AGENT-BASED MODELLING IN LINGUISTICS: WHAT, WHY AND HOW

Dirk Pijpops, Zurich, November 18, 2019

Structure

- What (brief)
- Why (brief)
- How
  - In general
  - Example: lectal contamination
    - What's good
    - What's bad
  - Recap & best practices

WHAT

Agent-based modeling is therefore an exceptionally ambitious undertaking, and the groups working in the field are often multidisciplinary All-Star teams composed of some of the best and brightest individuals in their respective professions.

- Nate Silver

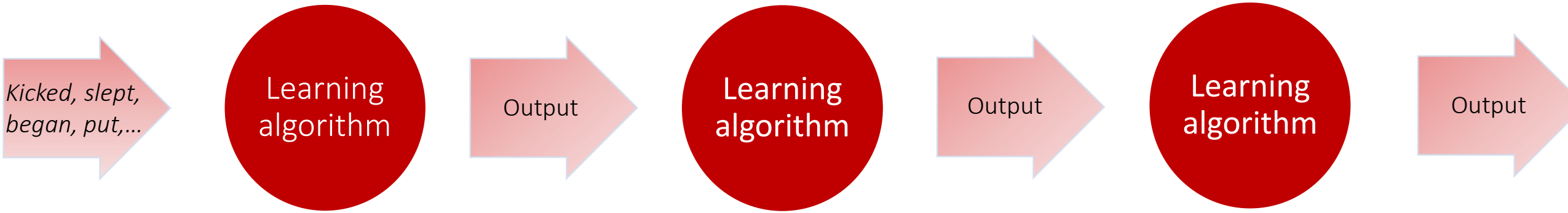
The Signal and the Noise, the Art and Science of Prediction, p. 228

WHAT

- Other types of computer simulations in science:
  - Mathematical/physical simulations (e.g. Liska et al. 2019)
  - Chemical/molecular simulations (e.g. Valverde 2001)
  - Iterated learning simulations (e.g. Rumelhart & McClelland 1986, Pinker & Prince 1988)

What

- Iterated learning simulations

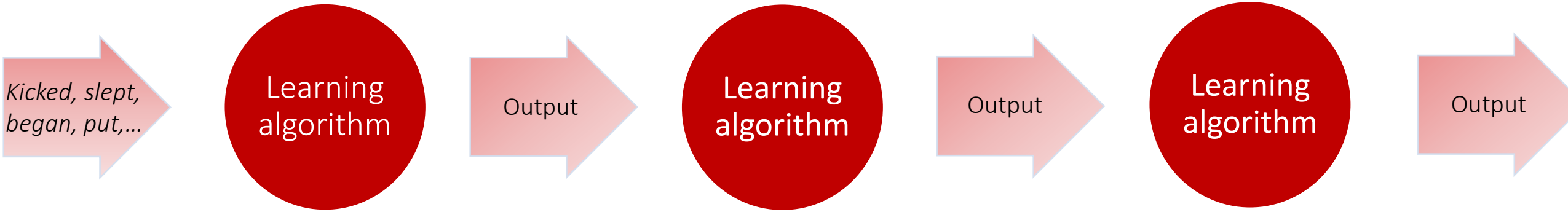


— Investigate the outputs

— Goal: What does such an algorithm minimally require in order to yield realistic results, e.g. make realistic mistakes (U-shaped learning), cause long-term tendencies (weakening),…?

What

- Iterated learning simulations



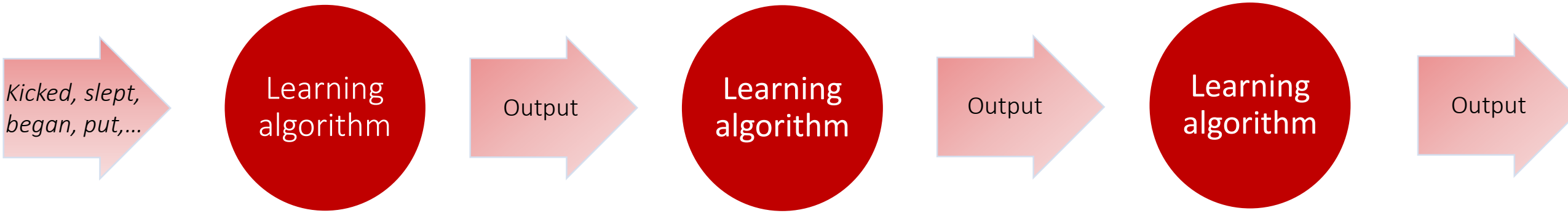
— (Classic) generative perspective: only locus of language change is language acquisition

— Assumption that population is uniform: no need to model population

— Assumption that language use (after critical period) has no affect on language system: no need to model language use

What

- Iterated learning simulations

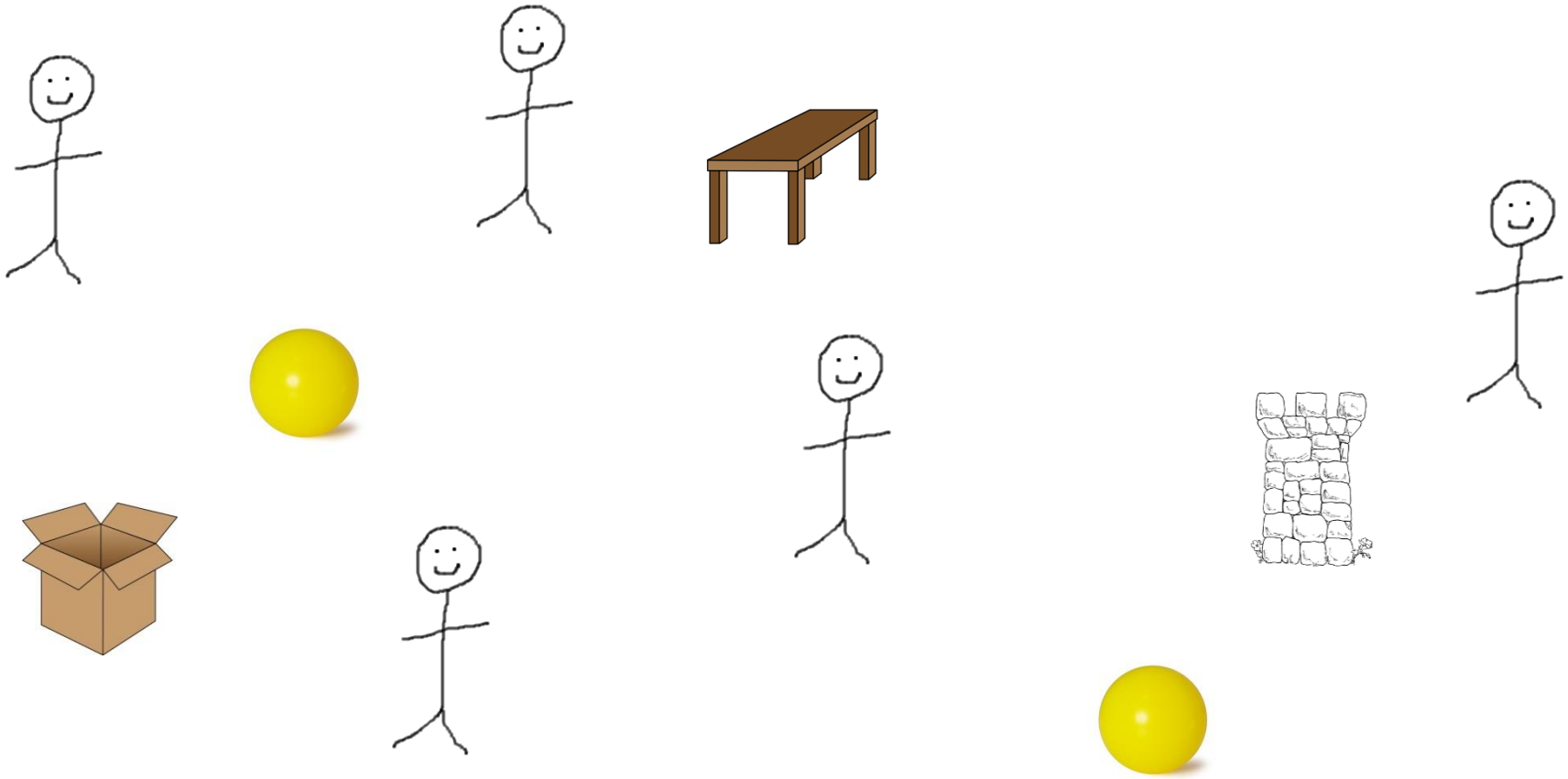


— Connectionism (neural network, deep learning) vs. symbolic rules (generative approaches)

— (Rumelhart and McClelland 1986; Pinker and Prince 1988; Macwhinney and Leinbach 1991; Plunkett and Marchman 1991; Ling and Marinov 1993; Plunkett and Marchman 1993; Plunkett and Juola 1999; van Noord and Spenader 2015)

What

- Agent-based simulations



What

- Agent-based simulations
  - Put multiple 'agents' in a world
  - Have them perform localized tasks/give them localized goals
  - They may collaborate/compete in these simple tasks (selfish)
  - Behavior of one agent affects behavior of the others
  - Agents are only aware of their direct environment
  - Only researcher has a god's eye view

What

- Agent-based simulations are used to model Complex Adaptive Systems, e.g.
  - Termite behavior (Guerreiro et al. 2013)
  - Traffic (Bazghandi 2012)
  - The development of the internet (Dhamdher & Dovrolis 2009)
  - The stock market (Brock, Hommes & Wagener 2009)
  - ….

What

- Usage-based linguists have argued that language is a complex adaptive system (Steels 2000, Beckner et al. 2009)
  - Complex: has emergent traits
  - Adaptive: changes in response to usage/environment
  - System: exhibits systematicity

What

- Usage-based linguists are turning to agent-based models to study how:
  - How language has evolved: Jaeger et al. (2009), Beuls & Steels (2013) (new question)
  - How language changes: Landsbergen et al. (2010), Lestrade (2015), Bloem et al. (2015), Pijpops et al. (2015)
  - Language evolution = language change: same mechanisms at work, e.g. grammaticalization, exaptation, reanalysis,…

What

- Usage-based perspectives on language evolution/change

  — Language as behavior, instead of pure cognition

  — Language acquisition AND usage (after critical period) may affect language

  — Population is not uniform

  $\Longrightarrow$ Agent-based models of language

Why

- Prove that an effect will emerge under certain assumptions
  - Verbal weak inflection (Ockham's razor)
  - Lectal contamination
- Formulate predictions that may then be tested empirically
  - Lectal contamination

Why

- NOT to try to build ultimately realistic simulations: doomed and useless

- NOT to study the simulations for the sake of the simulations: fact-free science (de Boer 2012)

How

1. Define an effect to be generated OR define something to predict

2. Conceptual design: hardest step
   - Which assumptions: be minimal
   - How will these assumptions be implemented: be as precise as possible
   - How will the model be evaluated

3. Implementation: easiest step

How

4. Evaluation
   - Test extreme parameter settings one by one (getting a feel for the model)
   - Get weird results: probably a bug, return to step 3
   - Narrow down the parameter range in which something is happening
   - Evaluate

5. Reverse engineer: remove assumptions one by one

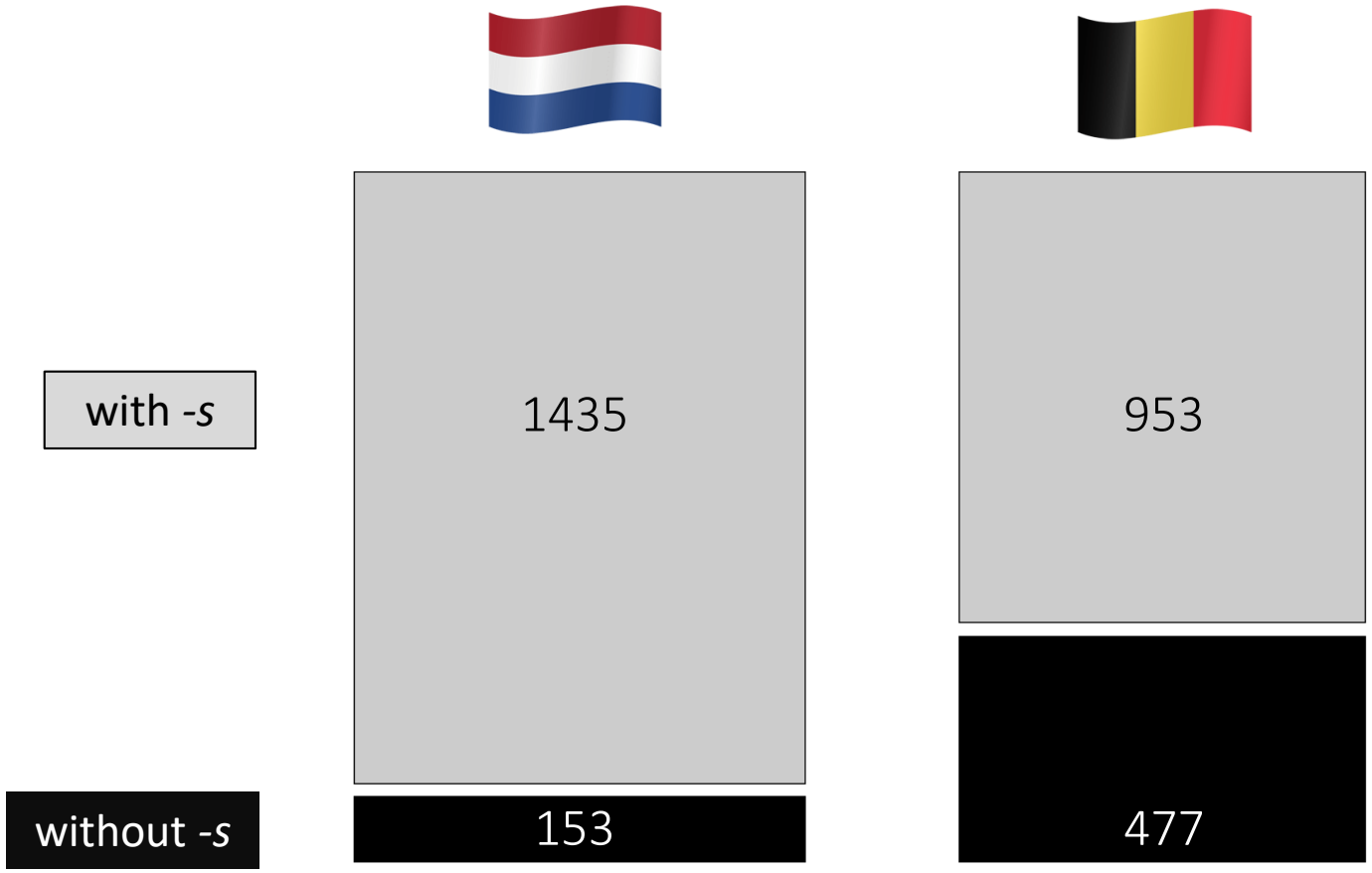
6. Write paper

EXAMPLE: LECTAL CONTAMINATION

- Case study: Dutch partitive genitive [Indefinite pronoun + adjective (-s)]$_{NP}$
  - *Iets leuk* *iets leuks* 'something fun'
  - *Iets erg speciaal* *iets erg speciaals* 'something very special'
  - *Veel bijzonder* *veel bijzonders* 'a lot of special things'
  - …

- What determines –s omission?

EXAMPLE: LECTAL CONTAMINATION



Other factors:

- Type of adjective
- Register
- Pronoun
- Frequency

$p < 0.001$, Cramér's V = 0.29

EXAMPLE: LECTAL CONTAMINATION

The Netherlands:

variant with *-s*

Belgium:

variant without *-s*

→

In general:

Typically Netherlandic strings:

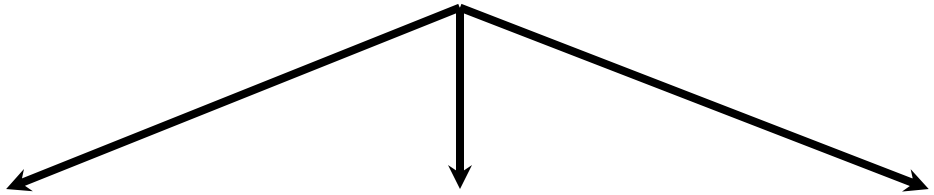
variant with *-s*

Typically Belgian strings:

variant without *-s*

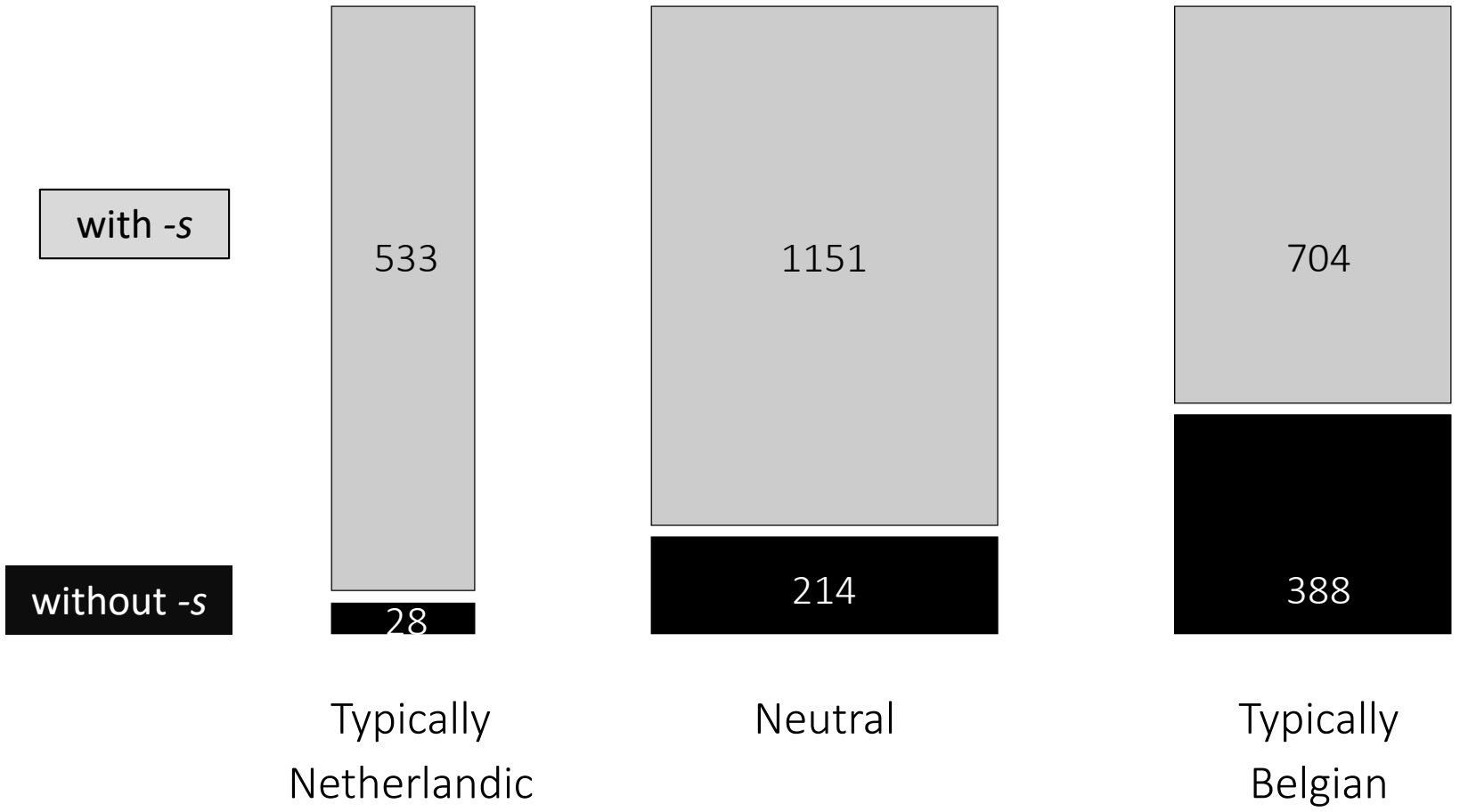
EXAMPLE: LECTAL CONTAMINATION

143 phrase types



Typically Netherlandic	Neutral	Typically Belgian
wat boeiend(s)	*weinig concreet(s)*	*iets interessant(s)*
iets bijzonder(s)	*iets zinnig(s)*	*niets speciaal(s)*
wat leuk(s)	*iets spannend(s)*	*iets deftig(s)*
iets leuk(s)	*niets erg(s)*	*iets raar(s)*
…	…	…

EXAMPLE: LECTAL CONTAMINATION



$p < 0.001$, kendall's $\tau = 0.27$

EXAMPLE: LECTAL CONTAMINATION

The Netherlands:

variant with *-s*

Belgium:

variant without *-s*

→

In general:

Typically Netherlandic strings:

variant with *-s*

Typically Belgian strings:

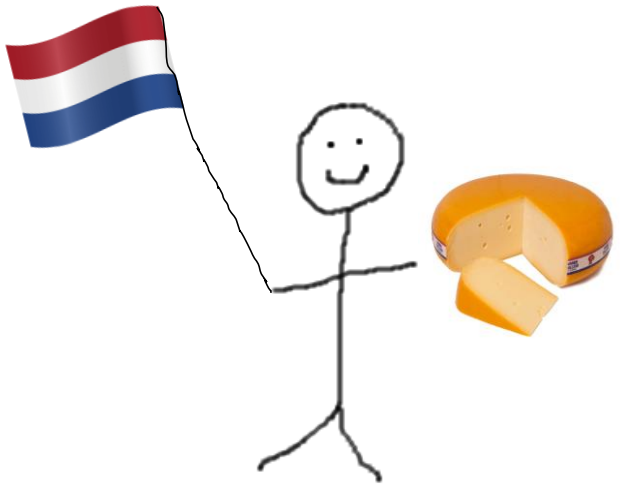
variant without *-s*

→

Strings are stored as ready-mades:

Lexical biases become entrenched

EXAMPLE: LECTAL CONTAMINATION



Hears:

Typically Belgian *iets speciaal(s)* without *-s* > Typically Netherlandic *iets bijzonder(s)* without *-s*

Produces:

P(without *-s* | *iets speciaal(s)*) > P(without *-s* | *iets bijzonder(s)*)

EXAMPLE: LECTAL CONTAMINATION

The Netherlands:

variant with *-s*

Belgium:

variant without *-s*

→

In general:

Typically Netherlandic strings:

variant with *-s*

Typically Belgian strings:

variant without *-s*

→

Strings are stored as ready-mades:

Lexical biases become entrenched

→

Even within a single lect:

Typically Netherlandic strings:

variant with *-s*

Typically Belgian strings:

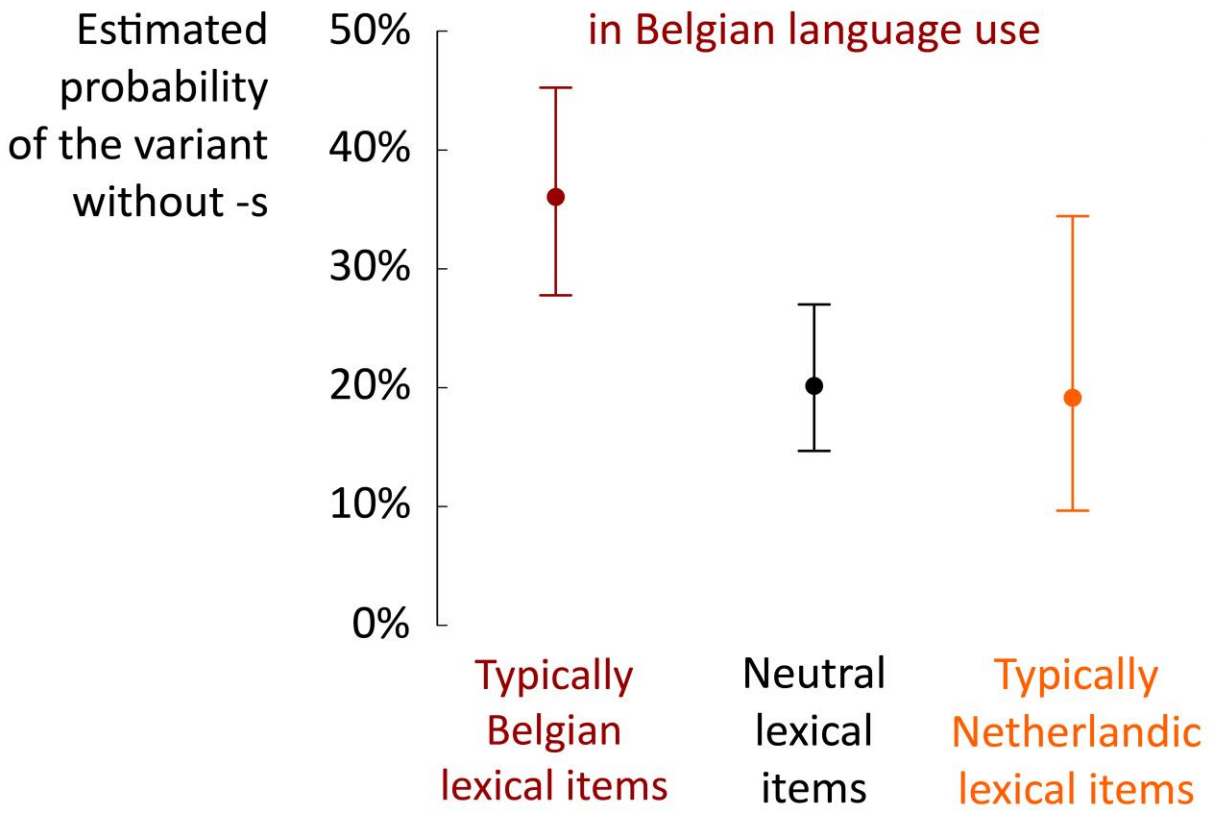
variant without *-s*

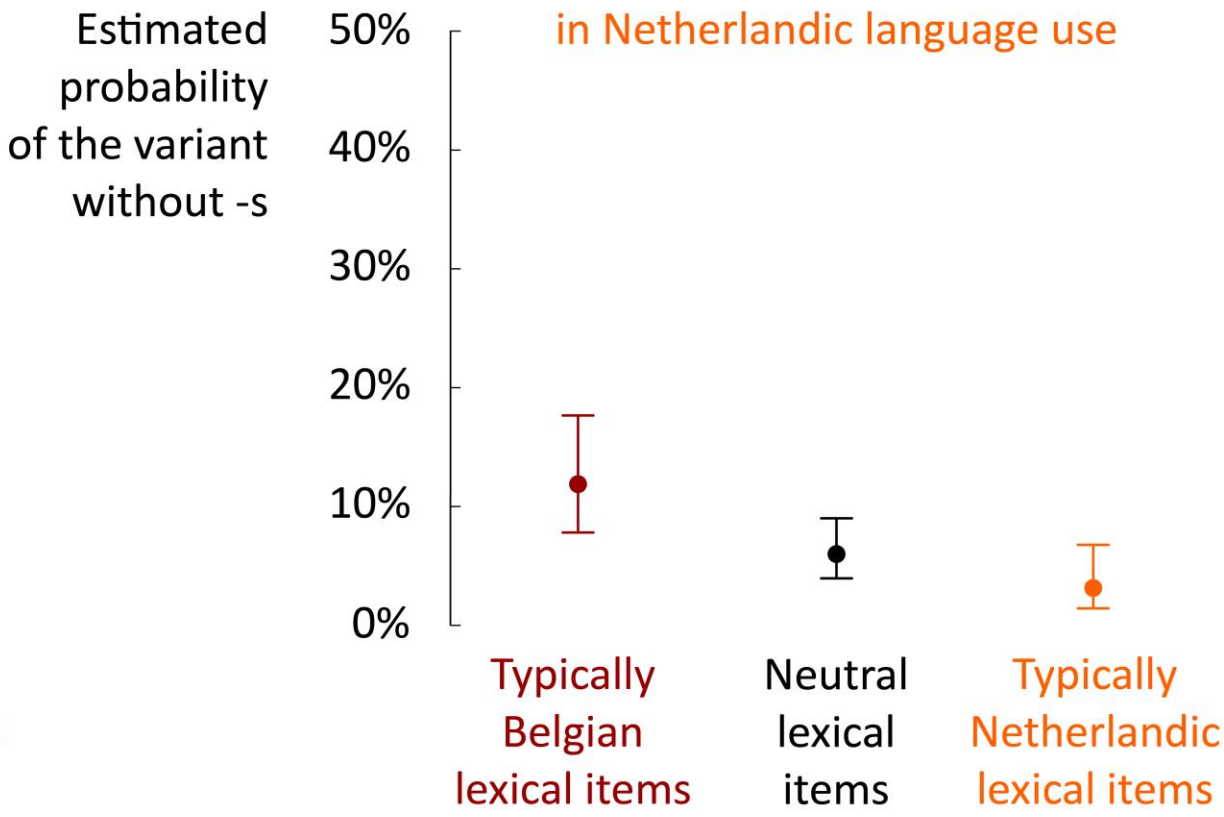
EXAMPLE: LECTAL CONTAMINATION

$$-s\ absence \sim AdjectiveType + Country + Register + Pronoun + Frequency + 1|Phrase$$

$$+ LectalProfile + Country{:}\ LectalProfile$$

EXAMPLE: LECTAL CONTAMINATION





EXAMPLE: LECTAL CONTAMINATION

- Conclusion: Belgian regiolect affected by Netherlandic use, and… Netherlandic regiolect affected by Belgian language use

- Result: Netherlandic **and** Belgian researchers did not believe us

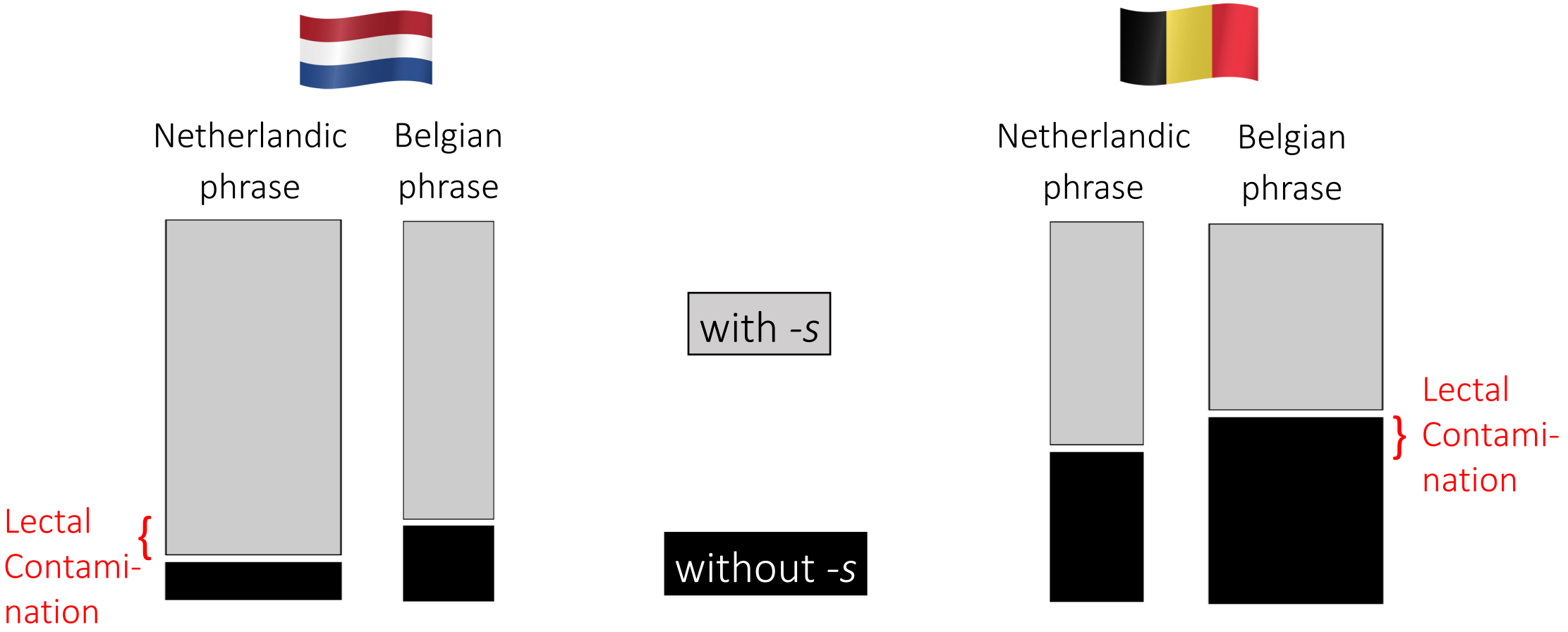
EXAMPLE: LECTAL CONTAMINATION

- Assumptions:
    1. Lectal difference in morphosyntactic preference, e.g. with *-s* vs. without *-s*
    2. Lectal differences in the lexicon
    3. Language contact
    4. Storage of ready-made strings, e.g. "iets leuks"

$\Longrightarrow$ Lectal contamination: lexical biases reflect lectal difference

EXAMPLE: LECTAL CONTAMINATION

- Step 1: effect to be explained



EXAMPLE: LECTAL CONTAMINATION

- Step 2: conceptual design
  - 2 populations of agents, e.g. 'Belgians', the 'Netherlandics'
  - 2 lexical phrases, e.g. *iets speciaal(s), iets bijzonder(s)*
  - 2 morphological variants, e.g. with *-s*, without *-s*
  - Initial preferences (NOT hard-coded): Belgians prefer *iets speciaal(s)* and *without –s* (relatively), Netherlandics prefer *iets bijzonder(s)* and *with -s*
  - Ready made storage, e.g. *iets bijzonders, iets bijzonder, iets speciaal, iets speciaals*
  - No initial lectal contamination
  - The more often an agent hears a string, the better it is entrenched in the agent's memory
  - The better a string is entrenched in an agent's memory, the more likely the agent is to use it
  - Occasional language contact, but mostly contact between the groups

EXAMPLE: LECTAL CONTAMINATION

- Step 3: Implementation
  - 2 populations of agents, e.g. 'Belgians', the 'Netherlandics'
  - 2 lexical phrases, e.g. *iets speciaal(s), iets bijzonder(s)*
  - 2 morphological variants, e.g. with *-s*, without *-s*
  - Initial preferences (NOT hard-coded): Belgians prefer *iets speciaal(s)* and *without -s* (relatively), Netherlandics prefer *iets bijzonder(s)* and *with -s*
  - Ready made storage, e.g. *iets bijzonders, iets bijzonder, iets speciaal, iets speciaals*
  - No initial lectal contamination

```
class Agent:

   def __init__(self, nationality, idnumber):
      self.nationality = nationality
      self.idnumber = idnumber
      if self.nationality == netherlandic':
         self.memory = copy.deepcopy({'iets bijzonders': 80, 'iets bijzonder': 0, 'iets speciaals': 20, 'iets speciaal': 0})
      elif self.nationality == 'belgian':
         self.memory = copy.deepcopy({'iets bijzonders': 12, 'iets bijzonder': 8, 'iets speciaals': 48, 'iets speciaal': 32})
      else:
         raise ValueError('unknown nationality')
```

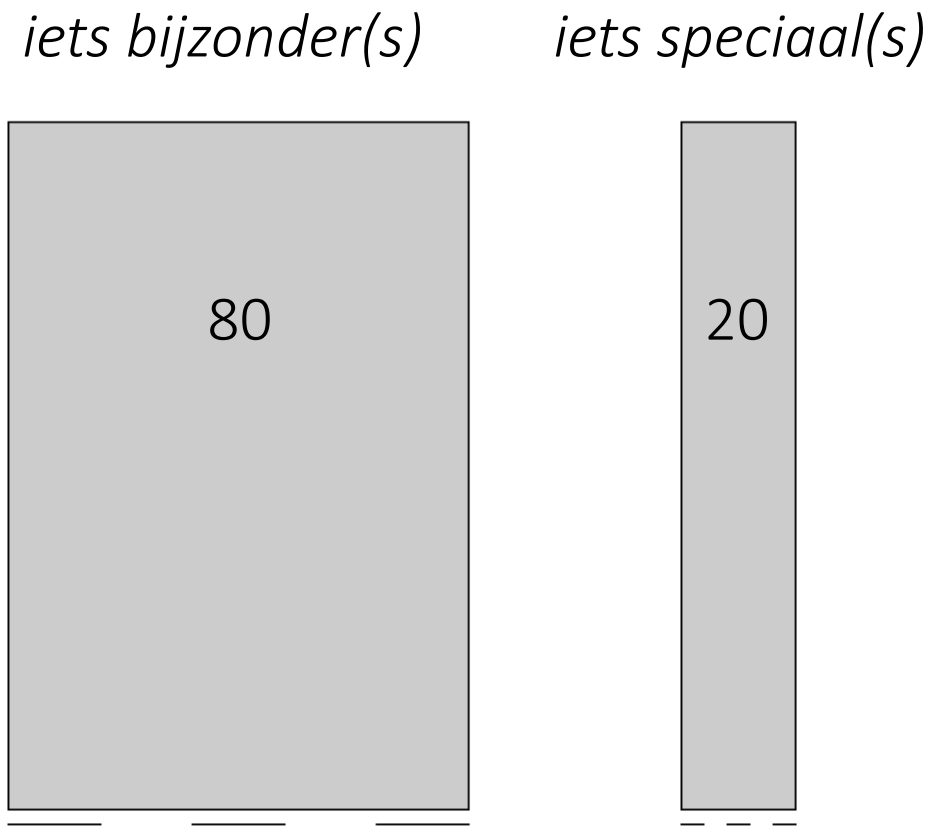
EXAMPLE: LECTAL CONTAMINATION

- Initial memory of a 'Netherlandic' agent
  - Heard 100 strings
  - Lexical preference: 80% - 20%
  - Morphological preference: 100% - 0%
  - Exactly independent

with *-s*

  - iets bijzonders: 80
  - iets bijzonder: 0
  - iets speciaals: 20
  - iets speciaal: 0

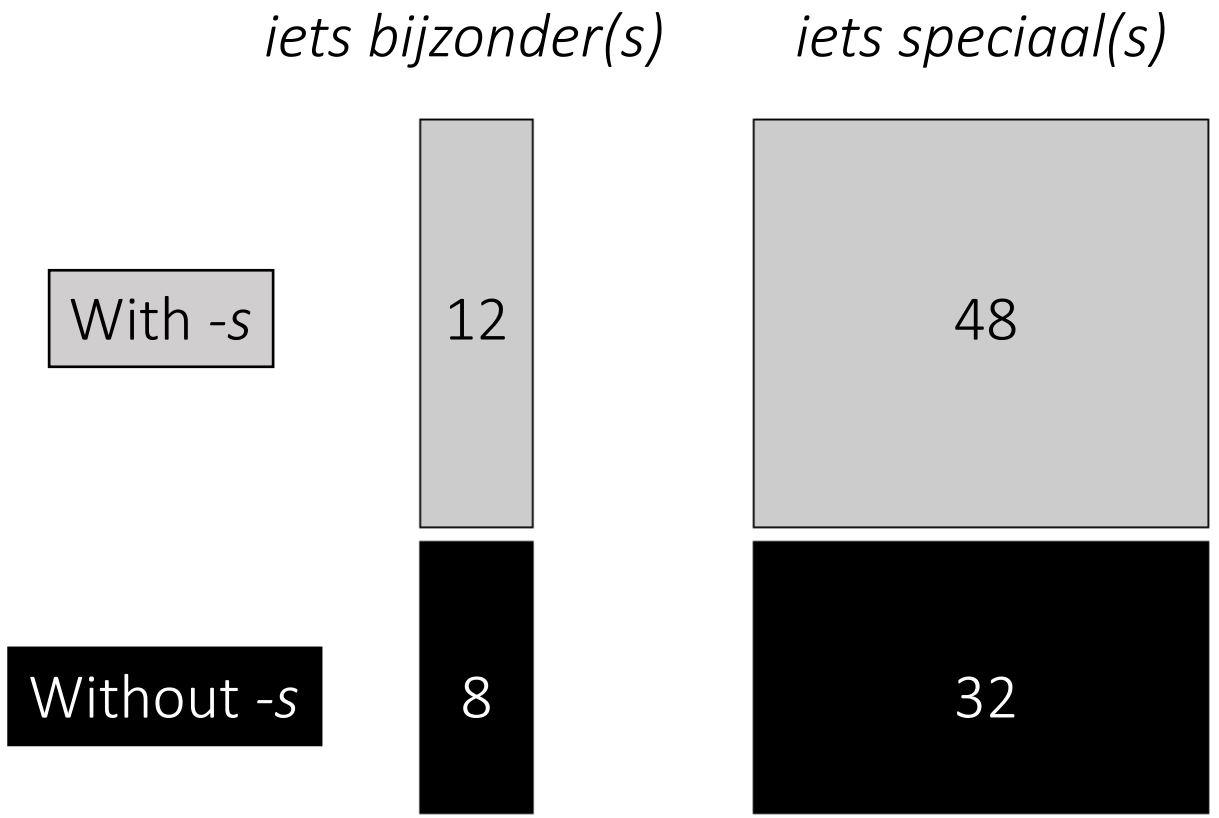
Without *-s*



EXAMPLE: LECTAL CONTAMINATION

- Initial memory of a 'Belgian' agent
  - Heard 100 strings
  - Lexical preference: 20% - 80%
  - Morphological preference: 60% - 40%
  - Exactly independent

  - iets bijzonders: 12
  - iets bijzonder: 8
  - iets speciaals: 48
  - iets speciaal: 32



EXAMPLE: LECTAL CONTAMINATION

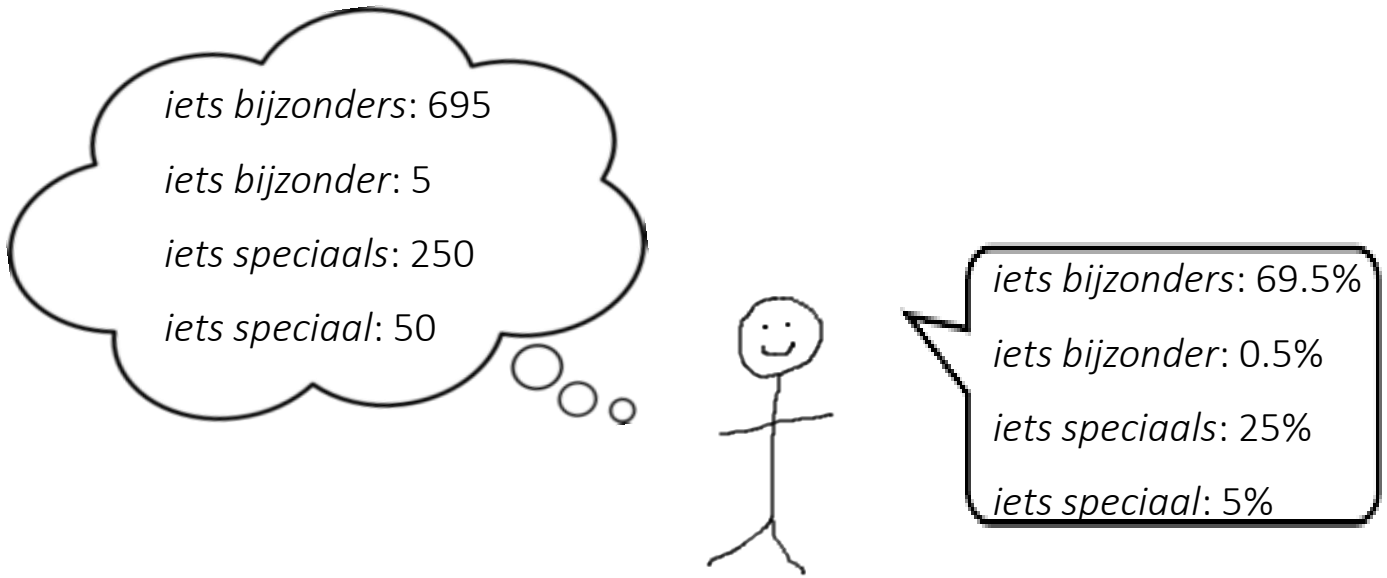
- Step 3: Implementation
  - — The more often an agent hears a string, the better it is entrenched in the agent's memory
  - → Many possible ways to implement: start with the simplest one

```
class Agent:

   def update(self, heard_form):
         self.memory[heard_form] += 1
```

EXAMPLE: LECTAL CONTAMINATION

- Step 3: Implementation
  - — The better a string is entrenched in an agent's memory, the more likely the agent is to use it
  - → Many possible ways to implement: start with the simplest one

```
class Agent:

   def produce(self):
```



EXAMPLE: LECTAL CONTAMINATION

- Step 3: Implementation
  - — Occasional language contact, but mostly contact between the groups
  - → Parameters: 'netherlandic_speaks_to_belgian': 0.01, 'belgian_speaks_to_netherlandic': 0.01

EXAMPLE: LECTAL CONTAMINATION

- Rest of simulation
  - World: keeps track of what has been said during last *record_every*

```
class World:

   def __init__(self):
      self.belgian_corpus = copy.deepcopy(OrderedDict(sorted({"iets bijzonder": 0, "iets bijzonders": 0, "iets speciaal": 0, "iets
      speciaals": 0}.items())))
      self.netherlandic_corpus = copy.deepcopy(OrderedDict(sorted({"iets bijzonder": 0, "iets bijzonders": 0, "iets speciaal": 0,
      "iets speciaals": 0}.items())))
```

  - run_interaction(): produce, update, register in world

```
def run_interaction(world, speaker, hearer, series_configurations):
```

EXAMPLE: LECTAL CONTAMINATION

- Rest of simulation
  - run_series(): Heavy lifting
    - Initializes everything
    - Point_in_time: makes sure time scales with population size
    - At each point_in_time: number of interactions = number of agents
    - 27 Belgian agents & 73 Netherlandic agents: at each point_in_time, 27x randomly select a Belgian speaker and 72x randomly select a Netherlandic speaker
    - Deletes everything

EXAMPLE: LECTAL CONTAMINATION

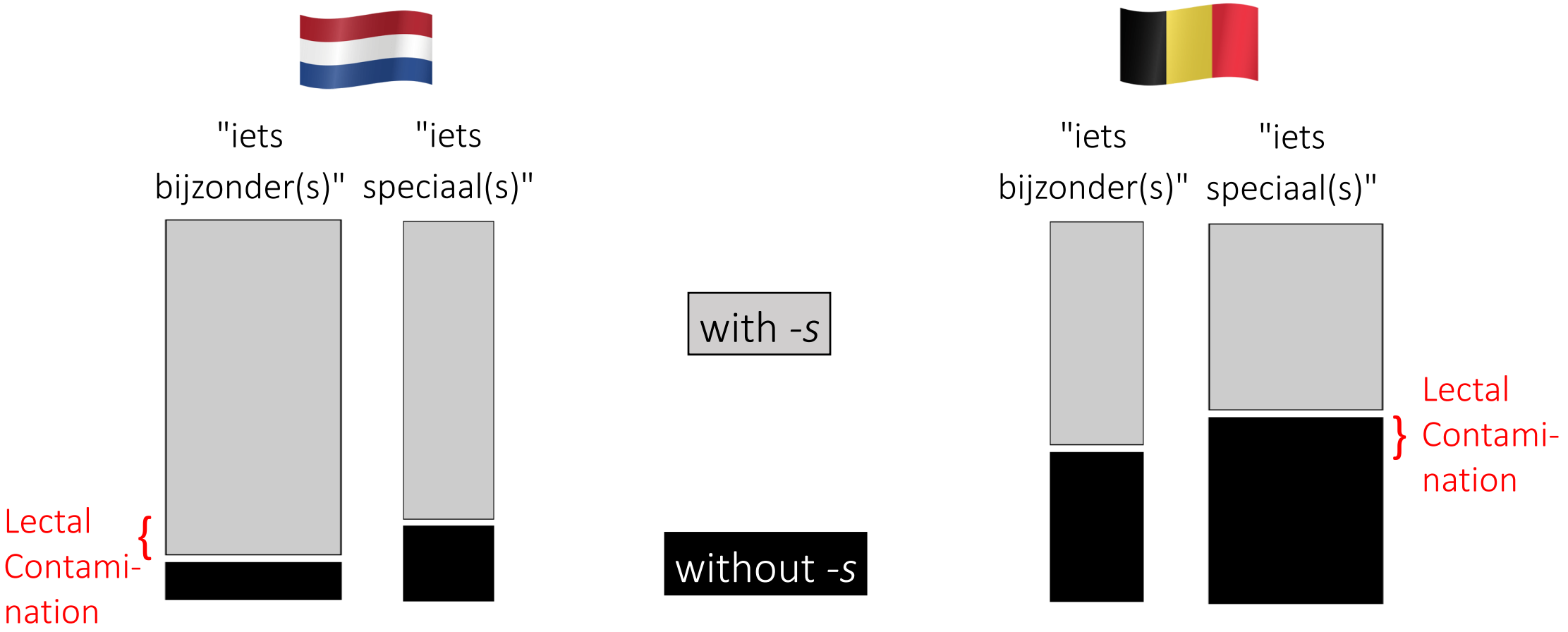
- Rest of simulation
  - — run_batch(): runs X series
  - — write_corpora_to_file(): writes the World + extra information to an outputfile, every *record_every*
  - ! Record corpora or record memory?

EXAMPLE: LECTAL CONTAMINATION

- interaction
- point_in_time (one "day", "week", "year")
- series
- batch
- Run multiple batches with multiple configurations

EXAMPLE: LECTAL CONTAMINATION

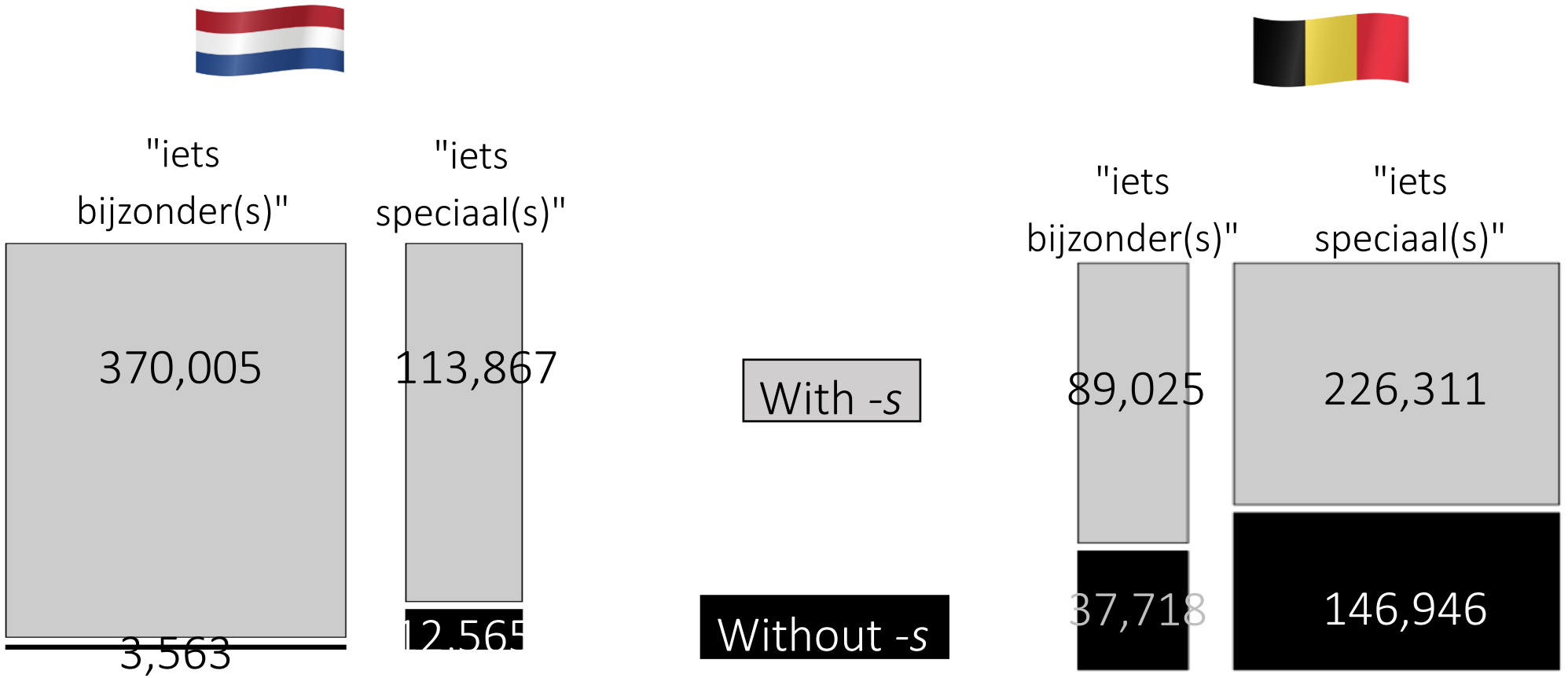
- Step 4: Evaluation. The effect to be explained



- 100 agenten: 50 Netherlandics, 50 Belgians
- 1 series
- 1.000.000 points_in_time
- Record_every: 10.000
- 0.01 language contact

EXAMPLE: LECTAL CONTAMINATION

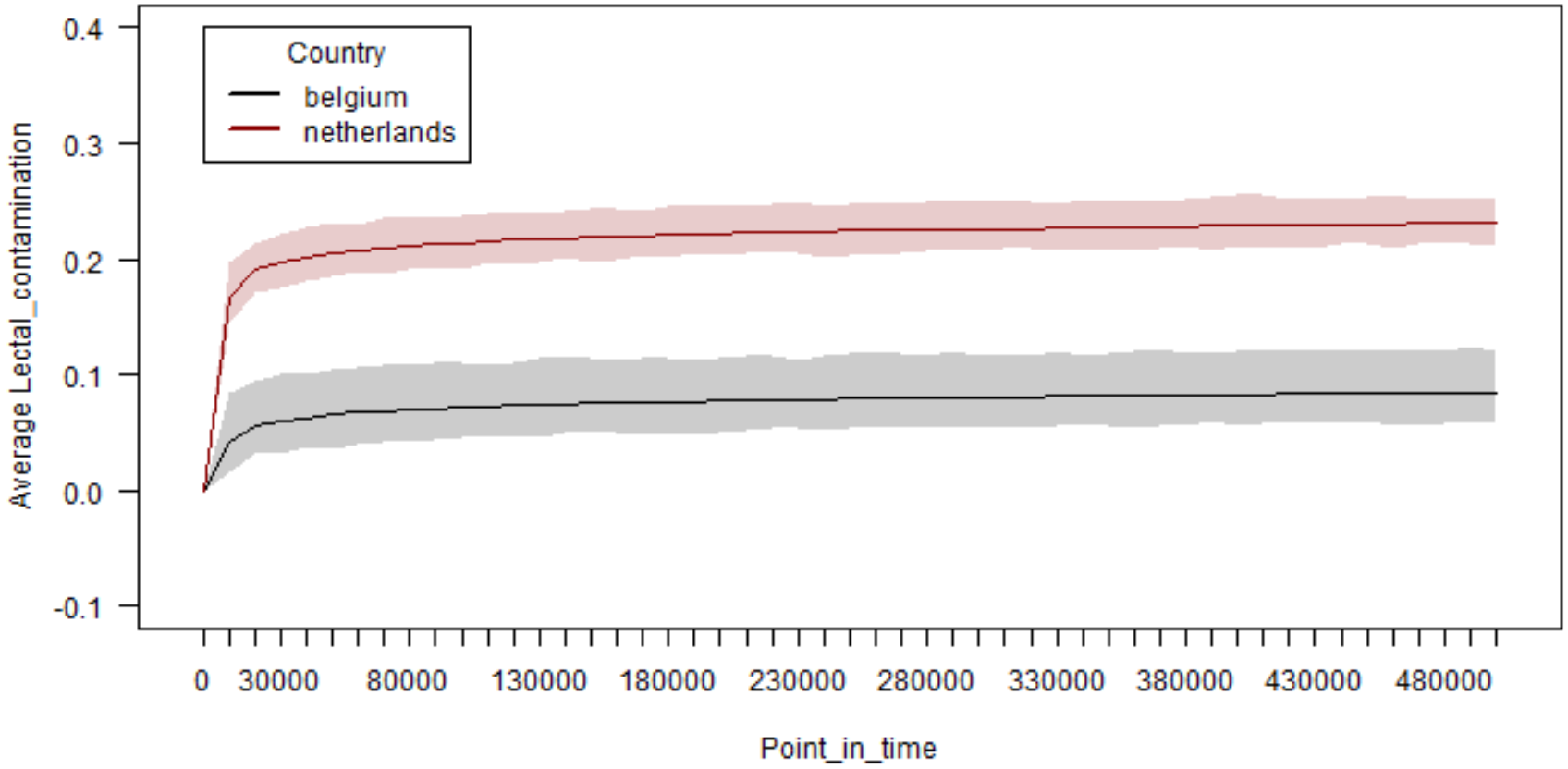
- Step 4: Evaluation. Final record_every



EXAMPLE: LECTAL CONTAMINATION

- Step 4: Evaluation. Graph design:
  - At every record_every (e.g. 10.000 points_in_time), calculate Cramer's V: positive if in the right direction, negative if in the opposite direction
  - Cramer's V on y-axis, point_in_time on x-axis
  - Two lines: among the Belgian agents & among the Netherlandic agents
  - 50 Netherlandic agents, 50 Belgian agents, language contact: 0.01, 1.000.000 points_in_time, 10 series, record_every: 10.000

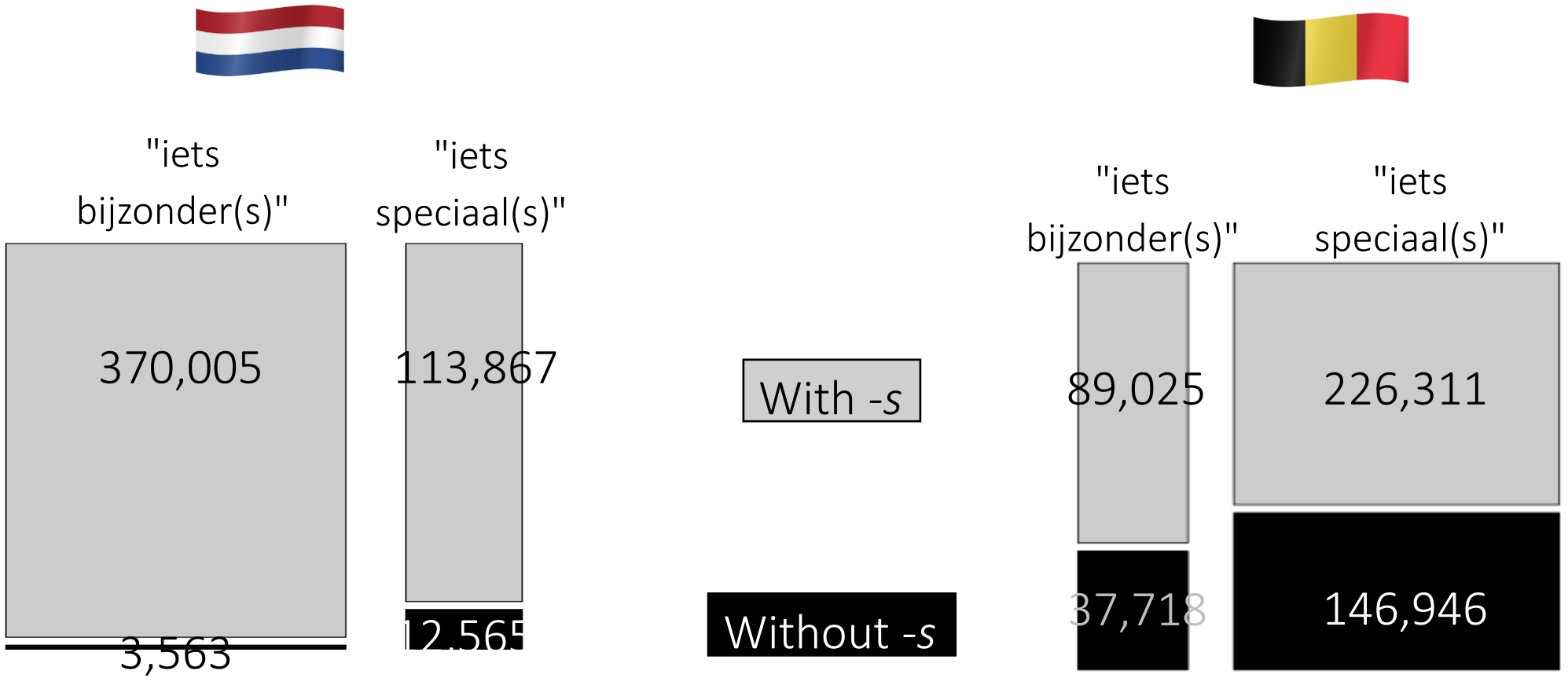
EXAMPLE: LECTAL CONTAMINATION



- 100 agenten: 50 Netherlandics, 50 Belgians
- 1 series
- 1.000.000 points_in_time
- Record_every: 10.000
- 0.01 language contact

EXAMPLE: LECTAL CONTAMINATION

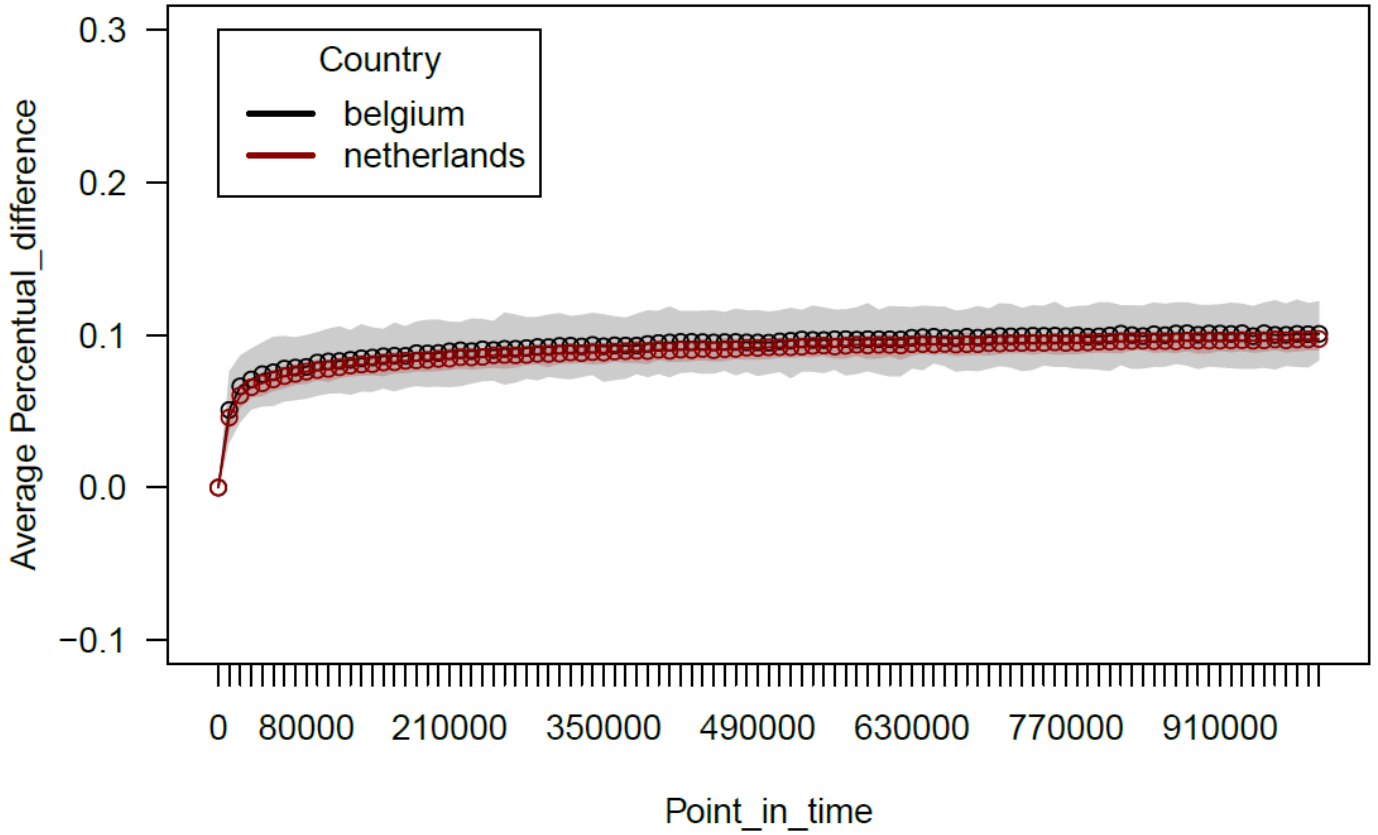
- Step 4: Evaluation. Final record_every



- 100 agenten: 50 Netherlandics, 50 Belgians
- 1 series
- 1.000.000 points_in_time
- Record_every: 10.000
- 0.01 language contact

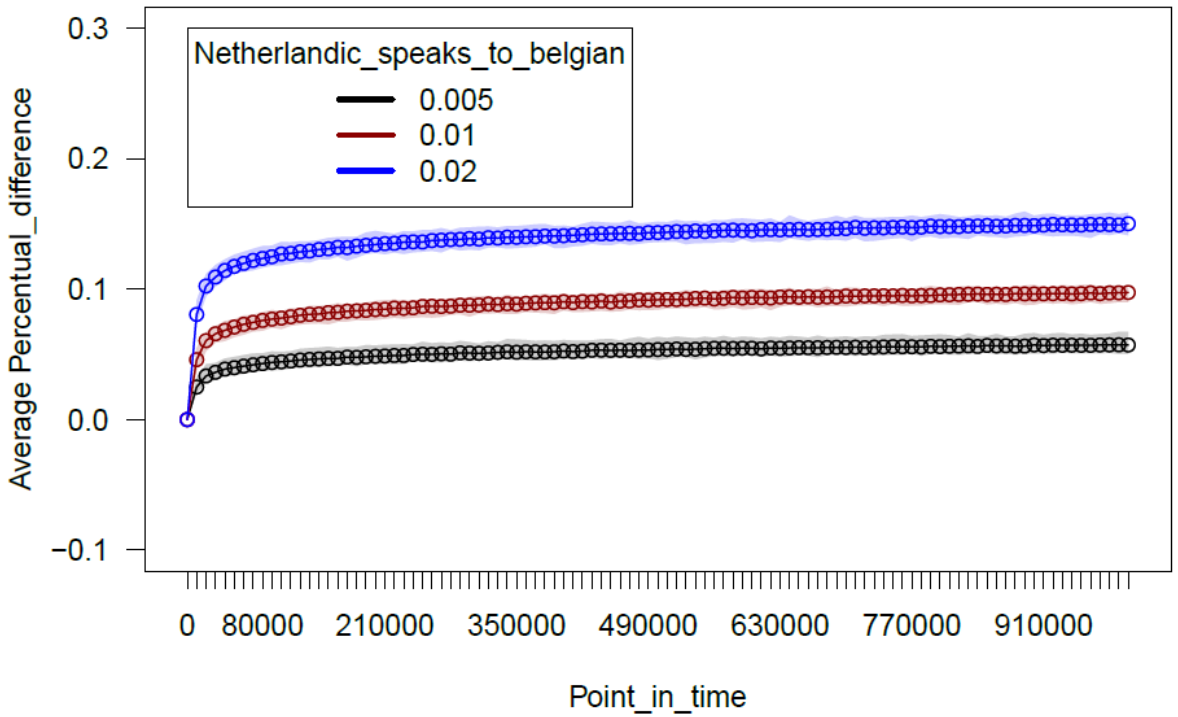
EXAMPLE: LECTAL CONTAMINATION

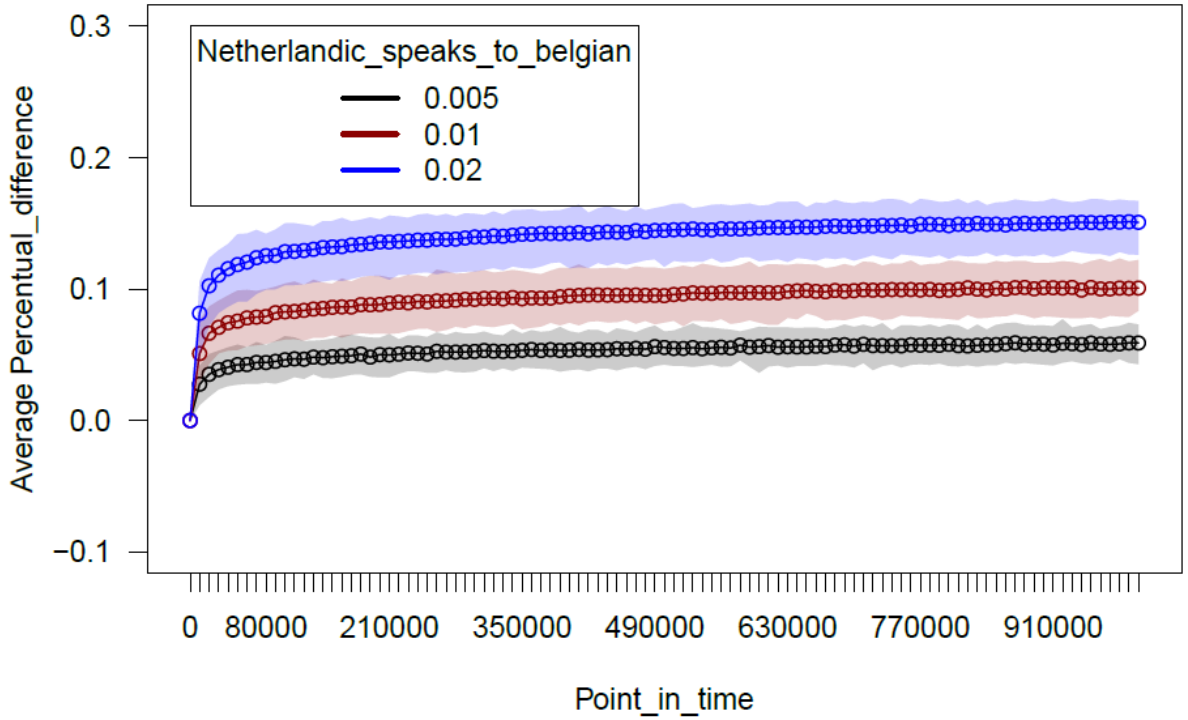
- Step 4: Evaluation. With language contact



EXAMPLE: LECTAL CONTAMINATION

- Step 4: Evaluation. Vary Language contact

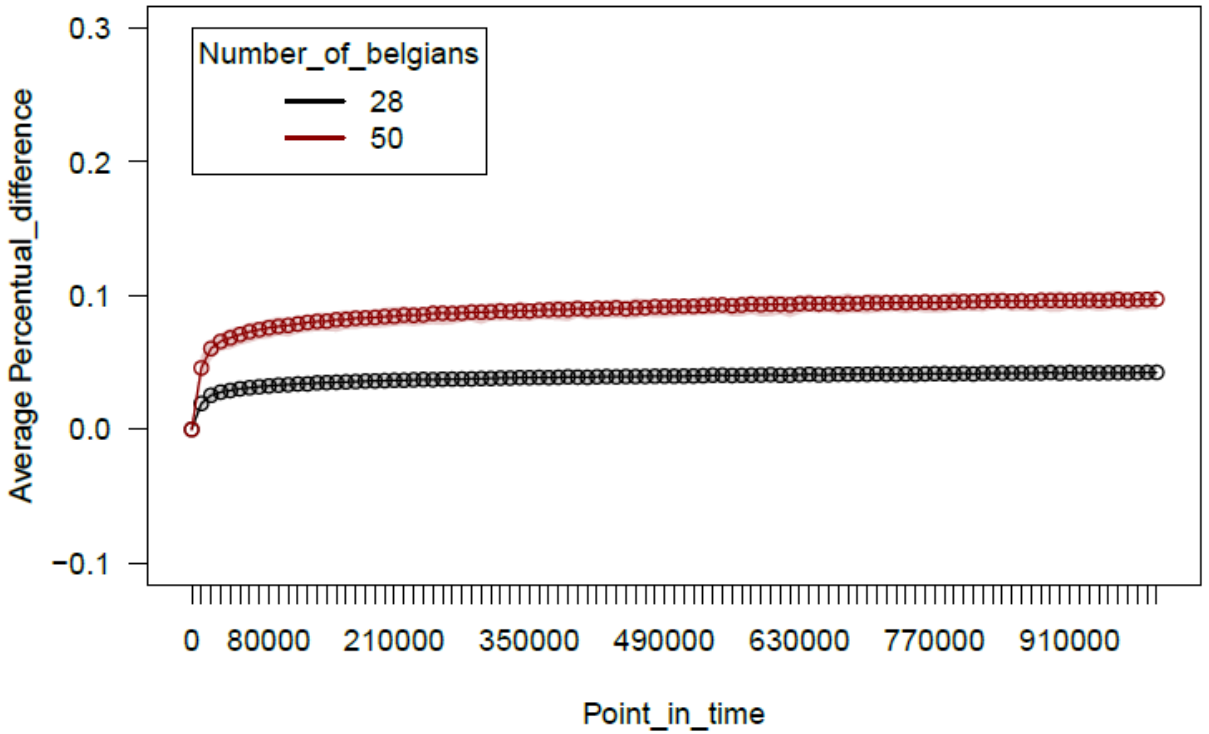


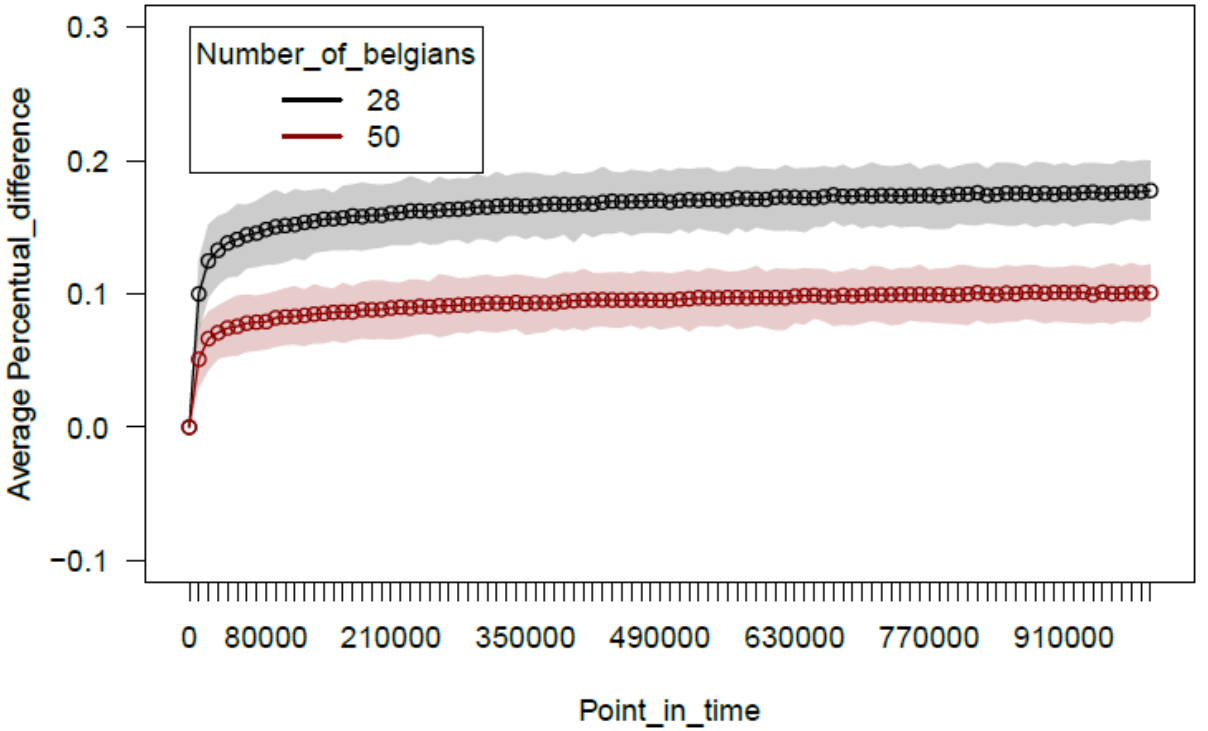


EXAMPLE: LECTAL CONTAMINATION

- Step 4: Evaluation. Assymetric population sizes

(! Side effects)

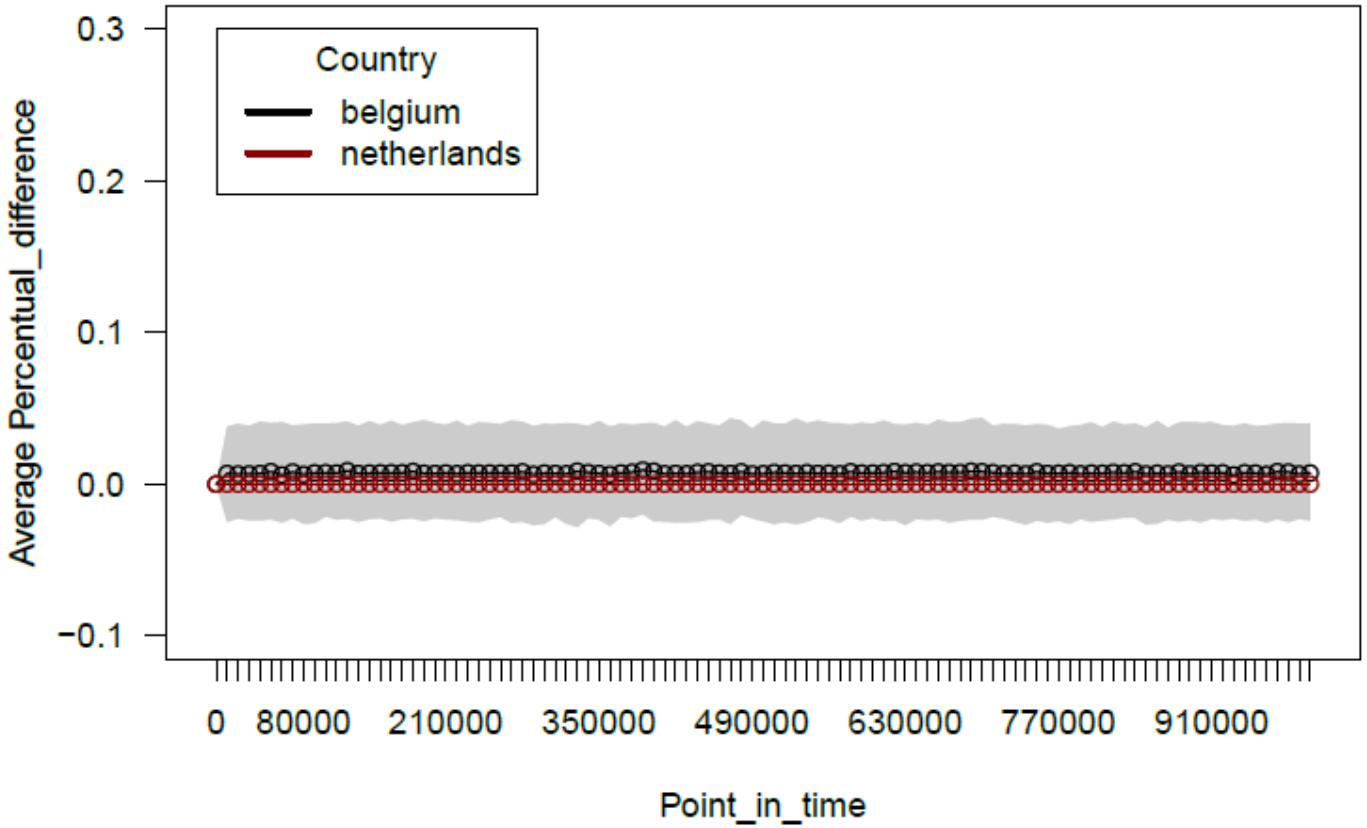




EXAMPLE: LECTAL CONTAMINATION

- 100 agenten: 50 Netherlandics, 50 Belgians
- 1 series
- 1.000.000 points_in_time
- Record_every: 10.000
- 0 language contact

- Step 5: reverse engineer. Remove Language contact entirely



EXAMPLE: LECTAL CONTAMINATION

- 100 agenten: 50 Netherlandics, 50 Belgians
- 1 series
- 1.000.000 points_in_time
- Record_every: 10.000
- 0 language contact

- Step 5: reverse engineer.
  - — Lectal difference in morphosyntactic preference
  - — Lectal differences in the lexicon
  - — Storage of ready-made strings, e.g. "iets leuks"

  ! To be done

EXAMPLE: LECTAL CONTAMINATION

- Why is this model useful?
  - Proves:

IF

1. Lectal difference in morphosyntactic preference, e.g. with –s vs. without –s
2. Lectal differences in the lexicon
3. Language contact
4. Storage of ready-made strings, e.g. "iets leuks"

THEN

Lectal contamination must emerge (unless something else is blocking it)

EXAMPLE: LECTAL CONTAMINATION

- Objection: Belgians have less prestige then Netherlandics

- Additional parameter: effect_of_belgians_on_netherlandics

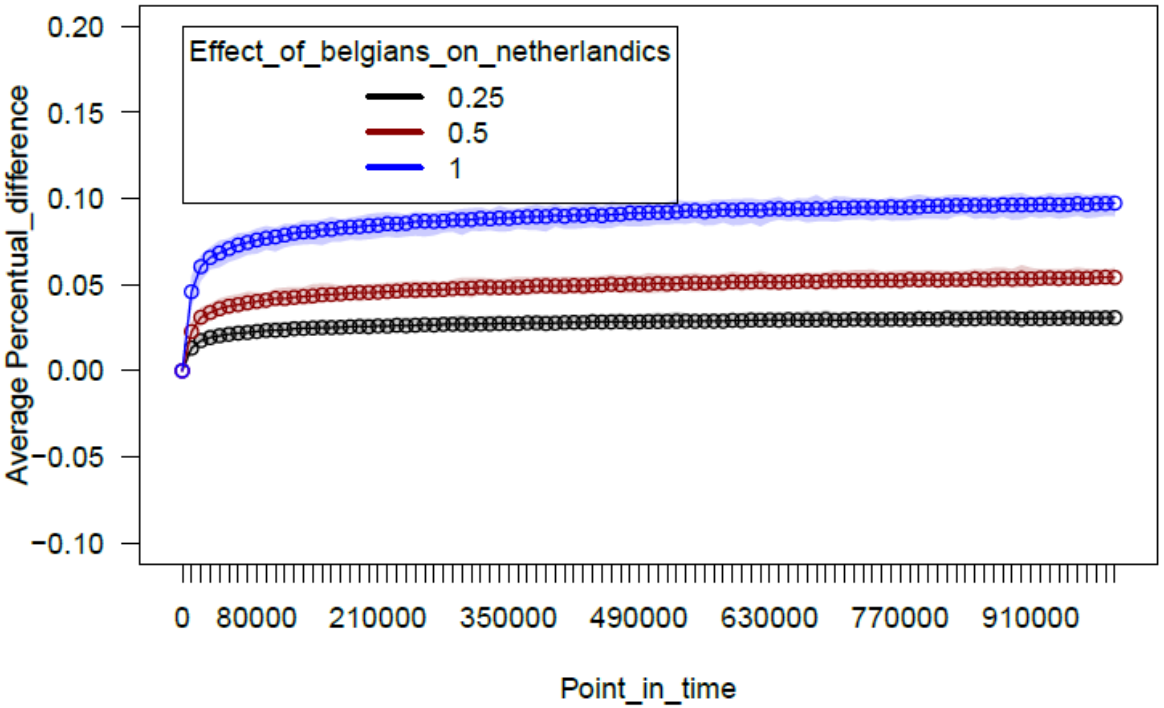
EXAMPLE: LECTAL CONTAMINATION

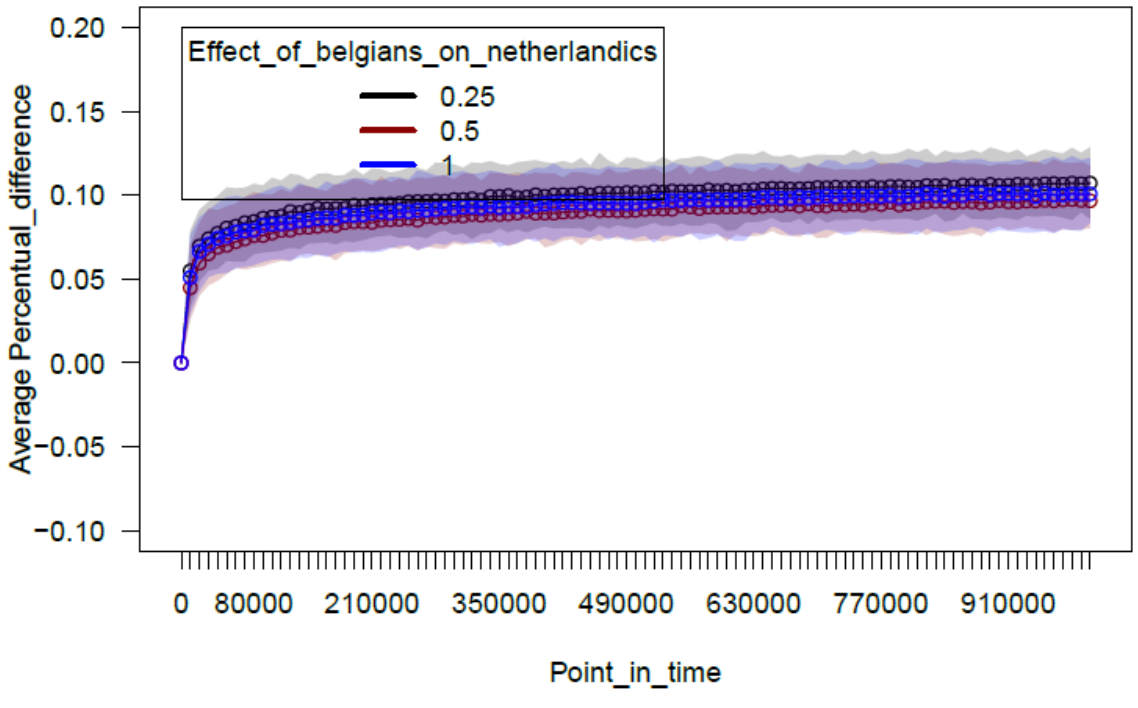
- Additional parameter: effect_of_belgians_on_netherlandics [0,1]

```
class Agent:


   def update(self, heard_form, speaker_nationality, effect_of_belgians_on_netherlandics):
      if self.nationality == 'netherlandic' and speaker_nationality == 'belgian':
         #print(effect_of_belgians_on_netherlandics)
         self.memory[heard_form] += effect_of_belgians_on_netherlandics
      else:
         self.memory[heard_form] += 1
```

EXAMPLE: LECTAL CONTAMINATION

- Step 4: Evaluation. Effect_of_belgians_on_netherlandics





EXAMPLE: LECTAL CONTAMINATION

- Why is this model useful?
  - Make predictions, e.g. relative population size
  - Extension: add 'border areas'

How

1. Define an effect
2. Conceptual design
3. Implementation (and debugging)
4. Evaluation
5. Reverse engineer
6. Write paper

How

- Components of an agent-based model
  - Agent class
    - Memory/language system
    - Produce & comprehend/update
  - World class
  - Interactions
  - Points in time
  - Series
  - Batches
  - Writer functions/record_every

How

- Best practices
  - Keep data and scripts separate, e.g. initial memories in separate files
  - Make it scalable: parameter settings should function independently of one another, e.g. time and interactions, language contact and population size
  - Separate: model ⟶ data ⟶ analyses
  - Write data to new folder for each batch
  - Keep a log-file with the batches you have run and their parameter settings
  - Add as many errors messages as possible
  - Always first assume it's a bug

How

- Best practices
  - KISS, e.g. Cramer's V

REFERENCES

Macwhinney, Brian and Jared Leinbach. 1991. Implementations are not conceptualizations: revising the verb learning model. *Cognition* 40(1–2). 121.

Landsbergen, Frank, Robert Lachlan, Carel ten Cate and Arie Verhagen. 2010. A cultural evolutionary model of patterns in semantic change. *Linguistics* 48(2). 363.

Silver, Nate. 2012. *The signal and the noise. Why so many predictions fail - but some don't*. New York: The Pinguin Press.

Bazghandi, Ali. 2012. Techniques, Advantages and Problems of Agent Based Modeling for Traffic Simulation. *International Journal of Computer Science Issues* 9(1). 115–119.

Dhamdher, Amogh and Constantine Dovrolis. 2009. An agent-based model for the evolution of the internet ecosystem. *COMSNETS. 1st International Conference on Communication Systems and Networks*, 1–10.

Brock, William, Cars Hommes and Florian Wagener. 2009. More hedging instruments may destabilize markets. *Journal of Economic Dynamics and Control* 33(11). Elsevier B.V. 1912–1928.

Boer, Bart de. 2012. Modelling and Language Evolution: beyond fact-free science. In Luke McCrohon, Tomomi Fujimura, Kazuo Okanoya, Koji Fujita, Reiji Suzuki, Roger Martin & Noriaki Yusa (eds.), *The Evolution of Language: Proceedings of the 9th International Conference*, 83–92. Evolang-9 Organizing Committee.

Guerreiro, Orlando and Miguel Ferreira. 2013. Towards an Agent Based Modeling : The Prediction and Prevention of the Spread of the Drywood Termite Cryptotermes brevis. *Progress in Artificial Intelligence. 16th Portuguese Conference on Artificial Intelligence, EPIA 2013*, 480–491.

Lestrade, Sander. 2015. Simulating the development of bound person marking. In Johannes Wahle, Marisa Köllner, Harald Baayen, Gerhard Jäger & Tineke Baayen-Oudshoorn (eds.), *Proceedings of the 6th Conference on Quantitative Investigations in Theoretical Linguistics*. Tübingen.

Pinker, Steven and Alan Prince. 1988. On language and connectionism: Analysis of a parallel distributed processing model of language acquisition. *Cognition* 28(1). 73–193.

Ling, Charles and Marin Marinov. 1993. Answering the connectionist challenge: a symbolic model of learning the past tenses of English verbs. *Cognition* 49(3). 235–290.

Noord, Rik van and Jennifer Spenader. 2015. Modeling the learning of the English past tense with memory-based learning. *Computational linguistics in the Netherlands Journal* 5. 65–80.

Valverde, Jose. 2001. Molecular Modelling: Principles and Applications. *Briefings in Bioinformatics* 2(2). Oxford: Oxford Publishing Limited(England). 199–200.

Beuls, Katrien and Luc Steels. 2013. Agent-Based Models of Strategies for the Emergence and Evolution of Grammatical Agreement. *PLoS ONE* 8(3). e58960.

Plunkett, Kim and Virginia Marchman. 1993. From rote learning to system building: acquiring verb morphology in children and connectionist nets. *Cognition* 48(1). 21–69.

Plunkett, Kim and Virginia Marchman. 1991. U-shaped learning and frequency effects in a multi-layered perception: Implications for child language acquisition. *Cognition* 38(1). 43–102.

Liska, Matthew, Alexander Tchekhovskoy, Ann Ingram and Michiel van der Klis. 2019. Bardeen–Petterson alignment, jets, and magnetic truncation in GRMHD simulations of tilted thin accretion discs. *Monthly Notices of the Royal Astronomical Society* 487(1). 550–561.

Bloem, Jelke. 2015. An agent-based model of a historical word order change. In Robert Berwick, Anna Korhonen, Alessandro Lenci, Thierry Poibeau & Aline Villavicencio (eds.), *Proceedings of the Sixth Workshop on Cognitive Aspects of Computational Language Learning*, 22–27. Lisbon: Association for Computational Linguistics.

Rumelhart, David and James McClelland. 1986. On learning the past tense of English verbs. In David Rumelhart & James McClelland (eds.), *Parallel distributed processing: explorations in the microstructure of cognition*, 216–271. Cambridge: MIT Press.

Pijpops, Dirk, Katrien Beuls and Freek Van de Velde. 2015. The rise of the verbal weak inflection in Germanic. An agent-based model. *Computational linguistics in the Netherlands Journal* 5. 81–102.

Plunkett, Kim and Patrick Juola. 1999. A Connectionist Model of English Past Tense and Plural Morphology. *Cognitive Science* 23(4). 463–490.

Jaeger, Herbert, Luc Steels, Andrea Baronchelli, Ted Briscoe, Morten Christiansen, Thomas Griffiths, Gerhard Jäger, et al. 2009. What Can Mathematical, Computational, and Robotic Models Tell Us about the Origins of Syntax? *Biological Foundations and Origin of Syntax*. The MIT Press.