Families and resemblances
Prokic, Jelena

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2010

Link to publication in University of Groningen/UMCG research database

Citation for published version (APA):

Copyright
Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

Take-down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): http://www.rug.nl/research/portal. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.
Chapter 6

Multiple string alignments in linguistics

In this chapter we present and evaluate an algorithm used to produce multiple sequence alignments in linguistics ALPHAMALIG (Alonso et al., 2004). Originally used for text alignment, we adapted it slightly and applied it to our dialect pronunciation data. The alignments produced are evaluated by comparing them to the manually corrected alignments, the so-called gold standard. The results of evaluating the two alignments show that automatically induced alignments are of a good quality, highly corresponding with the manually produced alignments. This chapter is structured as follows. In Section 6.1 we introduce multiple sequence alignments and give our motivation for using this type of alignment. Section 6.2 gives a description of the ALPHAMALIG algorithm. We then present our gold-standard, but also simple and advanced baseline alignments in Section 6.3. Both advanced and baseline alignments are used to evaluate the quality of the automatically induced alignments. The evaluation of the alignments was done using two novel methods that we present in more detail in Section 6.4. A short discussion and some pointers for future work are given in Section 6.5. Work presented in this chapter was published as Prokić, Wieling, and Nerbonne (2009).

6.1 Multiple sequence aligning

In bioinformatics, sequence alignment is a way of arranging DNA, RNA or protein sequences in order to identify regions of similarity and determine evolutionary, functional or structural similarity between the sequences. There are two main types of string alignment: pairwise and multiple string alignment. Pairwise string alignment methods compare two strings at a time and cannot directly be used to obtain multiple string alignment
methods (Gusfield, 1997, 343-344). In multiple string alignment all strings are aligned and compared at the same time, making it a good technique for discovering patterns, especially those that are weakly preserved and cannot be detected easily from sets of pairwise alignments. Multiple string comparison is considered to be the holy grail of molecular biology (Gusfield, 1997, 332):

It is the most critical cutting-edge tool for extracting and representing biologically important, yet faint or widely dispersed, commonalities from a set of strings.

Multiple string comparison is not new in linguistic research. In the late 19th century the Neogrammarians proposed the hypothesis of the regularity of sound change. According to the Neogrammarian hypothesis sound change occurs regularly and uniformly whenever the appropriate phonetic environment is encountered (Campbell, 2004). Ever since then the understanding of sound change has played a major role in the comparative method that is itself based on the simultaneous comparison of different languages, i.e. lists of cognate terms from the related languages. The correct analysis of sound changes requires the simultaneous examination of corresponding sounds in order to compare hypotheses about their evolution. Alignment identifies which sounds correspond. Historical linguists align the sequences manually, while we seek to automate this process.

In recent years there has been a strong focus in historical linguistics on the introduction of quantitative methods in order to develop tools for the comparison and classification of languages. For example, in his PhD thesis, Kondrak (2002) presents algorithms for the reconstruction of proto-languages from cognates. Warnow et al. (2006) applied methods taken from phylogenetics to Indo-European phonetic data in order to model language evolution. Heeringa and Joseph (2007) applied the Levenshtein algorithm to the Dutch pronunciation data taken from Reeks Nederlandse Dialectatlassen and tried to reconstruct a ‘proto-language’ of Dutch dialects using the pairwise alignments.

Studies in historical linguistics and dialectometry where string comparison is used as a basis for calculating the distances between language varieties will profit from tools to multi-align strings automatically and to calculate the distances between them. Good multiple alignment is of benefit to all those methods in diachronic linguistics such as the comparative reconstruction method or the so-called character-based methods taken from phylogenetics, which have also been successfully applied in linguistics (Gray and Jordan, 2000; Gray and Atkinson, 2003; Atkinson et al., 2005; Warnow et al., 2006). The multi-alignment systems can help historical linguistics by reducing the human labor needed to detect the regular sound correspondences and cognate pairs of words. They also systematize the linguistic knowledge in intuitive alignments and provide a basis for the application of the quantitative methods that lead to a better understanding of language variation and language change.

In this study we apply an iterative pairwise alignment program for linguistics, ALPHAMALIG, to the phonetic transcriptions of words used in dialectological research. We automatically multi-align all transcriptions and compare these generated alignments
with manually aligned gold standard alignments. At the same time we propose two methods for the evaluation of the multiple sequence alignments (MSA).

### 6.1.1 Example of multiple sequence alignment

In this section we will give an example of the automatically multi-aligned strings from our data set and point out some important features of the simultaneous comparison of more than two strings.

![Example of multiple string alignment for six villages. Sign for primary stress is moved to the first vowel in the stressed syllable.](image)

In Figure 6.1 we have multi-aligned pronunciations of the word *a3 /az/ ‘I’ automatically generated by ALPHAMALIG. The advantages of this kind of alignment over pairwise alignment are twofold:

- First, it is easier to detect and process corresponding phones in words and their alternations (like [s] and [z] in the third column in Figure 6.1).
- Second, the distances/similarities between strings can be different in pairwise comparison as opposed to multiple comparison. This is so because multi-aligned strings, unlike pairwise aligned strings, contain information on the positions where phones were inserted or deleted in both strings. For example, in Figure 6.1 the pairwise alignment of the pronunciations from the villages Aldomirovtsi and Beglezh would be:

  Aldomirovtsi: j ‘a - - - -
  Beglezh: - ‘a s - - -

  These two alignments have one matching element out of three in total, which means that the similarity between them is $1/3 = 0.33$. At the same time the similarity between these two strings calculated based on the multi-aligned strings in Figure 6.1 would be $4/6 = 0.66$:
The measurement based on multi-alignment takes the common missing material into account as well. For example, the last three positions are not present in the pairwise alignments, which is, in some cases, an important information loss.

### 6.2 Iterative pairwise alignment

Multiple alignment algorithms iteratively merge two multiple alignments of two subsets of aligned strings into a single multiple alignment that is union of those subsets (Gusfield, 1997). The simplest approach is to align the two strings that have the minimum distance over all pairs of strings and iteratively align strings having the smallest distance to the already aligned strings in order to generate a new multiple alignment. Other algorithms use different initializations and different criteria in selecting the new alignments to merge. Some begin with the longest (low cost) alignment instead of the pair with the least cost absolutely. A string with the smallest edit distance to any of the already merged strings is chosen to be added to the strings in the multiple alignment. In choosing the pair with the minimal distance, all algorithms are greedy, and risk missing optimal alignments.

ALPHAMALIG is an iterative pairwise alignment program for bilingual text alignment. It uses the strategy of merging multiple alignments of subsets of strings, instead of adding just one string at the time to the already aligned strings.\footnote{http://alggen.lsi.upc.es/recerca/align/alphamalig/intro-alphamalig.html} It was originally developed to align corresponding words in bilingual texts, i.e. to work with textual data, but it functions with any data that can be represented as a sequence of symbols of a finite alphabet. In addition to the input sequences, the program needs to know the alphabet and the distances between each token pair and each pair consisting of a token and a gap.

In order to perform multiple sequence alignments of X-SAMPA word transcriptions we modified ALPHAMALIG slightly so it could work with the tokens that consist of more than one symbol, such as [”e"], [”e:"] and [t_’], i.e. IPA [’e"], [’e:"] and [iʃ] respectively. The distances between the tokens were specified in such a way that vowels can be aligned only with vowels and consonants only with consonants. The same tokens are treated as identical and the distance between them is set to 0. The distance between any token in the data set to a gap symbol has the same cost as replacing a vowel with a vowel or a consonant with a consonant. Except for this very general linguistic knowledge, no other data-specific information was given to the program. In this chapter we do not use any phonetic features in order to define the segments more precisely or to calculate the distances between them in a more sensitive way other than making a binary ‘match/does-not-match-distinction’, since we want to keep the system language independent and robust to the highest possible degree.

To illustrate the algorithm we will look at the six pronunciations of the word /a3/ ‘I’ presented in Figure 6.1.
6.2. ITERATIVE PAIRWISE ALIGNMENT

Aldomirovtsi:  j 'a
Beglezh:  'a s
Belene:  'a s
Chukovets:  j 'a z e k a
Dinevo:  j 'a
Dobroselets:  'a s

Figure 6.2: Pronunciations of word ‘I’ collected at six places.

In the first step the algorithm forms 2 groups: pronunciations for villages Aldomirovtsi and Dinevo [j'α] are put in one and pronunciations for villages Beglezh, Belene and Dobroselets ['as] in the other. The distance between the strings within these two groups is 0, i.e. they have the same pronunciation of the word in question. Pronunciation for village Chukovets would still be non-aligned.

<table>
<thead>
<tr>
<th>Aldomirovtsi:</th>
<th>Beglezh:</th>
<th>Belene:</th>
<th>Dobroselets:</th>
</tr>
</thead>
<tbody>
<tr>
<td>j 'a</td>
<td>'a s</td>
<td>'a s</td>
<td>'a s</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chukovets:</th>
</tr>
</thead>
<tbody>
<tr>
<td>j 'a z e k a</td>
</tr>
</tbody>
</table>

Figure 6.3: In the first step strings that have distance 0 are grouped together.

In the next step, pronunciations for villages Aldomirovtsi and Dinevo are aligned with the pronunciations for villages Beglezh, Belene and Dobroselets since the distance between these pronunciations is smaller than the distance between any of the two groups of strings to the pronunciation from village Chukovets. The distance between [j'α] and ['as] is 2, between [j'α] and [j'azeka] is 4, while the distance between ['as] and [j'azeka] is 5.

<table>
<thead>
<tr>
<th>Aldomirovtsi:</th>
<th>Beglezh:</th>
<th>Belene:</th>
<th>Dobroselets:</th>
</tr>
</thead>
<tbody>
<tr>
<td>j 'a</td>
<td>'a s</td>
<td>'a s</td>
<td>'a s</td>
</tr>
<tr>
<td>Dinevo:</td>
<td>j 'a</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chukovets:</th>
</tr>
</thead>
<tbody>
<tr>
<td>j 'a z e k a</td>
</tr>
</tbody>
</table>

Figure 6.4: In the second step strings that have distance 2 are grouped together.

In the last step already aligned strings for 5 villages are aligned against the pronunciation for village Chukovets. The distance between them is 5 since they only match in the second position ['α].
6.3 Gold standard and baseline

In order to evaluate the performance of ALPHAMALIG, we compared the alignments obtained using this program to manually aligned strings, our gold standard, and to the alignments obtained using two very simple techniques that are described next: simple baseline and advanced baseline.

6.3.1 Simple baseline

The simplest way of aligning two strings would be to align the first element from one string with the first element from the other string, the second element with the second and so on. If two strings are not of equal length, the remaining unaligned tokens are aligned with the gap symbol which represents an insertion or a deletion. This is the alignment implicit in Hamming distance, which ignores insertions and deletions.

By applying this simple method, we obtained multiple sequence alignments for all words in our data set. An example of such a multiple sequence alignment is shown in Figure 6.6. These alignments were used to check how difficult the multiple sequence alignment task is for our data and how much improvement is obtained using more advanced techniques to multi-align strings.

Figure 6.6: Simple baseline produced by aligning the first element from one string with the first element from the other string, the second element with the second and so on.
6.3.2 Advanced baseline

Our second baseline is more advanced than the first and was created using the following procedure:

1. for each word the longest string among all pronunciations is located

2. All strings are pairwise aligned against the longest string using the Levenshtein algorithm. We refer to the two sequences in a pairwise alignment as ‘aligned sequences’. Note that aligned sequences include hyphens indicating the places of insertions and deletions.

3. the aligned sequences—all of equal length—are extracted

4. all extracted aligned sequences are placed below each other to form the multiple alignment

An example of combining pairwise alignments against the longest string (in this case [j'azeka]) is shown in Figure 6.7.

Chukovets: j 'a z e k a Chukovets: j 'a z e k a
Aldomirovtsi: j 'a - - - - Dobroeleets: - 'a s - - -

Aldomirovtsi: j 'a - - - -
Chukovets: j 'a z e k a
Dobroeleets: - 'a s - - -

Figure 6.7: Advanced baseline. The top two alignments each contain two aligned sequences, and the bottom one contains three.

6.3.3 Gold standard

Our gold standard was created by manually correcting the advanced baseline alignments described in the previous section. The gold standard results and both baseline results consist of 152 files with multi-aligned strings, one for each word. The pronunciations are ordered alphabetically according to the village they come from. If there are more pronunciations per site, they are all present, one under the other.

6.4 Evaluation

Although multiple sequence alignments are broadly used in molecular biology, there is still no widely accepted objective function for evaluating the goodness of the multiply
aligned strings (Gusfield, 1997). The quality of the existing methods used to produce multiple sequence alignments is judged by the ‘biological meaning of the alignments they produce’. Since strings in linguistics cannot be judged by the biological criteria used in string evaluation in biology, we are forced to propose evaluation methods that are suitable for the strings in question. One of the advantages we have is the existence of the gold standard alignments, which makes our task easier and more straightforward—in order to determine the quality of the multi-aligned strings, we compare outputs of the different algorithms to the gold standard. Since there is no off-the-shelf method that can be used for comparison of multi-aligned strings to a gold standard, we propose two novel methods—one sensitive to the order of positions in two alignments and another that takes into account only the content of each position.

6.4.1 Order dependent method

The first method we develop compares the contents of the position in two alignments and also takes the position sequence into account. A position is a certain vertical position in the multiple alignments and can be best illustrated on the multiply aligned strings in Figure 6.8 (repeated from Figure 6.1) where we mark the first position:

Aldomirovtsi: | j  | 'a  | -  | -  | -  | -  |
Beglezh:    |    | 'a  | s  | -  | -  | -  |
Belene:     |    | 'a  | s  | -  | -  | -  |
Chukovets:  | j  | 'a  | z  | e  | k  | a  |
Dinevo:     | j  | 'a  | -  | -  | -  | -  |
Dobroselets:|    | 'a  | s  | -  | -  | -  |

Figure 6.8: This multiple alignment contains 6 positions. We mark the first position.

The order dependent evaluation (ODE) procedure is as follows:

- Each gold standard column is compared to the most similar column out of two neighboring columns of a candidate multiple alignment. The two neighboring columns depend on the previous matched column \( j \) and have indices \( j + 1 \) and \( j + 2 \) (at the start \( j = 0 \)). It is possible that there are columns in the candidate multiple alignment which remain unmatched, as well as columns at the end of the gold standard which remain unmatched.

- The similarity of a candidate column to a gold standard column is calculated by dividing the number of correctly placed elements in every candidate column by the

\[^{2}\text{In Prokić, Wieling, and Nerbonne (2009) we use the name ‘column dependent method’ for the same method.}\]
total number of elements in the column. A score of 1 indicates perfect overlap, while a score of 0 indicates the columns have no elements in common. This calculation is performed for each column.

- The similarity score of the whole multiple alignment (for a single word) is calculated by summing the similarity score of each candidate column and dividing the resulting sum by the total number of matched columns plus the total number of unmatched columns in both multiple alignments.

- The final similarity score between the set of gold standard alignments with the set of candidate multiple alignments is calculated by averaging the multiple alignment similarity scores for all strings.

As an example consider the multiple alignments in Figure 6.9, with the gold standard alignment (GS) on the left and the generated alignment (GA) on the right.

![Figure 6.9: GS and ALPHAMALIG multiple string alignments, the gold standard alignment left, the ALPHAMALIG output right.](image)

The evaluation starts by comparing the first column of the GS with the first and second column of the GA. The first column of the GA is the best match, since the similarity score between the first columns is 0.75 (3 out of 4 elements match). In similar fashion, the second column of the GS is compared with the second and the third column of the GA and matched with the third column of GA with a similarity score of 1 (all elements match). The third GS column is matched with the fourth GA column, the fourth GS column with the fifth GA column and the fifth GS column with the sixth GA column (all three having a similarity score of 1). As a consequence, the second column of the GA remains unmatched. In total, five columns are matched and one column remains unmatched. The total score of the GA equals:

\[
\frac{(0.75 + 1 + 1 + 1)}{(5 + 1)} = 0.792
\]

It is clear that this method punishes unmatched columns by increasing the value of the denominator in the similarity score calculation. As a consequence, swapped columns are punished severely, which is illustrated in Figure 6.10. In the alignments in Figure 6.10, the first three columns of GS would be matched with the first three columns of GA with a score of 1, the fourth would be matched with the fifth, and two columns would be left unmatched: the fifth GS column and the fourth GA column yielding a
total similarity score of $4/6 = 0.66$. Especially in this case this is undesirable, as both sequences of these columns represent equally reasonable multiple alignment and should have a total similarity score of 1. We therefore need a less strict evaluation method which does not insist on the exact ordering. An alternative method is introduced and discussed in the following section.

Figure 6.10: Two alignments with swapped columns.

6.4.2 Modified Rand index

In developing an alternative evaluation we proceeded from the insight that the columns of a multiple alignment are a sort of partition of the elements of the alignment strings, i.e., they constitute a set of disjoint multi-sets whose union is the entire multi-set of segments in the multiple alignment. Each column effectively assigns its segments to a partition, which clearly cannot overlap with the elements of another column (partition). Since every segment must fall within some column, the assignment is also exhaustive.

Our second evaluation method is therefore based on the modified Rand index (Hubert and Arabie, 1985) described in Chapter 3. The modified Rand index is used in classification for comparing two different partitions of a finite set of objects. In Chapter 3 we have used it to compare the classification of sites done by various clustering algorithms to the traditional division of the sites. In this chapter we use it to assess the quality of each column from the automatically induced multiple sequence alignments.

We would like to emphasize that it is clear that the set of columns of a multi-alignment have more structure than a partition sec; in particular because the columns (subpartitions) are ordered, unlike the subpartitions in a partition. But we shall compensate for this difference by explicitly marking order.

Figure 6.11: Annotated multiple sequence alignment.

In our study, each segment token in each transcription is treated as a different object (see Figure 6.11), and every column is taken to be a sub-partition to which segment
tokens are assigned. Both alignments in Figure 6.10 have 12 phones that are put into 5 groups. We ‘tag’ each token sequentially in order to distinguish the different tokens of a single segment from each other, but note that the way we do this also introduces an order sensitivity in the measure. Since columns 4 and 5 in two of the multiple sequence alignments in Figure 6.10 are swapped, we obtain the following two partitions:

\[
\begin{align*}
GS1 &= \{1,5,9\} & GA1 &= \{1,5,9\} \\
GS2 &= \{2,6,10\} & GA2 &= \{2,6,10\} \\
GS3 &= \{3,7,11\} & GA3 &= \{3,7,11\} \\
GS4 &= \{4,12\} & GA4 &= \{8\} \\
GS5 &= \{8\} & GA5 &= \{4,12\}
\end{align*}
\]

Using the modified Rand index the quality of each column is checked, regardless of whether the columns are in order. The MRI for the alignments in Figure 6.10 will be 1, because both alignments group segment tokens in the same way. Even though columns four and five are swapped, in both classifications phones [j] and [f] are grouped together, while sound [u] forms a separate group.

The MRI itself only takes into account the quality of each column separately since it simply checks whether the same elements are together in the candidate alignment as in the gold-standard alignment. It is therefore insensitive to the ordering of columns. While it may have seemed counterintuitive linguistically to proceed from an order-insensitive measure, the comparison of ‘tagged tokens’ described above effectively reintroduces order sensitivity.

In the next section we describe the results of applying both evaluation methods on the automatically generated multiple alignments.

### 6.4.3 Results

After comparing all files of the baseline algorithms and ALPHAMALIG against the gold standard files according to the order dependent evaluation method and the modified Rand index, the average score is calculated by summing up all scores and dividing them by the number of word files (152).

The results are given in Table 6.1 and also include the number of words with perfect multi-alignments (i.e. identical to the gold standard). Using ODE, ALPHAMALIG scored 0.932 out of 1.0 with 103 perfectly aligned files. The result for the simple baseline was 0.710 with 44 perfectly aligned files. As expected, the result for the advanced baseline was in between these two results—0.869 with 72 files that were completely identical to the GS files. Using MRI to evaluate the alignments generated we obtained generally higher scores for all three algorithms, but with the same ordering. ALPHAMALIG scored 0.982, with 104 perfectly aligned files. The advanced baseline had a lower score of 0.937 and 74 perfect alignments. The simple baseline performed worse, scoring 0.848 and having 44 perfectly aligned files.
Table 6.1: Results of evaluating outputs of the different algorithms against the GS.

<table>
<thead>
<tr>
<th></th>
<th>ODE</th>
<th>ODE perfect columns</th>
<th>MRI</th>
<th>MRI perfect columns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple baseline</td>
<td>0.710</td>
<td>44</td>
<td>0.848</td>
<td>44</td>
</tr>
<tr>
<td>Advanced baseline</td>
<td>0.869</td>
<td>72</td>
<td>0.937</td>
<td>74</td>
</tr>
<tr>
<td>ALPHAMALIG</td>
<td>0.932</td>
<td>103</td>
<td>0.982</td>
<td>104</td>
</tr>
</tbody>
</table>

The scores of the ODE evaluation method are lower than the MRI scores, which is due to the first method’s problematic sensitivity to column ordering in the alignments. It is clear that in both evaluation methods ALPHAMALIG outperforms both baseline alignments by a wide margin.

It is important to notice that the scores for the simple baseline are reasonably high, which can be explained by the structure of our data set. The variation of word pronunciations is relatively small, making string alignment easier. However, ALPHAMALIG obtained much higher scores using both evaluation methods.

Additional qualitative error analysis reveals that the errors of ALPHAMALIG are mostly caused by the vowel-vowel consonant-consonant alignment restriction. In the data set there are 21 words that contain metathesis, i.e. switched sounds within the words. More on metathesis can be found in Section 5.2.1. Since vowel-consonant alignments were not allowed in ALPHAMALIG, alignments produced by this algorithm were different from the gold standard, as illustrated in Figure 6.12. The vowel-consonant restriction is also responsible for wrong alignments in some words where metathesis is not present, but where the vowel-consonant alignment is still preferred over aligning vowels and/or consonants with a gap (see for example Figure 6.9).

The other type of error present in the ALPHAMALIG alignments is caused by the fact that all vowel-vowel and consonant-consonant distances receive the same weight. In Figure 6.13 the alignment of word /saxme/ ‘were - 1st pl’ produced by ALPHAMALIG is wrong because instead of aligning [m] with [m] and [m] it is wrongly aligned with two tokens of [x], while a third token of [x] is aligned with [f] instead of aligning it with [x] and [x]. This is the sort of error which segment-weighted alignments...
6.5. DISCUSSION

In this chapter we have presented a technique to automatically multi-align phonetic transcriptions. We have also introduced two novel techniques that can be used to evaluate the quality of the multi-aligned strings. Both evaluation methods are based on comparing the automatically induced alignments to gold-standard alignments. The results have shown that the automatically produced multi-alignments are of a good quality with less than 2 per cent error on the segment level. However, in order to apply either of these two methods it is necessary to have a gold standard alignment against which the automatically induced alignments are compared to. We are aware that for many data sets this is neither available nor easily obtainable. But in cases where it exists, we find these techniques a useful evaluation tool.

The comparison between our simple baseline to the gold standard alignments has shown that our data set contains strings with relatively simple structure. Pronunciation variation in our dialects is relatively small, especially if compared to cross-linguistic data. The structure of syllables is also relatively simple, very often showing only CV structure. It would be very important to apply the ALPHAMALIG algorithm to the data from some other languages in order to obtain further insight into the quality of the alignments produced.

In the alignment procedure, we have used vowel-vowel consonant-consonant restriction as the only ‘linguistic’ knowledge given as an input to the ALPHAMALIG algorithm. This, on one side, makes the alignment robust and language independent, but, on the other, introduces some errors in the alignments like those presented in Figure 6.13. We believe that the quality of the generated alignments could be further improved if some kind of segment weighting were introduced into the alignment procedure, such as the one presented in Chapter 5. Weighting of the segments could also enable the vowel-vowel consonant-consonant constraint to be completely eliminated from the aligning. This could lead to better alignments in the cases where vowels and consonants need to be aligned (see for example Figure 6.12).

The automatic multi-aligning could be improved in many other ways. There are
also various algorithms used to multi-align sequences in biology. Some of them could potentially be adopted to work with the strings in linguistics. We hope that our first experiments with the multiple aligned phonetic transcriptions have shown the usefulness of this type of approach to string comparison in linguistics and that in future further experiments in this direction will be conducted.