

ADAPT

Algorithm for Dynamic Alignment of Phonetic Transcriptions

Bram Elffers, Christophe Van Bael, Helmer Strik

bramelf@lands.let.ru.nl
c.v.bael@let.ru.nl
h.strik@let.ru.nl

Department of Language and Speech
Radboud University Nijmegen
The Netherlands

1. Introduction

ADAPT (*Algorithm for Dynamic Alignment of Phonetic Transcriptions*) is a dynamic programming algorithm that computes the optimal alignment between two strings of phonetic symbols. The optimal alignment is computed from a matrix in which the distances between phonetic symbols are defined in terms of articulatory features. The optimal alignment yields a distance measure, expressing the phonetic similarity of the transcriptions.

This document discusses the purpose of using ADAPT, provides a technical description of its alignment algorithm, and documents its usage.

2. Purpose

ADAPT can be used to determine the phonetic similarity of two transcriptions of a spoken utterance. These transcriptions, provided as strings of phonetic symbols, typically represent a *reference transcription* (describing the optimal pronunciation of the utterance) and a *hypothesis transcription* (describing an actual pronunciation of the utterance). A smaller distance between the hypothesis transcription and its reference transcription indicates a greater phonetic similarity, usually interpreted as a higher quality assessment of the hypothesis transcription.

Given two strings of phonetic symbols, ADAPT produces two types of output. Firstly, it produces the optimal alignment between the symbols in the two transcriptions. Secondly, it produces two statistic measures given this optimal alignment: the *number of substitutions, deletions, and insertions* at the phone level, and the overall *phonetic distance* between the reference and hypothesis transcriptions.

For example, imagine a reference transcription and a hypothesis transcription of the word *hello*: /hElɔ/ resp. /@lɔw/ (see Appendix A for a list of common phonetic transcription symbols). If we let ADAPT align the two transcriptions, the result will be:

Reference transcription	hElɔ-
Hypothesis transcription	-@lɔw

The alignment shows that the substring /@lɔ/ from the realisation transcription is optimally aligned to the substring /Elɔ/ from the reference transcription. Since no symbol in the hypothesis transcription is aligned to /h/ in the reference transcription, this /h/ is considered to be *deleted*. Similarly, the presence of /w/ in the hypothesis transcription (absent in the reference transcription) is an example of a phone *insertion*. Since /@/ in the hypothesis transcription is aligned with /E/ in the reference transcription, the /E/ is considered to be substituted by /@/. Note that deletions, insertions, and substitutions are always observed from the viewpoint of the reference transcription.

For another example, consider two transcriptions of the Dutch phrase *op een gegeven moment*. (at a given moment). The first transcription is a canonical reference transcription, indicating how the phrase is pronounced when speaking carefully. The hypothesis transcription represents an extremely reduced realisation of the same utterance. ADAPT’s alignment of the hypothesis and reference transcriptions yields the following alignment:

Orthographic transcription	op een gegeven moment
Reference transcription	Op @n G@Gev@n momEnt
Hypothesis transcription	-p @- --xef@- --mEnt

This alignment shows two substitutions (/G/ as /x/ and /v/ as /f/), seven deletions (/ɔ/, /n/, /G/, /@/, /n/, /m/, and /o/), and no insertions. The total phonetic distance between the reference transcription and the hypothesis transcription is 22.5. This is calculated as the sum of phonetic distances between each aligned phoneme pair, i.e. the sum of the costs for two substitutions and seven deletions.

In this example, note that *word boundaries* (signified by the vertical bars ‘|’) were inserted to guarantee the alignment of each word in the reference transcription with each word in the realisation transcription. If the word boundaries were omitted, the alignment would have looked slightly different:

Orthographic transcription	op een gegeven moment
Reference transcription	Op@nG@Gev@nmomEnt
Hypothesis transcription	-p---@xef@---mEnt

This alignment is incorrect in assigning the first /@/ phoneme in the realisation to ‘gegeven’ as opposed to ‘een’. However, the alignment algorithm does not distinguish the first two /@/ symbols if no word boundaries are supplied. Note that, even though this alignment is in fact incorrect, the number of substitutions, insertions, and deletions, as well as the phonetic distance (22.5) are computed correctly from one of several alternative optimal alignments.

3. Description

This section describes ADAPT's internal representation of phonemes as feature vectors, and the computation of *phonetic distance* between two phonemes, based on these vectors. The last subsection describes the alignment process, and the expression of similarity between two transcriptions in terms of *total phonetic distance* and number of *substitutions*, *deletions*, and *insertions*.

3.1. Phoneme feature vectors

In ADAPT, each phoneme is defined as a feature vector, and signified by a single character. The feature vectors of vowels are defined as sets of five real values, representing *length*, *front/back*, *tongue*, *roundedness*, and *diphthong*. The feature vectors for consonants are defined as sets of eleven real values, representing *place of articulation*, *voice*, *nasal*, *stop*, *glide*, *lateral*, *fricative*, *trill*, *aspirant*, *dental*, and *strength*. The definition of phonemes in terms of phonetic feature vectors was based on Cucchiari [1].

The feature vector declarations for vowels and consonants are stored in a file named *features.txt*.¹ The example file contains 23 vowels, and 24 consonants, with feature values applicable to the Dutch language. If needed, the feature definitions can be altered to reflect the phonetic properties of other languages. Also, it is possible to remove or add phonemes, as well as separate features for vowels or consonants.

In the example feature vector declarations in *features.txt*, each feature is abbreviated as shown in the tables below.

Vowel features	
len	length
fbk	front/back
tng	tongue placing
rnd	roundedness
dph	diphthong

Consonant features	
plc	Place of articulation
voc	Voicing
nas	Nasal
stp	Stop
gld	Glide
lat	Lateral
frc	Fricative
trl	Trill
asp	Aspirant
dnt	Dental
str	Strength

¹ This file must be present in the directory ADAPT is called from.

Prior to calculating the optimal alignment between two phonetic transcriptions, ADAPT reads the file named *features.txt* to retrieve the feature vectors for each phoneme. For every pair of vowels, and for every pair of consonants, it computes the absolute difference between the two phoneme feature vectors as the sum of the absolute difference between every feature value pair. The absolute vector difference is memorised as the phonetic distance between two phonemes.

3.2. Phonetic distance

In ADAPT, the *phonetic distance* between two symbols is defined as the sum of absolute differences between the feature values of the two phonemes they represent. For example, the phonetic distance between /t/ and /d/ is defined as the absolute difference between their feature vectors (i.e. the sum of the differences between each of their feature values):

<i>feature</i>	<i>plc</i>	<i>voc</i>	<i>nas</i>	<i>stp</i>	<i>gld</i>	<i>lat</i>	<i>frc</i>	<i>trl</i>	<i>asp</i>	<i>dnt</i>	<i>str</i>	<i>total</i>
<i>/t/ vector</i>	4.0	1.0	0.0	0.5	0.0	0.0	0.0	0.0	1.0	1.0	1.0	
<i>/d/ vector</i>	4.0	2.0	0.0	0.5	0.0	0.0	0.0	0.0	1.0	1.0	1.0	
<i>difference</i>	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0

Note that the distances between several phoneme pairs may be identical, but composed differently:

<i>feature</i>	<i>plc</i>	<i>voc</i>	<i>nas</i>	<i>stp</i>	<i>gld</i>	<i>lat</i>	<i>frc</i>	<i>trl</i>	<i>asp</i>	<i>dnt</i>	<i>str</i>	<i>total</i>
<i>/t/ vector</i>	4.0	1.0	0.0	0.5	0.0	0.0	0.0	0.0	1.0	1.0	1.0	
<i>/s/ vector</i>	4.0	1.0	0.0	0.0	0.0	0.0	0.5	0.0	1.0	1.0	1.0	
<i>difference</i>	0.0	0.0	0.0	0.5	0.0	0.0	0.5	0.0	0.0	0.0	0.0	1.0

If ADAPT is forced to decide which of the phonemes /d/ and /s/ is phonetically closer to the phoneme /t/, it will first check the feature vector differences of (t, d) and (t, s) as a whole. If these are equal (in this case, 1.0), the phoneme with the least number of different features is assumed to be phonetically closer. Since /d/ has a single feature difference of 1.0, and /s/ has a two feature differences of 0.5 compared to /t/, ADAPT assumes that the phonetic distance between /d/ and /t/ is less than the phonetic distance between /s/ and /t/.

Phonetic distance is defined for pairs of consonants and pairs of vowels only. Since the feature vectors for vowels and consonants are defined by different (and therefore incomparable) features, ADAPT cannot determine the phonetic distance between a vowel and a consonant. Instead, ADAPT assigns a fixed high distance to any vowel/consonant pair, which prevents their alignment.

3.3. Alignment

Based on the phonetic distance between pairs of phonemes, ADAPT constructs a *search space* in terms of a matrix of symbol comparisons. This matrix is used to compare every phoneme in the reference transcription to every phoneme in the hypothesis transcription. The sequence of substitutions, deletions, and insertions which yields the minimum phonetic distance between

the reference and hypothesis transcriptions is represented by the optimal path through search space.

Once the optimal alignment between the reference and hypothesis transcriptions has been determined, ADAPT computes and reports the total *number of substitutions, deletions, and insertions*. Finally, ADAPT also reports the *total phonetic distance*, where every insertion or deletion is assigned a phonetic distance of 3.0, and every substitution is assigned a variable phonetic distance, according to the feature vector difference between each coaligned phoneme pair (as discussed above).

4. Usage

ADAPT is provided as a Perl module. This allows any Perl script to include ADAPT, and use its functionality. In order to use the module, the following files are required:

<code>adapt.pm</code>	The ADAPT module
<code>features.txt</code>	A data file containing phoneme feature declarations

ADAPT will abort with an appropriate error message if *features.txt* is not found.

The ADAPT module is supplied with a Perl script named *run_adapt.pl*. This script serves to illustrate the usage of ADAPT, and can easily be modified to serve its user's specific needs. A custom Perl script may include ADAPT for its own processing by placing a `use` statement below the magic line, as follows:

```
#!/usr/bin/perl
use adapt;
```

When the program using the module is executed, ADAPT will first perform its memorisation step, after which control is passed back the main program.

The script *run_adapt.pl* is called with two command line parameters, as follows:

```
run_adapt.pl references.txt hypotheses.txt
```

The output of the script can be redirected to a file in the usual manner:

```
run_adapt.pl references.txt hypotheses.txt >output.txt
```

If *references.txt* contains a list of reference transcriptions, and *hypotheses.txt* contains a list of hypothesis transcriptions, *output.txt* will contain the alignment results and statistics for each reference/hypothesis alignment. The output will also contain file statistics, showing the averages and totals for all alignments.

In order to compare a norm and a realisation transcription, the main script should call the subroutine *align_dist* in the ADAPT module. The subroutine requires a reference transcription and a hypothesis transcription as its parameters.² It returns an array containing six values: the

² Note that every symbol occurring in the reference and hypothesis transcriptions must be declared in *features.txt*.

total distance (unsigned real), the number of substitutions (unsigned int), the number of deletions (unsigned int), the number of insertions (unsigned int), the aligned reference transcription (string), and the aligned hypothesis transcription (string). For the *hello* example, you would use the following call to the *align_dist* subroutine:

```
$norm = "hElo";
$real = "@low";
($dist, $sub, $del, $ins, $al_ref, $al_real) = &align_dist ($norm, $real);
```

The result of this call is as follows:

<i>Variable</i>	<i>Value</i>	<i>Description</i>
\$dist	7.5	Total distance
\$sub	1	Number of substitutions
\$del	1	Number of deletions
\$ins	1	Number of insertions
\$al_ref	"hElo-"	Aligned reference transcription
\$al_real	"-@low"	Aligned hypothesis transcription

Calls to the *align_dist* subroutine follow the usual Perl syntax conventions, allowing for several methods, whichever is most comfortable given the application. The following calls are valid (if obscure):

```
@results = &align_dist ("hElo", "@low");
($dist, $s, $d, $i, $aref, $areal) = &align_dist (@pair);
```

By default, the 'void phoneme' that represents insertions or deletions is signified by a dash character (-). If for some application a different value is required, the following line in *adapt.pm* can be modified to a different symbol, such as "0".

```
$zero_char = "0";
```

References

1. Cucchiarini, C. (1993). *Phonetic transcription: a methodological and empirical study*. Ph.D. dissertation, University of Nijmegen, the Netherlands.
2. Cucchiarini, C. (1996). *Assessing transcription agreement: methodological aspects*. In *Clinical Linguistics & Phonetics*, 10/2, pp. 131-155.

Appendix A: Phonetic symbols for Dutch

This appendix shows the phonetic symbols as defined by Cucchiarini [1,2]. The phonetic symbols and feature vector declarations may be altered or modified for different languages. They may also be modified to allow a higher degree of phonetic detail in the transcriptions and their alignments. Currently, the encoding of phonemes is restricted to single characters; future versions of ADAPT may include multi-character (SAMPA-compliant) encoding.

Consonants

Plosives

ADAPT	SAMPA	CGN	Example
p	p	p	<u>p</u> ak
b	b	b	ba <u>k</u>
t	t	t	<u>t</u> ak
d	d	d	<u>d</u> ak
k	k	k	<u>k</u> ap
g	g	g	<u>g</u> oal

Fricatives

f	f	f	<u>f</u> el
v	v	v	<u>v</u> el
s	s	s	<u>s</u> ein
z	z	z	<u>z</u> ijn
S	S	S	<u>S</u> how
Z	Z	Z	ba <u>g</u> age
G	G	G	<u>G</u> oed
X/x	x	x	to <u>ch</u>
h	h	h	<u>h</u> and

Sonorants

m	m	m	<u>m</u> et
n	n	n	<u>n</u> et
N	N	N	ba <u>N</u> g
l	l	l	<u>l</u> and
r	r	r	<u>r</u> and
w	w	w	<u>w</u> it
j	j	j	<u>j</u> a
J		J	cam <u>p</u> agne

Vowels

i	i	i	<u>vier</u>
e	e	e	<u>veer</u>
a	a	a	<u>naam</u>
o	o	o	<u>voor</u>
u	u	u	<u>voer</u>
y	y	y	<u>vuur</u>
I	I	I	<u>pit</u>
E	E	E	<u>pet</u>
A	A	A	<u>pad</u>
O	O	O	<u>pot</u>
Y	Y	Y	<u>put</u>
@	@	@	<u>gemak</u>
&	Ei	E+	<u>fijn</u>
1	9y	Y+	<u>huis</u>
2		2	<u>deur</u>
3	Au	A+	<u>goud</u>
4	E:	E:	<u>crème</u>
5	9:	Y:	<u>freule</u>
6	O:	O:	<u>roze</u>
7		E~	<u>bassin</u>
8		A~	<u>ensemble</u>
9		O~	<u>compagnon, fond</u>
%		Y~	<u>vingt-et-un</u>

Appendix B: Phoneme feature definitions for Dutch

This appendix shows the example phoneme feature definitions for the Dutch language, as defined by Cucchiaroni [1,2]. These feature values are found in the file named *features.txt*, which is required by the ADAPT module.

Consonants

	plc	voc	nas	stp	gld	lat	frc	trl	asp	dnt	str
p	5.0	1.0	0.0	0.5	0.0	0.0	0.0	0.0	1.0	1.0	1.0
b	5.0	2.0	0.0	0.5	0.0	0.0	0.0	0.0	1.0	1.0	1.0
t	4.0	1.0	0.0	0.5	0.0	0.0	0.0	0.0	1.0	1.0	1.0
d	4.0	2.0	0.0	0.5	0.0	0.0	0.0	0.0	1.0	1.0	1.0
k	2.0	1.0	0.0	0.5	0.0	0.0	0.0	0.0	1.0	1.0	1.0
g	2.0	2.0	0.0	0.5	0.0	0.0	0.0	0.0	1.0	1.0	1.0
f	5.0	1.0	0.0	0.0	0.0	0.0	0.5	0.0	1.0	1.0	1.0
v	5.0	2.0	0.0	0.0	0.0	0.0	0.5	0.0	1.0	1.0	1.0
s	4.0	1.0	0.0	0.0	0.0	0.0	0.5	0.0	1.0	1.0	1.0
z	4.0	2.0	0.0	0.0	0.0	0.0	0.5	0.0	1.0	1.0	1.0
S	3.0	1.0	0.0	0.0	0.0	0.0	0.5	0.0	1.0	1.0	1.0
Z	3.0	2.0	0.0	0.0	0.0	0.0	0.5	0.0	1.0	1.0	1.0
G	2.0	1.0	0.0	0.0	0.0	0.0	0.5	0.0	1.0	1.0	1.0
X	2.0	1.0	0.0	0.0	0.0	0.0	0.5	0.5	1.0	1.0	1.0
x	2.0	1.0	0.0	0.0	0.0	0.0	0.5	0.5	1.0	1.0	1.0
m	5.0	2.0	0.5	0.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0
n	4.0	2.0	0.5	0.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0
N	2.0	2.0	0.5	0.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0
J	3.0	2.0	0.5	0.0	0.5	0.0	0.0	0.0	1.0	1.0	1.0
l	4.0	2.0	0.0	0.0	0.0	0.5	0.0	0.0	1.0	1.0	1.0
r	4.0	2.0	0.0	0.0	0.0	0.0	0.0	0.5	1.0	1.0	1.0
w	5.0	2.0	0.0	0.0	0.5	0.0	0.0	0.0	1.0	1.0	1.0
j	3.0	2.0	0.0	0.0	0.5	0.0	0.0	0.0	1.0	1.0	1.0
h	1.0	1.0	0.0	0.0	0.0	0.0	0.5	0.0	1.0	1.0	1.0

Vowels

	len	fbk	tng	rnd	dph
i	2.0	3.0	4.0	1.0	1.0
I	1.0	2.5	3.0	1.0	1.0
e	2.0	3.0	3.0	1.0	1.5
2	2.0	2.5	3.0	2.0	1.5
E	1.0	3.0	2.0	1.0	1.0
a	2.0	2.5	1.0	1.0	1.0
A	1.0	1.0	1.0	1.0	1.0
o	2.0	1.0	3.0	2.0	1.5
O	1.0	1.0	2.0	2.0	1.0
u	2.0	1.0	4.0	2.0	1.0
y	2.0	2.5	4.0	2.0	1.0
Y	1.0	2.5	3.0	2.0	1.0
@	1.0	2.0	2.5	1.0	1.0
&	2.0	3.0	1.5	1.0	2.0
1	2.0	2.5	1.5	1.0	2.0
3	2.0	1.0	1.5	1.0	2.0
4	2.0	3.0	2.0	1.0	1.0
5	3.0	2.5	3.0	2.0	1.5
6	2.0	1.0	2.0	2.0	1.0
7	1.0	3.0	3.0	1.0	1.0
8	2.0	1.0	1.5	1.0	1.0
9	2.5	1.0	2.0	2.0	1.0
%	2.5	2.5	3.0	2.0	1.0