Testing the Robustness of Linguistic Distances

Wilbert Heeringa & Franz Manni
University of Groningen, Faculty of Arts, Humanities Computing, Groningen
National Museum of Natural History, Eco-Anthropology Group, Paris

Statistics Seminar
Wednesday, April 19, 2006

Introduction

- Linguistic distance measurements between dialects based on a limited number of words.
- What is the minimum words required to obtain reliable results?
  Measure consistency.
- What is the influence of the selection of the words on the results?
  Perform bootstrapping.
- We consider pronunciation and lexical measurements.

Overview

Measurements

Finding pronunciation distances

- Pronunciation includes phonetic and morphologic variation.
- Pronunciation differences measured with Levenshtein distance.

Algorithm

- *Levenshtein distance* was applied for comparison of Irish dialects by Kessler in 1995. Later it was applied to Dutch, Sardinian, Norwegian, American English, German and Bulgarian dialects, and Bantu languages.
- Calculate the cost of changing one string into another.
- Example: *afternoon* may be pronounced as [ˈæəftəˌnʉˑn] in the dialect of Savannah and as [ˌæftərˈnuˑn] in the dialect of Lancaster.
- Change the first pronounciation into the other.

æəftənʉn	delete ə	1
æftənʉn	insert r	1
æftərnʉn	subst. ʉ/u	1
æftərnun		
		3

Algorithm

- Many sequence operations map [æəftənʉn] $\longrightarrow$ [æftərnun]. Levenshtein distance = cost of cheapest mapping.
- The sum of the operations is divided by the length of the longest alignment which gives the minimum cost. The longest alignment has the greatest number of matches.
- Example:

æ	ə	f	t	ə	∅	n	ʉ	n
æ	∅	f	t	ə	r	n	u	n
	1				1		1	

  A total cost of 3 divided by a length of 9 gives a word distance of 0.33 or 33%.
- Using m words the distance between two dialects is equal to the average of m Levenshtein distances.
- All distances between n dialects are arranged in a $n \times n$ matrix.

Algorithm

- Refinement: use gradual acoustic segment distances as operation weights.
- We used the logarithmic segment distances: greater distances are reduced more than smaller ones.
- We assure that the minimum cost is based on a alignment in which
  - a vowel matches with a vowel
  - a consonant matches with a consonant
  - the [j] or [w] matches with a vowel
  - the [i] or [u] matches with a consonant
  - the schwa matches with a sonorant

Aggregate

- Simplified example: Find distance between Middelstum and Ommen on the basis of 6 words. Diacritics are ignored, all operation costs have a weight of 1.

	Middelstum	Ommen		
schip	sxɪp	sxɪp	0/4	0
pet	pɛt	pɛtə	1/4	0.25
geroepen	rɔupm	ərupm	2/6	0.33
springen	sprɪŋ	sprɪŋkt	2/7	0.29
kelder	kɛlər	kɛldər	1/6	0.17
huis	hus	hys	1/3	0.33
				1.37

- Distance: $(1.37/6) \times 100 = 23\%$.
- From the RND questionaire we selected 125 words. Since only word pairs are considered which have the same lexemes, the number of word pairs per dialect pair may vary.

Finding lexical distances

- We use Goebl's Gewichteter Identitätswert.
- Similarity weighted by frequency of lexemes.
- We want to obtain distances. Therefore we calculate inverse similarity weights.
- Distance between *similar* lexemes varies between $2/n$ (mininum number of similar lexemes) and 1 (if lexemes across all dialects are the same), where n is the number of varieties.
- Example: In our 360 RND dialects we found the following lexemes for *schip*: schip (353), boot (2), lager (1), schuit (4).

schip	vs.	schip	:	353/360	=	0.981
schuit	vs.	schuit	:	4/360	=	0.011
boot	vs.	boot	:	2/360	=	0.006

- Distance between *different* lexemes is *always* 1.

Aggregate

- Example: Find distance between Middelstum and Ommen on the basis of 6 words.

	Middelstum	Ommen		
vriend	kɑmərʊˑt̯	kɑmərɔːt	140/354	0.40
schip	sxɪp	sxɪp	353/360	0.98
ver	v̥ɛːr	ʋit̯		1
zijn	bɪn̩	bɪnt	176/360	0.49
nog	nɔx	nɔx	354/355	1.00
duwen	støˑᵗn̩	drʏk̯ŋ̍		1
				4.87

- Distance: $(4.87/6) \times 100 = 81.2\%$.

Consistency

Consistency

- Cronbach's α is a popular method to measure consistency or reliability.
- When calculating the distances between n_v varieties on the basis of n_w words, n_w matrices are obtained, each containing the distances between the n_v varieties on the basis of the pronunciations of one word.
- The average inter-correlation $\bar{r}$ among the words is calculated as:

$$\bar{r} = \frac{\sum_{i=2}^{n_w} \sum_{j=1}^{i-1} r(w_i, w_j)}{\frac{n_w \times (n_w - 1)}{2}}$$

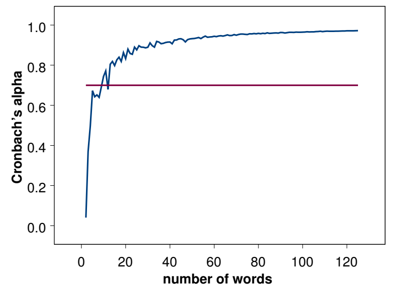
- Cronbach's α can be written as a function of the number of words and the average inter-correlation among the words:

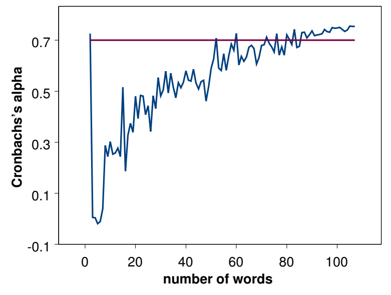
$$\alpha = \frac{n_w \times \bar{r}}{1 + (n_w - 1) \times \bar{r}}$$

Data source

- Reeks Nederlandse Dialectatlassen (RND).
- Compiled by E. Blancquaert and W. Pée.
- Published by 'de Sikkel', Antwerp.
- 16 volumes, 16 transcribers.
- 1956 dialects
- Texts in phonetic script, consisting of 139 sentences.
- Texts are in phonetic script.
- About 2-5 informants per site.
- We selected 360 dialects, 125 words.
- 125 words vary phonetically, 107 words vary lexically.

Consistency





Left: Cronbach's α values for random subsets of 2 through 125 words (pronunciation). From 13 words on α is always higher than 0.70. For 125 words α is equal to 0.97. Right: Cronbach's α values for random subsets of 2 through 107 words (lexis). From 86 words on α is always higher than 0.70. For 107 words α is equal to 0.75.

Classification

Clustering

	Grouw	Haarlem	Delft	Hattem	Lochem
Grouw	0	41	44	45	46
Haarlem	41	0	16	34	36
Delft	44	16	0	37	38
Hattem	45	34	37	0	20
Lochem	46	36	38	20	0

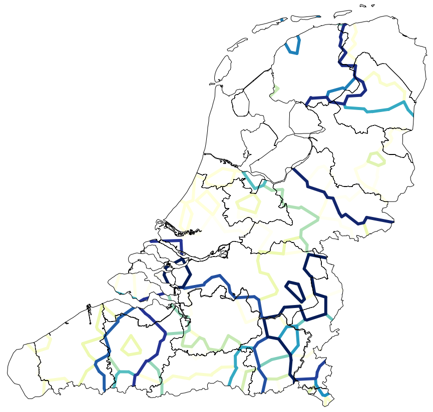
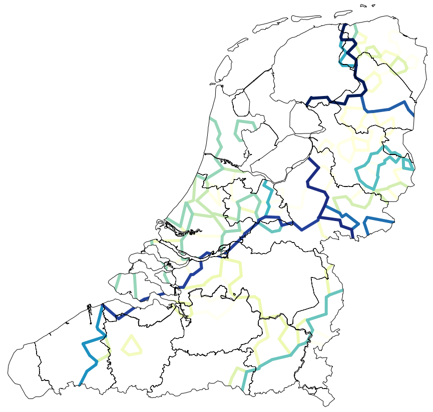
Apply Johnson's algorithm to the upper half of the matrix (blue values):

- Iteratively,
  1. select shortest distance in matrix,
  2. fuse the two datapoints involved.
- To iterate, we have to assign a distance from the newly formed cluster to all other points (several alternatives, we used UPGMA).
- Repeat until one cluster is left over.

Composite Cluster Map

- Introduced by Peter Kleiweg
- Darker lines separate distant groups, lighter lines more similar groups.
- In the first step the border between the two most significant groups is drawn.
- In the second step, two borders are drawn which separate the three most significant groups.
- The first border, which was already drawn in the previous step, is drawn again, resulting in a darker color.
- The second border is drawn for the first time, so it will be lighter than the other one.
- In the i-th step, the $i - 1$ borders of the i most significant groups are drawn again (they get darker), and a new border is added.
- If the cluster contains n varieties, we start with drawing borders which separate the 2 most signficant groups, and end with drawing borders which separate the n most significant groups.

Clustering

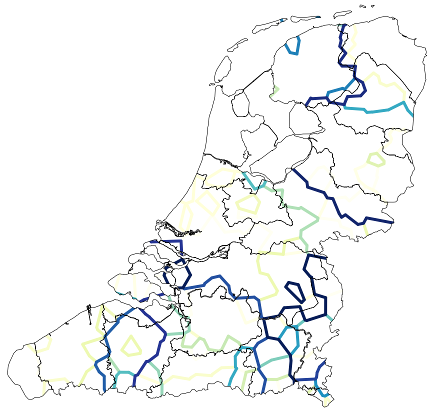
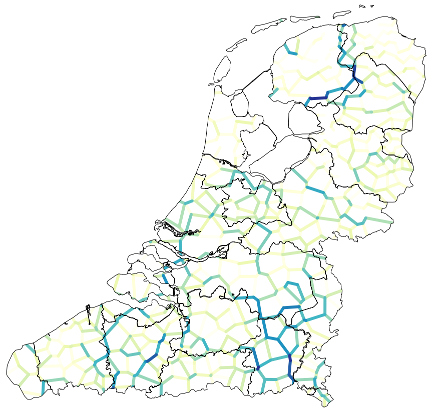
Two composite cluster maps. Left: map obtained on the basis of pronunciation distances. Right: map obtained on the basis of lexical distances.

Bootstrapping

Aggregate on classification level

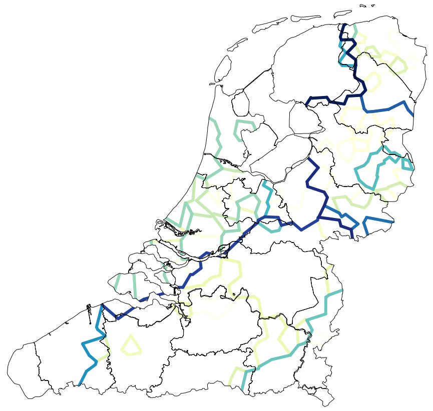
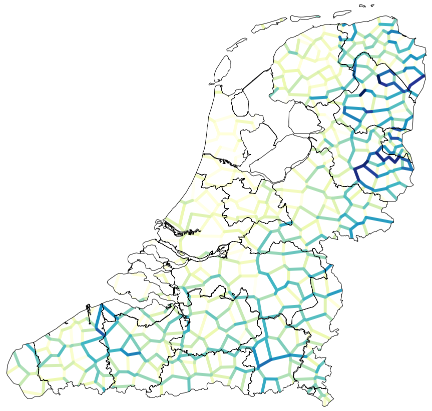
- We have one matrix per word. The distances in the matrix are based on the variation of that word (pronunciation or lexical).
- Create a composite cluster map for each matrix.
- We have n words, so we get n composite cluster maps.
- Show the aggregate of the n maps.

Pronunciation

Two composite cluster maps. The left one is obtained on the basis of the aggregate of 125 distance matrices. The right one is obtained on the basis of the aggregate of 125 single word-based cluster maps.

Lexis

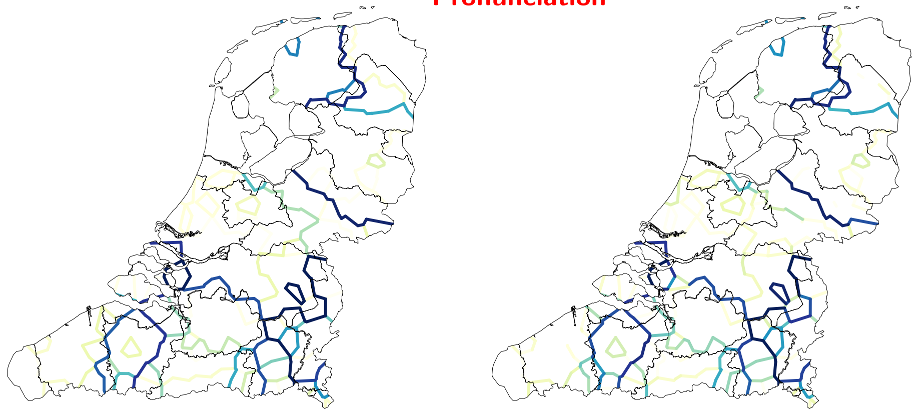



Two composite cluster maps. The left one is obtained on the basis of the aggregate of 107 distance matrices. The right one is obtained on the basis of the aggregate of 107 single word-based cluster maps.

Bootstrapping

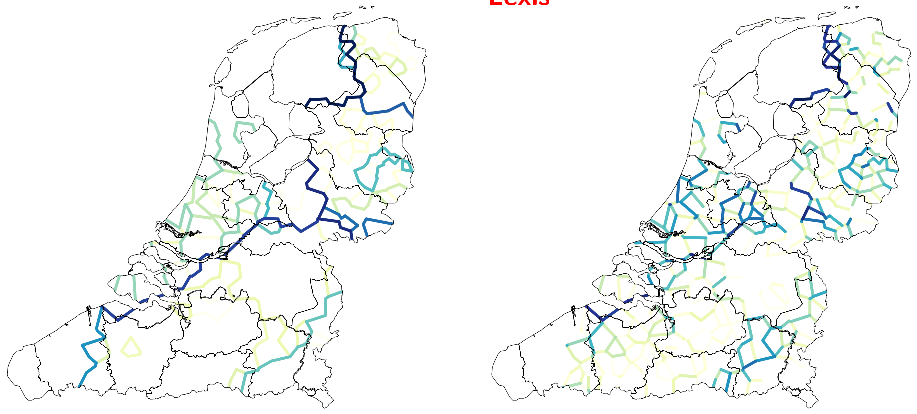
- We have one matrix per word. The distances in the matrix are based on the variation of that word (pronunciation or lexical).
- Usually if we have n words, we take the aggregate of all n corresponding matrices.
- Bootstrap matrix: select randomly m matrices from the n matrices and take the aggregate.
- In our case: $n = m$. Some matrices will not be selected, others may be selected more than once.
- Generate 100 bootstrap matrices, and generate a composite cluster map for each.
- Superimpose all 100 cluster maps.

Pronunciation



Two composite cluster maps. The left one is obtained on the basis of one distance matrix, which is based on all 125 words. The right one is based on 100 bootstrapping matrices. Each matrix is based on 125 randomly selected words.

Lexis



Two composite cluster maps. The left one is obtained on the basis of one distance matrix, which is based on all 107 words. The right one is based on 100 bootstrapping matrices. Each matrix is based on 107 randomly selected words.

Conclusions

- Pronunciation distances are more consistent than lexical distances (Cronbach's α: 0.97 versus 0.75).
- Both aggregation of cluster maps and bootstrapping shows the pronunciation distances to be more robust than the lexical distances.

Final remarks

We thank:

- Peter Kleiweg (visualization software)

More about dialectometry in Groningen and Amsterdam can be found at:

- `http://www.dialectometry.net/`