Towards a web-based platform for plotting, visualizing and enriching diachronic semantic maps

With a case study on the Greek and Egyptian temporal semantic field
Outline of the talk

- Introduction
  - What are semantic maps?
  - Le Diasema (Lexical DIAchronic SEMantic MAps)

- Building semantic maps
  - The network inference problem
  - An algorithm for automatic plotting (Regier et al. 2013)

- Two steps forwards with the automatic plotting
  - The significance of weights
  - Inferring diachronic semantic maps

- Dynamicizing a crosslinguistic semantic map of time-related meanings with Ancient Greek and Egyptian
Introduction

Semantic maps & Le Diasema
What are semantic maps?

‘A semantic map is a geometrical representation of functions (…) that are linked by connecting lines and thus constitute a network’ (Haspelmath 2003). It constitutes a ‘model of attested variation’ (Cysouw 2007).
What are semantic maps?

‘A semantic map is a geometrical representation of functions (…) that are linked by connecting lines and thus constitute a network’ (Haspelmath 2003). It constitutes a ‘model of attested variation’ (Cysouw 2007).

- Sense distinctions are based on cross-linguistic evidence and designed to have cross-linguistic validity.
- They combine the onomasiological and the semasiological perspective.
- Multifunctionality. No commitment to a particular claim about conventionalization of senses.

Figure 1. A semantic map of typical dative functions / the boundaries of English to and French à (based on Haspelmath 2003: 213, 215)
What are semantic maps?

Weighted semantic maps
What are semantic maps?

Weighted semantic maps

The weighted edges capture the frequencies of each polysemy pattern.
What are semantic maps?

Diachronic (‘dynamicized’) semantic maps
What are semantic maps?

Diachronic (‘dynamicized’) semantic maps

Figure 1. A semantic map of dative functions (Haspelmath 2003: 213)

Figure 3. A dynamicized semantic map of dative functions (Haspelmath 2003: 234)

The arrows designate directionality of change
What are semantic maps?

Lexical semantic maps

Colexification = polyfunctionality

“A given language is said to colexify two functionally distinct senses if, and only if, it can associate them with the same lexical form”
(François 2008: 170)
What are semantic maps?

Lexical semantic maps

Colexification = polyfunctionality

“A given language is said to colexify two functionally distinct senses if, and only if, it can associate them with the same lexical form” (François 2008: 170)

“A function is put on the map if there is at least one pair of languages that differ with respect to this function” (Haspelmath 2003: 217; cf. François 2008: 168-169)

Figure 4. Overlapping polysemies: Eng. *straight* vs. Fr. *droit* (François 2008: 167)
Le Diasema

Filling a gap

- Adding a diachronic dimension to semantic maps of content words
Le Diasema

Filling a gap

- Adding a diachronic dimension to semantic maps of content words

“[T]he best synchronic semantic map is a diachronic one”
(van der Auwera 2008: 43)
Le Diasema

Duration
- December 2016 – December 2018

Main research question
- How semantic maps make significant predictions about language change at the lexical level?

Funding schemes

http://web.philo.ulg.ac.be/lediasema/
To incorporate the diachronic dimension into semantic maps of content words

To extend the method so as to also include information about the cognitive and cultural factors behind the development of the various meanings

To create an online platform for automatically plotting diachronic semantic maps based on polysemy data from the languages of the world
Le Diasema

Specific objective for today

- To incorporate the diachronic dimension into semantic maps of content words

- To extend the method so as to also include information about the cognitive and cultural factors behind the development of the various meanings

- To create an online platform for automatically plotting diachronic semantic maps based on polysemy data from the languages of the world
Inferring semantic maps

A network inference problem
Inferring semantic maps

“ideally (...) it should be possible to generate semantic maps automatically on the basis of a given set of data”
(Narrog and Ito 2007: 280)
Inferring semantic maps

- Indeed, a significant limitation of the semantic map method is that it is practically impossible to handle large-scale crosslinguistic datasets manually.
- Up until recently, they were considered “not mathematically well-defined or computationally tractable, making it impossible to use with large and highly variable crosslinguistic datasets” (Croft & Poole, 2008: 1)
- This led to the development of other approaches using statistical scaling techniques, like Multi-Dimensional Scaling (MDS).

Figure 5. MDS analysis of Haspelmath’s 1997 data on indefinite pronouns (Croft & Poole 2008: 15)
Inferring semantic maps

- However, Regier, Khetarpal, and Majid showed that the semantic map inference problem is “formally identical to another problem that superficially appears unrelated: inferring a social network from outbreaks of disease in a population” (Regier et al., 2013: 91)
Inferring semantic maps

- What’s the idea?
Inferring semantic maps

- What’s the idea?
  - Let’s consider a group of social agents (represented by the nodes of a potential graph)
Inferring semantic maps

What’s the idea?

- If one observes the same disease for five of these agents (technically called a constraint on the nodes of the graph)
Inferring semantic maps

What’s the idea?

- One can postulate that all the agents met, so that all the nodes of the graph are connected (10 edges between the 5 nodes)
Inferring semantic maps

What’s the idea?
- This is neither a very likely, nor a very economic explanation
Inferring semantic maps

- What’s the idea?
  - The goal would be to have all the social agents connected with as few edges as possible.
Inferring semantic maps

What’s the idea?

- The goal would be to have all the social agents connected with as few edges as possible
- Such a **Network Inference** problem looks intuitively simple, but is computationally hard to solve
- Cf. the travelling salesman problem [TSP]: “Given a list of cities and the distance between each pair of cities, what is the shortest possible route that visits each city exactly once?”
Inferring semantic maps

What’s the idea?

- The goal would be to have all the social agents connected with as few edges as possible.
- Such a **Network Inference** problem looks intuitively simple, but is computationally hard to solve.
- Cf. the travelling salesman problem [TSP]: “Given a list of cities and the distance between each pair of cities, what is the shortest possible route that visits each city exactly once?”
- Angluin et al. (2010) concluded that the problem is indeed *computationally intractable*, but proposed an algorithm that **approximates the optimal solution** nearly as well as is theoretically possible.

**Inferring Social Networks from Outbreaks**

Dana Angluin¹,*; James Aspnes¹;**, and Lev Reyzin²;***
Inferring semantic maps

- How does it transfer to semantic maps?
Inferring semantic maps

- How does it transfer to semantic maps?
  - Nodes are meanings
Inferring semantic maps

- How does it transfer to semantic maps?
  - Nodes are meanings
  - Constraints are Polysemic items

<table>
<thead>
<tr>
<th>Meaning</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polysemic item A</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polysemic item B</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polysemic item C</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
Inferring semantic maps

- How does it transfer to semantic maps?
  - Nodes are meanings
  - Constraints are Polysemeic items
  - One connects the nodes economically based on these constraints
Inferring semantic maps

- How does it transfer to semantic maps?
  - Nodes are meanings
  - Constraints are Polysemic items
  - One connects the nodes economically based on these constraints
Inferring semantic maps

- How does it transfer to semantic maps?
  - Nodes are meanings
  - Constraints are Polysemic items
  - One connects the nodes economically based on these constraints

<table>
<thead>
<tr>
<th>Meaning</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polysemic item A</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polysemic item B</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

(but no edge needed between meaning 2 and meaning 4, as they are connected through meaning 3)
Inferring semantic maps

- How does it transfer to semantic maps?
  - Nodes are meanings
  - Constraints are Polysemic items
  - One connects the nodes economically based on these constraints

<table>
<thead>
<tr>
<th>Meaning</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polysemic item A</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polysemic item B</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polysemic item C</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Inferring semantic maps

- How does it transfer to semantic maps?

The result is a map that accounts for all the polysemy patterns, while remaining as economic as possible.

<table>
<thead>
<tr>
<th>Meaning</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polysemic item A</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polysemic item B</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polysemic item C</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Inferring semantic maps

- Having tested the algorithm on the crosslinguistic data of Haspelmath (1997) and Levinson et al. (2003), Regier et al. (2013) conclude that the approximations produced by the algorithm are of *high quality*, which means that they produce equal or better results than the manually plotted maps.
Inferring semantic maps

**INPUT**
(lexical matrix)
Inferring semantic maps

**INPUT** (lexical matrix)

<table>
<thead>
<tr>
<th>Language</th>
<th>Word</th>
<th>Specific Known</th>
<th>Specific Unknown</th>
<th>Irrealis Non-specific</th>
<th>Question</th>
<th>Conditional</th>
<th>Indirect Negation</th>
</tr>
</thead>
<tbody>
<tr>
<td>German</td>
<td>&quot;etwas&quot;</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>German</td>
<td>&quot;irgend&quot;</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>German</td>
<td>&quot;jo&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>German</td>
<td>&quot;jeder&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>German</td>
<td>&quot;n&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Dutch</td>
<td>&quot;dan ook&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Dutch</td>
<td>&quot;enig&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Dutch</td>
<td>&quot;iets&quot;</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Dutch</td>
<td>&quot;niets&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>English</td>
<td>&quot;any&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>English</td>
<td>&quot;ever&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>English</td>
<td>&quot;no&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>English</td>
<td>&quot;some&quot;</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**ALGORITHM** (python script)

```python
# MAIN LOOP
objfn = C(G,T)
while (objfn < 0):
    print("objective fn is currently", objfn,)
    max_score = 0
    # choose next edge greedily: the one that increases objfn the most
    for e in PossE:
        # temporarily add e to graph G
        G.add_edge(e)
        score = C(G,T) - objfn
        G.remove_edge(e)
        if (score > max_score):
            max_score = score
            max_edge = e
```


Inferring semantic maps

**INPUT**
(lexical matrix)

<table>
<thead>
<tr>
<th>Language</th>
<th>Word</th>
<th>Specific Known</th>
<th>Specific Unknown</th>
<th>Irrealis Non-specific</th>
<th>Question</th>
<th>Conditional</th>
<th>Indirect Negation</th>
</tr>
</thead>
<tbody>
<tr>
<td>German</td>
<td>&quot;etwas&quot;</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>German</td>
<td>&quot;irgend&quot;</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>German</td>
<td>&quot;jo&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>German</td>
<td>&quot;jeder&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>German</td>
<td>&quot;n.&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Dutch</td>
<td>&quot;dan ook&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Dutch</td>
<td>&quot;enig&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Dutch</td>
<td>&quot;iets&quot;</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Dutch</td>
<td>&quot;niets&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>English</td>
<td>&quot;any&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>English</td>
<td>&quot;ever&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>English</td>
<td>&quot;no&quot;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>English</td>
<td>&quot;some&quot;</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**ALGORITHM**
(python script)

```python
# MAIN LOOP
objfn = C(G,T)
while (objfn < 0):
    print ("objective fn is currently", objfn,)
    max_score = 0
    # choose next edge greedily: the one that increases objfn the most
    for e in PossE:
        # temporarily add e to graph G
        G.add_edge("e")
        score = C(G,u,T) - objfn
        G.remove_edge("e")
        if (score > max_score):
            max_score = score
            max_edge = e
```

**RESULT**
(semantic map)

```
specific known    specific unknown    irrealis non-specific
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>question</td>
<td>indirect negation</td>
<td>direct negation</td>
</tr>
<tr>
<td>conditional</td>
<td>comparative</td>
<td>free choice</td>
</tr>
</tbody>
</table>
```
Automatic plotting: Two steps forwards
Weighted and diachronic semantic maps
Automatic plotting: Two steps forward

- Weighted semantic maps are much more informative than regular semantic maps, because they visually provide information about the frequency of polysemy patterns.

- Diachronic semantic maps are much more informative than regular semantic maps, because they visually provide information about possible pathways of change.
Automatic plotting: Two steps forward

- Weighted semantic maps are much more informative than regular semantic maps, because they visually provide information about the frequency of polysemy patterns.

- Diachronic semantic maps are much more informative than regular semantic maps, because they visually provide information about possible pathways of change.

Can weighted and diachronic semantic map be plotted automatically?
Automatic plotting: Two steps forward

- Weighted semantic maps are much more informative than regular semantic maps, because they visually provide information about the frequency of polysemy patterns.

- Diachronic semantic maps are much more informative than regular semantic maps, because they visually provide information about possible pathways of change.

Can weighted and diachronic semantic map be plotted automatically?

YES!
Automatic plotting: Two steps forward

- Weighted semantic maps are much more informative than regular semantic maps, because they visually provide information about the frequency of polysemy patterns.

- Diachronic semantic maps are much more informative than regular semantic maps, because they visually provide information about possible pathways of change.

Can weighted and diachronic semantic map be plotted automatically? **YES!**

How?
Automatic plotting: Two steps forward

Weighted semantic maps
Automatic plotting: Two steps forward

Weighted semantic maps

- Generate the map with a modified version of the algorithm of Regier et al. (2013)
Automatic plotting: Two steps forward

Weighted semantic maps

- Generate the map with a modified version of the algorithm of Regier et al. (2013)
  - **PRINCIPLE:** for each edge that is being added between two meanings of the map by the algorithm, check in the lexical matrix how many times this specific polysemy pattern is attested, and increase the weight of the edge accordingly

```python
edgeWeight = 0
for sns in sensesTupleList:
    if (max_edge[0] in sns) and (max_edge[1] in sns):
        edgeWeight += 1
G.add_edge(*max_edge, weight=edgeWeight)
```
Automatic plotting: Two steps forward

Weighted semantic maps

- Generate the map with a modified version of the algorithm of Regier et al. (2013)
  - **PRINCIPLE:** for each edge that is being added between two meanings of the map by the algorithm, check in the lexical matrix how many times this specific polysemy pattern is attested, and increase the weight of the edge accordingly

```python
edgeWeight = 0
for sns in sensesTupleList:
    if (max_edge[0] in sns) and (max_edge[1] in sns):
        edgeWeight += 1
G.add_edge(*max_edge,weight=edgeWeight)
```

- Based on the data of Haspelmath (1997), kindly provided by the author, the result between a non-weighted and a weighted semantic map are markedly different
Automatic plotting: Two steps forward

Weighted semantic maps

Automatically plotted semantic maps: non-weighted vs. weighted (data from Haspelmath 1997)

The graph is visualized in Gephi® with the Force Atlas algorithm
Automatic plotting: Two steps forward

Diachronic semantic maps
Automatic plotting: Two steps forward

Diachronic semantic maps

- Expand the lexical matrix so as to include information about diachrony

<table>
<thead>
<tr>
<th>Source of constraint</th>
<th>Constraint name</th>
<th>Constraint Time</th>
<th>Sense_1</th>
<th>Sense_2</th>
<th>Sense_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language_1</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Language_1</td>
<td>Word_1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Language_3</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_3</td>
<td>Word_2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Automatic plotting: Two steps forward

Diachronic semantic maps

- Expand the lexical matrix so as to include information about diachrony

<table>
<thead>
<tr>
<th>Source of constraint</th>
<th>Constraint name</th>
<th>Constraint Time</th>
<th>Sense_1</th>
<th>Sense_2</th>
<th>Sense_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language_1</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Language_1</td>
<td>Word_1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Language_3</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_3</td>
<td>Word_2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

The diachronic stages are indexed by numbers: 0, 1, 2, etc.
Automatic plotting: Two steps forward

Diachronic semantic maps

- Expand the lexical matrix so as to include information about diachrony

<table>
<thead>
<tr>
<th>Source of constraint</th>
<th>Constraint name</th>
<th>Constraint Time</th>
<th>Sense_1</th>
<th>Sense_2</th>
<th>Sense_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language_1</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Language_1</td>
<td>Word_1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Language_3</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_3</td>
<td>Word_2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

The meaning of a word can change from one stage to another (e.g., Word_2 of Language_2 expresses the meaning Wood during stage 0 and Wood & Forest during stage 1)
Automatic plotting: Two steps forward

Diachronic semantic maps

- Expand the lexical matrix so as to include information about diachrony
- Generate the graph with the algorithm of Regier et al. (2013)
Automatic plotting: Two steps forward

Diachronic semantic maps

- Expand the lexical matrix so as to include information about diachrony

- Generate the graph with the algorithm of Regier et al. (2013)

- Enrich the graph with oriented edges (where relevant)
  - **PRINCIPLE:** for each edge in the graph, if the meaning of node A is attested for one diachronic stage, while the meaning of node B is not, check in the lexical matrix if there is a later diachronic stage of the same language for which this specific word has both meaning A and B (or just meaning B). If this is the case, we can infer a meaning extension from A to B.
**Automatic plotting: Two steps forward**

**Diachronic semantic maps**

**INPUT** (diachronic lexical matrix)

<table>
<thead>
<tr>
<th>Source of constraint</th>
<th>Constraint name</th>
<th>Constraint Time</th>
<th>Sense_1</th>
<th>Sense_2</th>
<th>Sense_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language_1</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Language_1</td>
<td>Word_1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Language_3</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_3</td>
<td>Word_2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Automatic plotting: Two steps forward

Diachronic semantic maps

**INPUT**
(diachronic lexical matrix)

**Algorithm**
(python script for inferring oriented edges)

<table>
<thead>
<tr>
<th>Source of constraint</th>
<th>Constraint name</th>
<th>Constraint Time</th>
<th>Sense_1</th>
<th>Sense_2</th>
<th>Sense_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language_1</td>
<td>Word_1</td>
<td>0</td>
<td>Tree</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_1</td>
<td>Word_1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_3</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_3</td>
<td>Word_2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

```python
G = nx.Graph()  # create an empty graph
G.add_node('A')
G.add_node('B')
G.add_edge('A', 'B', weight=1)
G.add_edge('B', 'A', weight=2)
G.add_edge('A', 'C', weight=3)
G.add_edge('C', 'A', weight=4)
G.add_edge('B', 'C', weight=5)
G.add_edge('C', 'B', weight=6)
G.add_edge('D', 'A', weight=7)
G.add_edge('A', 'D', weight=8)
G.add_edge('D', 'B', weight=9)
G.add_edge('B', 'D', weight=10)
G.add_edge('D', 'C', weight=11)
G.add_edge('C', 'D', weight=12)

H = G.to_directed()  # convert the graph 'G' into a directed Graph 'H' in order to explore all the possibilities as regards the relationship between the nodes (i.e., both A -> B and B -> A for all the connected nodes, crucial not only A -> B)
nx.set_edge_attributes(H, 'type', 'undirected')  # set the default value to "undirected"
```

```python
for u,v,e in H.edges(data=True):  # loop over all the edges in the DiGraph 'H'
    for t in T_Full:  # look at the metadata and senses for one line in T
        if t.count(u) == 1 and t.count(v) == 0:  # if the meaning of node 'u' in the
            if t.count(u) == 1 and t.count(v) == 0:  # while the meaning of node 'v' is
```

```python
```
Automatic plotting: Two steps forward
Diachronic semantic maps

**INPUT**
(diachronic lexical matrix)

<table>
<thead>
<tr>
<th>Source of constraint</th>
<th>Constraint name</th>
<th>Constraint Time</th>
<th>Sense_1</th>
<th>Sense_2</th>
<th>Sense_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language_1</td>
<td>Word_1</td>
<td>0</td>
<td>Tree</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_1</td>
<td>Word_1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Language_2</td>
<td>Word_2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Language_3</td>
<td>Word_1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Language_3</td>
<td>Word_2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**ALGORITHM**
(python script for inferring oriented edges)

```python
H = G.to_directed()  # convert the graph 'G' into a directed Graph 'H' in order to explore all the possibilities as regards the relationship between the nodes, i.e. both A -> B and B -> A for all the connected nodes, crucial not only A -> B
nx.set_edge_attributes(H, 'type', 'undirected')  # set the default value to "undirected" for all the edges
for u, v, e in H.edges(data=True):
    for t in T_Full:
        if t.count(u) == 1 and t.count(v) == 0:
            # if the meaning of node 'u' in the # while the meaning of node 'v' is...
```

**RESULT**
(dynamic semantic map)
The semantic extension of time-related lexemes

Inferring a semantic map based on cross-linguistic colexification patterns and enriching it with diachronic data
The semantic extension of time-related lexemes

Plotting a synchronic semantic map

- For the purpose of universality and stability, we chose the entries for time-related concepts in the Swadesh 200-word list (Swadesh 1952: 456-457)
  - DAY/DAYTIME
  - NIGHT
  - YEAR
The semantic extension of time-related lexemes

Plotting a synchronic semantic map

- We identified in the database of Crosslinguistic Colexifications (CLICs; http://clics.lingpy.org/main.php; List et al. 2014) the main polysemy patterns attested for these three meanings (subgraph approach) [16 meanings]
  - **DAY/DAYTIME**: CLOCK/TIMEPIECE, HOUR, SEASON, SUN, TIME, WEATHER
  - **NIGHT**: DARK (in color), DARKNESS, BLACK, OBSCURE
  - **YEAR**: AGE, SPRING, SUMMER
The semantic extension of time-related lexemes
Plotting a synchronic semantic map

- All the colexification patterns attested for these 16 meanings were gathered in the CLICs source files (http://clics.lingpy.org/download.php), ending up with 381 colexification patterns.
The semantic extension of time-related lexemes

Plotting a synchronic semantic map

- All the colexification patterns attested for these 16 meanings were gathered in the CLICs source files (http://clics.lingpy.org/download.php), ending up with 381 colexification patterns
- These synchronic polysemy patterns were converted into a lexical matrix

Python script α

Lexical matrix

1 when a meaning is attested for one form
The semantic extension of time-related lexemes

Plotting a synchronic semantic map

- All the colexification patterns attested for these 16 meanings were gathered in the CLICs source files (http://clics.lingpy.org/download.php), ending up with 381 colexification patterns.

- These synchronic polysemy patterns were converted into a lexical matrix.

- From this lexical matrix, we inferred a weighted semantic map based on the adapted version of the algorithm suggested by Regier et al. (2013).
Full semantic map for time-related senses, visualized with modularity analysis* (Blondel et al. 2008) in Gephi

* A method to extract the community structure of large networks. Here, the different colors point to modules (also called clusters or communities) with dense connections between the nodes within the network.
The semantic extension of time-related lexemes

Plotting a synchronic semantic map

- All the colexifications patterns attested for these 16 meanings were gathered in the CLICs source files (http://clics.lingpy.org/download.php), ending up with 381 colexifications patterns.

- These synchronic polysemy patterns were converted into a lexical matrix.

- From this lexical matrix, we inferred a weighted semantic map, based on an adapted version of the algorithm by Regier et al. (2013).

- The weighted edges allow us to get rid of poorly attested patterns of polysemy (keeping only those attested in 2+ languages).
Semantic maps of time-related senses (colexification patterns attested in 2+ languages)

Two connected sub-networks
- NIGHT/DARKNESS/DARK
- DAY/TIME/AGE/YEAR
Semantic maps of time-related senses (colexification patterns attested in 2+ languages)

Two connected sub-networks
- NIGHT/DARKNESS/DARK
- DAY/TIME/AGE/YEAR
The semantic extension of time-related lexemes
Towards a dynamicized semantic map
The semantic extension of time-related lexemes
Towards a dynamicized semantic map

- In order to investigate directionality of change, 13 meanings that are connected on this map in at least 8 different languages were kept as a basis for diachronic investigation.
The semantic extension of time-related lexemes
Towards a dynamicized semantic map

**Diachronic data**

- **Ancient Greek** (8\(^{th}\) – 4\(^{th}\) c. BC; in a few cases till 1\(^{st}\) c. BC)
  - Cunliffe (*A lexicon of the Homeric Dialect*), LSJ

- **Ancient Egyptian** (26th c. BC – 10th c. AD)
  - Thesaurus Linguae Aegyptiae ([http://aaew.bbaw.de/TLA/](http://aaew.bbaw.de/TLA/))
  - The Ramses corpus ([http://ramses.ulg.ac.be](http://ramses.ulg.ac.be)),
  - Lexical resources (Coptic etymological dictionaries)
The semantic extension of time-related lexemes

Towards a dynamicized semantic map

- The diachronic material allows us to add diachronic information (graphically, oriented edges) between frequent colexification patterns

- TIME?
# Ancient Greek

### hóra ‘season/time/moment’

1. **hóssá** te **phúlla** kai **ánthea**
   - REL.NOM.PL.N  PTC  leave:ACC.PL.N  CONJ  flower:ACC.PL.N
   - become:PRS.3SG  season:DAT.SG.F

   `as are the leaves and the flowers in their season’ (Homer, *Iliad* 2.468)

2. **óphra** Poseidáõi kai állois athanátoisin
   - CONJ  Poseidon:DAT.SG.M  CONJ  other:DAT.PL  immortal:DAT.PL
   - speísantes koitoio medómetha:
     - pour.libation:PART.AOR.NOM.PL.M  bed:GEN.SG.M  think.of:PRS.1PL.SUBJ.M/P

   `that when we have poured libations to Poseidon and the other immortals, we may bethink us of sleep; for it is the time thereto’ (Homer, *Odyssey* 3.333-334)
Ancient Greek

*hóra* ‘season/time/moment’ ⇒ ‘hour’

(3) anastâs  dè  pròi  pseustheîs
raise.up:PTCP.AOR.NOM.SG.M  PTC  early  deceive:PTCP.AOR.PASS.NOM.SG.M
tês  hóras  badízein
ART.GEN.SG.F  time:GEN.SG.F  walk:PRS.INF

‘He arose early, mistaking the *time/hour*, and started off on his walk’
(Andocides, *On the Mysteries* 1.38)

(4) oukhì  dōdeka  hôraiè  eisin  tês  hêméras;
NEG  twelve  hour:NOM.PL.F  be.PRS.3PL  ART.GEN.SG.F  day:GEN.SG.F

‘Aren’t there twelve hours of daylight?’ (New Testament, John 11.9.2)
Ancient Greek

**Metonymy**: due to the correlation between the canonical time periods and the time these take to unfold
The semantic extension of time-related lexemes

Towards a dynamicized semantic map

- The diachronic material allows us to add diachronic information (graphically, oriented edges) between frequent colexification patterns

  - **TIME** $\Rightarrow$ **HOUR**
The semantic extension of time-related lexemes

Towards a dynamicized semantic map

- The diachronic material allows us to add diachronic information (graphically, oriented edges) between frequent colexification patterns.

- The material allows us to add new polysemy patterns, and to provide a diachronic account.

- SUMMER?
(Ancient) culture-specific colexification patterns

Summer?

<table>
<thead>
<tr>
<th>Concept</th>
<th>IDS-Key</th>
<th>Occurrences</th>
<th>Families</th>
<th>Languages</th>
<th>Network</th>
<th>Forms</th>
</tr>
</thead>
<tbody>
<tr>
<td>year</td>
<td>14.73</td>
<td>233</td>
<td>10</td>
<td>16</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>age</td>
<td>14.12</td>
<td>257</td>
<td>2</td>
<td>3</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>bow</td>
<td>20.24</td>
<td>231</td>
<td>2</td>
<td>2</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>spring</td>
<td>14.75</td>
<td>174</td>
<td>2</td>
<td>3</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>autumn</td>
<td>14.77</td>
<td>167</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>cave</td>
<td>1.28</td>
<td>256</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>cousin</td>
<td>2.55</td>
<td>346</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>hang up</td>
<td>9.341</td>
<td>280</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>hot</td>
<td>15.85</td>
<td>303</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>put</td>
<td>12.12</td>
<td>306</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>rain (noun)</td>
<td>1.75</td>
<td>257</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>reach, arrive</td>
<td>10.55</td>
<td>329</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>rise</td>
<td>10.21</td>
<td>334</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>season</td>
<td>14.78</td>
<td>193</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>sun</td>
<td>1.52</td>
<td>245</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>wall</td>
<td>7.27</td>
<td>239</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
<tr>
<td>wine</td>
<td>5.92</td>
<td>162</td>
<td>1</td>
<td>1</td>
<td>COM</td>
<td>SUB</td>
</tr>
</tbody>
</table>
Ancient Greek

tréros ‘summer’ ⇒ ‘harvest’

(5) autàr epèn élthēisi théros tethaluiá
PTC when come:AOR.SUBJ.3SG summer:NOM.SG.M thrive:PART.PERF.NOM.SG.F
't' opórē
PTC autumn:NOM.SG.F

‘But when summer comes and rich autumn’ (Homer, Odyssey 11.192)

(6) kāit' anèr édoksen eînai, tallótrion
ADV man:NOM.SG.M seem:AOR.3SG be.INF another:GEN.SG

amôn théros
reap.corn:PTCP.PRS.NOM.SG.M summer:ACC.SG.N

‘he has only made himself a name by reaping another’s harvest’ (Aristophanes, Knights 392)
Ancient Egyptian

Smw ‘summer’ ⇒ Smw ‘harvest’

Old Kingdom

Middle Kingdom
The semantic extension of time-related lexemes
Towards a dynamicized semantic map

- The diachronic material allows us to add diachronic information (graphically, oriented edges) between frequent colexification patterns

- The material allows us to add new polysemy patterns, and to provide a diachronic account

- SUMMER

  ![Diagram of semantic relationships with nodes labeled 'SUMMER', 'HARVEST', 'spring', 'bow', 'year', 'winter', 'water', 'leaf', 'yesterday']
The semantic extension of time-related lexemes
Towards a dynamicized semantic map

- The diachronic material allows us to add diachronic information (graphically, oriented edges) between frequent colexification patterns

- The material allows us to add new polysemy patterns, and to provide a diachronic account

- The material allows us to highlight unexpected pathways of change: from temporal proximity to spatial proximity
Language-specific colexification patterns

(7)  
mark  Hm-f  nswt-bity  nb-kAw-ra  
in  time  Majesty-3SG.M  King of U. and L. Egypt  Nebkaure  

‘(Now, the peasant spoke these word) **during the time** of his Majesty, the King of Upper and Lower Egypt, Nebkaure (the justified)’ (= Parkinson 1991: 19)

(8)  
sbty  Dr  mark  mSa-f  (= KRI II, 6,8)  
rampart  strong  in  proximity  army-3SG.M  

(speaking of the King who is)  

‘A strong rampart around his army, (their shied in the day of fighting)’
Language-specific colexification patterns

(Stage I)
- 'temporal proximity'
- rk

(Stage II)
- 'spatial proximity'
- rk

+
Language-specific colexification patterns

- Counterexample to the TIME IS SPACE metaphor?
  - Cross-linguistically Time to Space transfers are extremely rare (cf. French depuis; Haspelmath 1997)
Language-specific colexification patterns

(9) m hAw nb tA-wj nb-pH.tj-ra
in prox-time lord land-DU Nebphtire
(And then I became a soldier (...),)
‘during the time of the lord of the Two Lands, Nebpehtire (justified, when I was a young man, not having a wife yet)’ (= Urk. IV, 2,13)

(10) m hAw nh.t
in prox-space Sycamore
‘(I crossed the place called The Two Truths,) in the vicinity of The Sycamore” (and I landed at The Island of Snefru)” (= Koch 1990: 14)
Language-specific colexification patterns

(Stage I)
Language-specific colexification patterns

(Stage I)
Language-specific colexification patterns

(Stage I)

- 'temporal proximity'
- 'spatial proximity'

(Stage II)

hAw

rk

rk

'temporal proximity'

'spatial proximity'
Conclusions

The web-based platform
Conclusions

- In this talk, we have shown that weighted diachronic semantic maps can be automatically plotted based on lexical matrix of crosslinguistic colexification patterns.

- Furthermore, we demonstrated that language-specific studies reveal interesting polysemy patterns, with unexpected pathways of change.
Conclusions

- In this talk, we have shown that weighted diachronic semantic maps can be automatically plotted based on lexical matrix of crosslinguistic colexification patterns.

- Furthermore, we demonstrated that language-specific studies reveal interesting polysemy patterns, with unexpected pathways of change.

- One can expect the linguists to build solid polysemy matrices.

- BUT, one cannot expect them to master the technicalities involved for plotting semantic maps automatically and exploring the result.
Conclusions

- In this talk, we have shown that weighted diachronic semantic maps can be automatically plotted based on lexical matrix of crosslinguistic colexification patterns.

- Furthermore, we demonstrated that language-specific studies reveal interesting polysemy patterns, with unexpected pathways of change.

- One can expect the linguists to build solid polysemy matrices.

- BUT, one cannot expect them to master the technicalities involved for plotting semantic maps automatically and exploring the result.

A web-based platform can easily do the work.
Upload the lexical matrix (.xls file)
Upload the lexical matrix (.xls file)
Lexical matrix uploaded ✓

Generate the map

- Weighted ✓
- Diachronic □
Conclusions
Selected references


Thanks!

athanasios.georgakopoulos@ulg.ac.be
s.polis@ulg.ac.be

http://web.philo.ulg.ac.be/lediasema/