound correspondence patterns, two computer-assisted methods for their identification from alignments are available: a (rather poor) built-in application provided by the EDICTOR tool, and a sophisticated variant, based on a network-algorithm by List (2019), provided as a Python package.
- The EDICTOR offers for a convenient inspection of inferred correspondence patterns, and allows scholars to directly see the consequences of their data analysis.

4 Discussion

Our workflows are not complete yet, and they are also not perfect. Many more aspects need to be integrated, discussed, and formalized. In the following, we will point to two important aspects, namely, (a) possible improvements of the algorithm, and (b) general challenges for all future endeavor in computer-assisted or computational historical linguistics.

4.1 Possible improvements

The possible improvements include to further integrate our preliminary attempts for *semi-automatic reconstruction*, starting from readily identified sound correspondence patterns. Experiments are ongoing in this regard, but we have not yet had time to integrate it fully so far. In general, our workflow also needs a clearer integration of automatic and manual approaches, ideally accompanied by extensive tutorials that would allow users to start with the tools without specifically getting in touch with us and asking us (although we tend to respond quickly to all kinds of inquiries). We should also work on a better handling of output formats, both for correspondence patterns, where we may want to produce some format similar to Ratliff's representation, also to allow readers to make use of the format most convenient to them, and for other aspects of the data. We should also further work on initial attempts on testing the consistency of the data, be it with help of *prediction experiments* (Bodt and List 2019), or with more elaborated quasi-probabilistic accounts that test how well a given word fits into a given cognate set.

4.2 General challenges

General challenges include the full-fledged lexical reconstruction of words, i.e., a reconstruction that would potentially also provide compounds in etymological dictionaries. While current reconstructions (also Ratliff 2010) often only provide morphemes (and thus only carry out a morpheme-based phonological reconstruction, instead of complete reconstructions of the words that were potentially used in the languages before. Furthermore, we will need a convincing annotation of sound change that would ideally allow us to even check which sounds changed when during language history on which branch of the language.

5 Outlook

This draft has provided a rather detailed account on what we consider the current state-of-the-art in computer-assisted language comparison. Starting from raw data, we have shown how these can be successively lifted to higher levels of annotation. While our five-step workflow can be safely applied purely manually, purely automatically, or in a computer-assisted manner, we have shown that even with a pure automatic approach, one can already get quite far in *computer-assisted language comparison.* In the future, we hope to further enhance the workflows and make them more accessible to users from different backgrounds.

References

Anthony, D. W. and D. Ringe (2015). "The Indo-European homeland from linguistic and Archaeological perspectives". *Annual Review of Linguistics* 1, 199–219.

Arnaud, A. S., D. Beck, and G. Kondrak (2017). "Identifying cognate sets across dictionaries of related languages". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. (Copenhagen, 09/07–09/11/2017). Association for Computational Linguistics, 2509–2518.

Barrachina, S. et al. (2008). "Statistical approaches to computer-assisted translation". *Computational Linguistics* 35.1, 3–28.

Beekes, R. S. P. (1996). *Comparative Indo-European linguistics. An introduction*. Amsterdam and Philadelphia: John Benjamins.

Bodt, T. A. and J.-M. List (2019). *Testing the predictive strength of the comparative method: An ongoing experiment on unattested words in Western Kho-Bwa langauges*. Preprint, not peer-reviewed.

Covington, M. A. (1996). "An algorithm to align words for historical comparison". *Computational Linguistics* 22.4, 481–496.

– (1998). "Alignment of multiple languages for historical comparison". In: *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*. "COLING-ACL 1998" (Montreal, 08/10–08/14/1998). Association of Computational Linguistics, 275–279.

Dellert, J. and G. Jäger (2017). *NorthEuraLex (Version 0.9)*. Tübingen: Eberhard-Karls University Tübingen.

Dybo, A. and G. S. Starostin (2008). "In defense of the comparative method, or the end of the Vovin controversy". In: *Aspekty komparativistiki* [Aspects of comparative linguistics]. Vol. 3: *Aspekty komparativistiki*. Ed. by I. S. Smirnov. Moscow: RGGU, 119–258.

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.

Hill, N. W. and J.-M. List (2017). "Challenges of annotation and analysis in computer-assisted language comparison: A case study on Burmish languages". *Yearbook of the Poznań Linguistic Meeting* 3.1, 47–76.

Hoenigswald, H. M. (1960). "Phonetic similarity in internal reconstruction". *Language* 36.2, 191–192. JSTOR: 410982.

Holm, H. J. (2007). "The new arboretum of Indo-European "trees?. Can new algorithms reveal the phylogeny and even prehistory of Indo-European?" *Journal of Quantitative Linguistics* 14.2-3, 167–214.

Kondrak, G. (2000). "A new algorithm for the alignment of phonetic sequences". In: *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*. (Seattle, 04/29–05/03/2000), 288–295.

– (2003). "Identifying complex sound correspondences in bilingual wordlists". In: *Computational linguistics and intelligent text processing*. Ed. by A. Gelbukh. Berlin: Springer, 432–443.

List, J.-M. (2012a). "LexStat. Automatic detection of cognates in multilingual wordlists". In: *Proceedings of the EACL 2012 Joint Workshop of Visualization of Linguistic Patterns and Uncovering Language History from Multilingual Resources*. "LINGVIS & UNCLH 2012" (Avignon, 04/23–04/24/2012). Stroudsburg, 117–125.

– (2012b). "SCA: Phonetic alignment based on sound classes". In: *New directions in logic, language, and computation*. Ed. by M. Slavkovik and D. Lassiter. Berlin and Heidelberg: Springer, 32–51.

– (2014). *Sequence comparison in historical linguistics*. Düsseldorf: Düsseldorf University Press.

– (2016). "Beyond cognacy: Historical relations between words and their implication for phylogenetic reconstruction". *Journal of Language Evolution* 1.2, 119–136.

– (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.

– (2018a). *LingRex: Linguistic Reconstruction with LingPy*. URL: https://doi.org/10.5281/zenodo.1544944.

– (2018b). *Regular cognates: A new term for homology relations in linguistics*. Vol. 5. 8.

– (2018c). *SinoPy: A Python library for quantitative tasks in Chinese historical linguistics*. Version 0.3.1. URL: https://github.com/lingpy/sinopy.

– (2019). "Automatic inference of sound correspondence patterns across multiple languages". *Computational Linguistics* 1.45, 137–161.

List, J.-M., J. S. Pathmanathan, P. Lopez, and E. Bapteste (2016a). “Unity and disunity in evolutionary sciences: process-based analogies open common research avenues for biology and linguistics”. *Biology Direct* 11.39, 1–17.

List, J.-M., P. Lopez, and E. Bapteste (2016b). “Using sequence similarity networks to identify partial cognates in multilingual wordlists”. In: *Proceedings of the Association of Computational Linguistics 2016 (Volume 2: Short Papers)*. Association of Computational Linguistics. Berlin, 599–605.

List, J.-M., S. Greenhill, C. Anderson, T. Mayer, T. Tresoldi, and R. Forkel, eds. (2018a). *CLICS: Database of Cross-Linguistic Colexifications*. URL: `http://clics.clld.org/`.

List, J.-M., S. Greenhill, T. Tresoldi, and R. Forkel (2018b). *LingPy. A Python library for quantitative tasks in historical linguistics*. URL: `http://lingpy.org`.

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

Moran, S. and M. Cysouw (2018). *The Unicode Cookbook for Linguists: Managing writing systems using orthography profiles*. Berlin: Language Science Press.

Ratliff, M. (2010). *Hmong-Mien language history*. Canberra: Pacific Linguistics.

Wahle, J. (2016). “An approach to cross-concept cognacy identification”. In: *Proceedings of the Leiden Workshop on Capturing Phylogenetic Algorithms for Linguistics*. “Capturing Phylogenetic Algorithms for Linguistics” (Leiden, 10/26–10/30/2015). Ed. by C. Bentz, G. Jäger, and I. Yanovich. Tübingen.

Wang, W. S.-Y. (1996). “Linguistic diversity and language relationships”. In: *New horizons in Chinese linguistics*. Ed. by C.-t. J. Huang. Studies in natural language and linguistic theory 36. Dordrecht: Kluwer, 235–267.

陳其光, C. Q. (2012). *Miàoyáo yǔwén* 妙药语文 [Miao and Yao language]. Ed. by Anonymous. Běijīng: Zhōngyāng Mínzú Dàxué 中央民族大学[Central Institute of Minorities].