11.06.2019

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

The Case for Custom Software Development

The Example of *HGSimpleCorpusNetwork*

Beatrix Busse, Ingo Kleiber

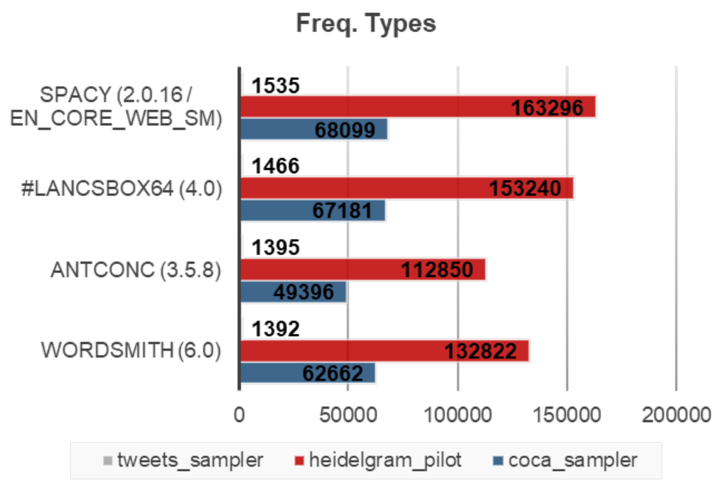
Gefördert durch
DFG Deutsche Forschungsgemeinschaft

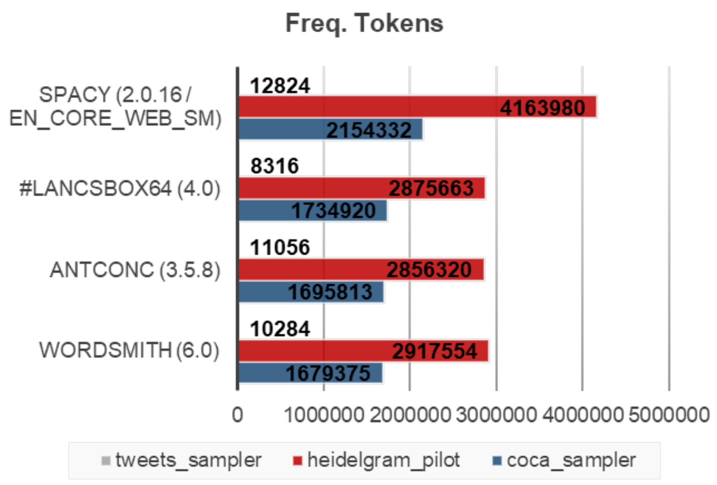
Agenda

1. The Role of Software in Linguistic Research
2. Requirements for Academic Software
3. The Need for Project-Specific Software
4. HeidelGram
   (1) Project Overview
   (2) HGSimpleCorpusNetwork
5. Do You Need to Develop Project-Specific Software?
6. Best Practices

The Role of Software in Linguistic Research

Three **corpora** and four common **toolkits** (frequency analysis)





SD	58	19399	7470

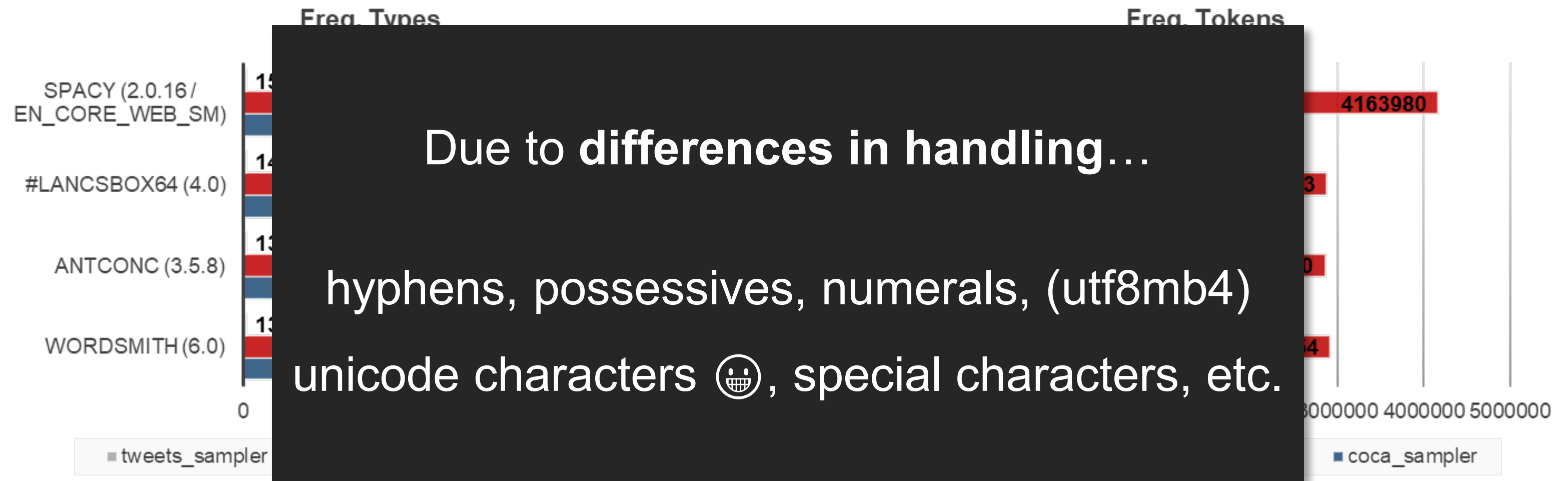
SD	1617	555044	196312

→ Default settings for all toolkits
→ No efforts were made to mitigate differences (e.g. encoding or adjusting delimiters)

The Role of Software in Linguistic Research

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

Three **corpora** and four common **toolkits** (frequency analysis)



Due to **differences in handling**…

hyphens, possessives, numerals, (utf8mb4)

unicode characters 😬, special characters, etc.

SD	58	19399	7470	**SD**	1617	555044	196312

→ Default settings for all toolkits
→ No efforts were made to mitigate differences (e.g. encoding or adjusting delimiters)

The Role of Software in Linguistic Research

“The functionality offered by software tools largely dictates what corpus linguistics research methods are available to a researcher.” **and** “… differences in the way tools are designed will have an impact on almost all corpus analyses.” (Anthony 2013: 141, 151)

We need to consider software as a key part of research which is as important as theory, methodology, and data.

Anthony’s (2013: 158f.) Proposal

“[A] next generation of corpus tools that are built in an open-source and modular fashion, and are developed as a community effort.”

Requirements for Academic Software

Academic software → Part of the research process

- Reproducibility
- Availability (Will the software be available in 5/10/20 years?)
- Transparency (e.g. models, measures, pre-/post-processing)
- Flexibility (cf. Mason 2000: 4)
- Compatibility (e.g. standard and open formats)
- “Pipelinability” (Will the software work as part of a data-analysis pipeline?)
- Usability + Sensible defaults
- Openness (Will there be licensing issues?)

The Need for Project-Specific Software

“[Relying on existing programs] not only severely **limits the possibilities for exploring corpus data**, but also **introduces unknown factors** into the research, as it is not always obvious how the software will handle certain features of language.” (Mason 2008: 155)

- We don’t want to be limited by existing tools and approaches

  → Theory <> Tools instead of Tools → Theory

- We want to be able to solve complex tasks highly specific to our research
- We want to be able to fully understand the tools and the underlying models

  → could be solved by better/full documentation and OSS

  → development forces us to understand all the parts

Example
HeidelGram
HGSimpleCorpusNetwork

HeidelGram
Project

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

HeidelGram

- Compiling and analyzing a corpus of historical English grammars
- Combining corpus-based historical linguistics and the analysis of networks

Research Questions (Examples)

- How do grammar writers react to other authors, and by which means do they try to influence their respective audiences?
- Where do authors position their work with respect to the prescriptive-descriptive continuum?
- What does the interplay between language norms and language usage with respect to normative grammar writing look like?
- Which fields of linguistic study and which topics are treated in grammar writing over the centuries and how do new concepts emerge while others become obsolete?

HeidelGram
Project

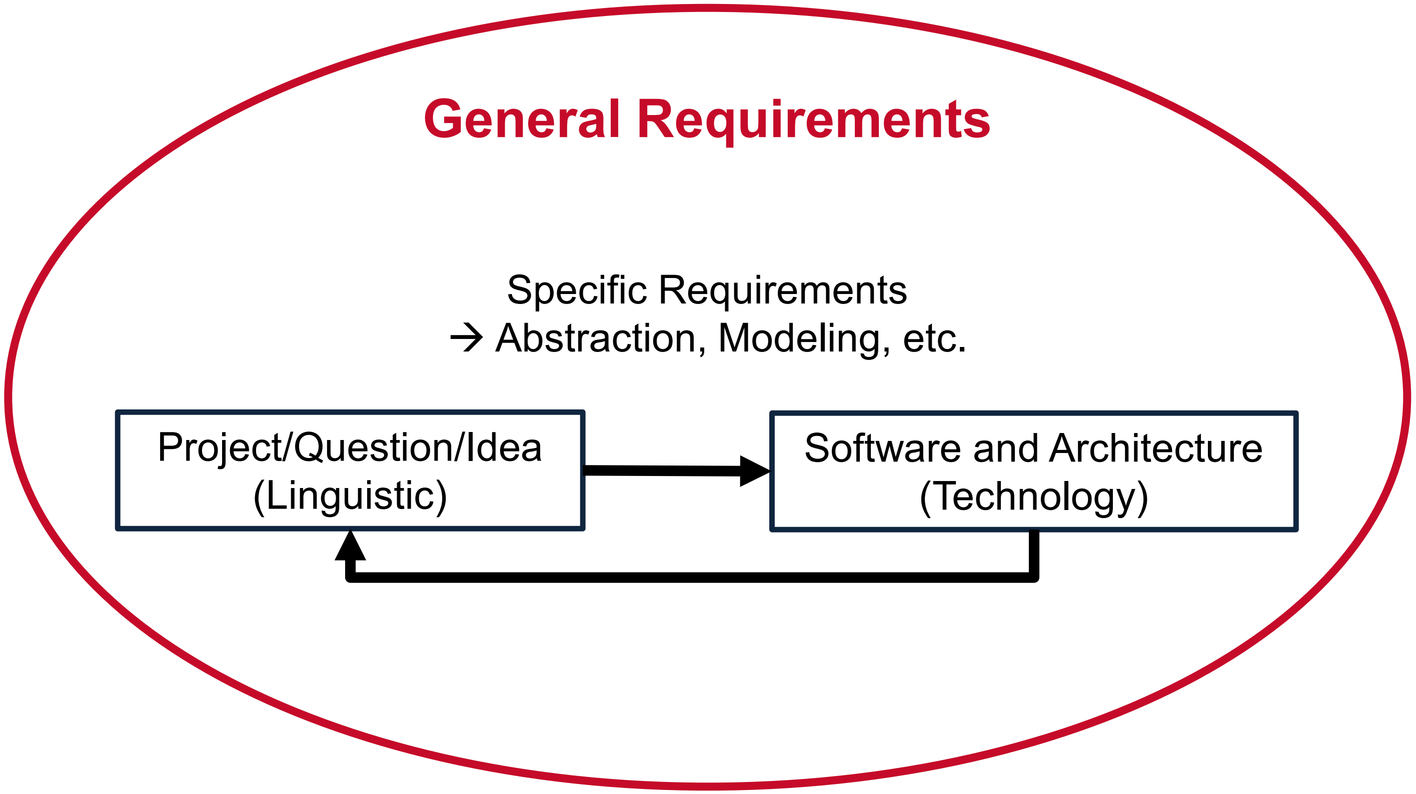
UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

So far …

- Pilot-corpus of 19th-century grammar books (approx. 3m words)
- Analysis of the references made to grammarians/grammars in these grammar books (= scholarly networks → texts and authors)
- Identification of generalizable Verbal Hygiene (Cameron [1995] 2012) patterns
- Development of a specialized **software toolkit for the project**

(cf. Busse, Gather, Kleiber 2018, 2019, forth; Busse and Kleiber 2018)

HeidelGram
Development Process


General Requirements
Specific Requirements
→ Abstraction, Modeling, etc.
Project/Question/Idea (Linguistic)
Software and Architecture (Technology)

HeidelGram
Software: General Requirements

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

The toolbox needs to …

1. grow and be adapted as the project grows
2. be able to handle textual data from a variety of periods (Historical!)
3. be able to handle uncleaned OCR data
4. be able to work as part of an automized analysis pipeline
5. be able to interface with network/graph analysis toolkits such as *Gephi*

HeidelGram
Specific Required Functionalities

1. Generate document-term matrices based on sets of search terms and OCR-based corpora.

Reason: We want to analyze which grammarians (and grammars) are referenced where (by whom) in a corpus of grammar books. (cf. Busse, Gather, Kleiber 2019)

Challenges:

- Uncleaned OCR data is not well-structured and contains spelling errors.
- The mere existence of a token (e.g. *Crombie*) does not necessarily entail a reference.

HeidelGram
Specific Required Functionalities

2. Automatically extract a set of forms which possibly correspond to a particular function (e.g. Verbal Hygiene (Cameron [1995] 2012)).

Reason: We want to find and analyze patterns of Verbal Hygiene in historical grammar books. In order to do that, we need a comprehensive set of forms and examples.

For example: “shows an unpardonable indifference to perspicuity, consistency, and common sense” (Murray 1847) (cf. Busse & Kleiber 2018)

Challenges:

- How can we identifiy patterns/n-grams related to a specific function?
- How can we identify (word-)forms related to a specific function which have not been previously identified as such?

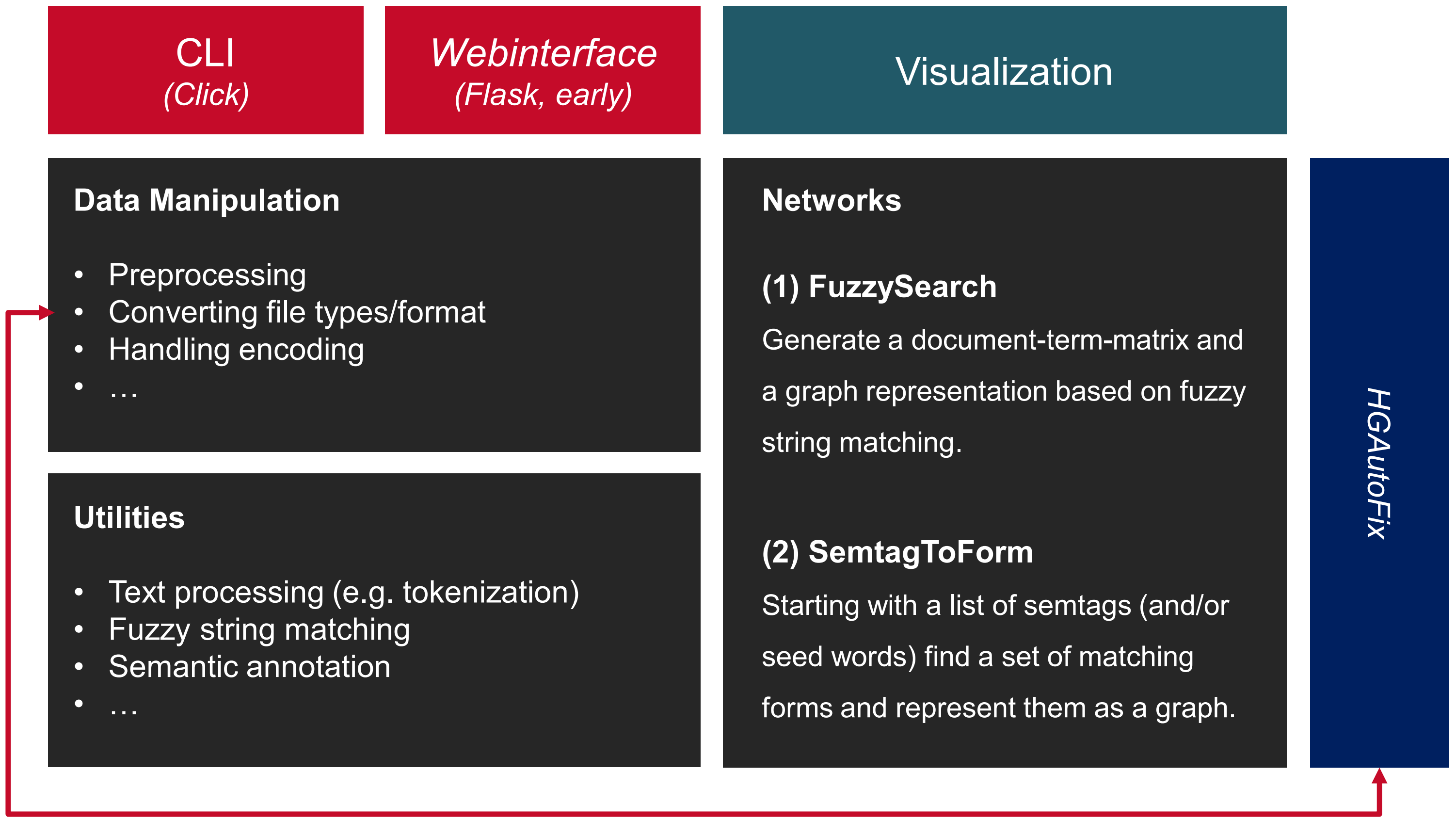
HGSimpleCorpusNetwork
Overview

HGSimpleCorpusNetwork is a project-specific toolbox has been developed alongside the HeidelGram project.

- Written in modern Python (3.6+)
- Both a library and a toolkit
- Optimized for modularity and flexibility
- Research- and theory-driven development
- Roughly based on a Model-View-Presenter approach
- Input data → output data without interaction
- Currently under active development
- Open Source (current version will be published on GitHub soon)

HGSimpleCorpusNetwork
The Toolbox *(as of now)*

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386



Data Manipulation

- Preprocessing
- Converting file types/format
- Handling encoding
- …

Utilities

- Text processing (e.g. tokenization)
- Fuzzy string matching
- Semantic annotation
- …

Networks

(1) FuzzySearch

Generate a document-term-matrix and a graph representation based on fuzzy string matching.

(2) SemtagToForm

Starting with a list of semtags (and/or seed words) find a set of matching forms and represent them as a graph.

HeidelGram
Specific Required Functionalities

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

1. Generate document-term matrices based on sets of search terms and OCR-based corpora.

Challenges:

- Uncleaned OCR data is not well-structured and contains spelling errors.
- The mere existence of a token (e.g. *Crombie*) does not necessarily entail a reference.

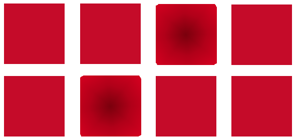
HGSimpleCorpusNetwork
FuzzySearch

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

In & Out

Corpus

(containing **misreadings**)



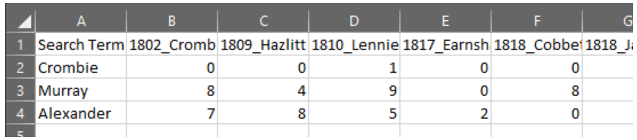
Search Terms

e.g.
Crombie
Murray
Alexander

Crombie ≡ Cr0mble ?

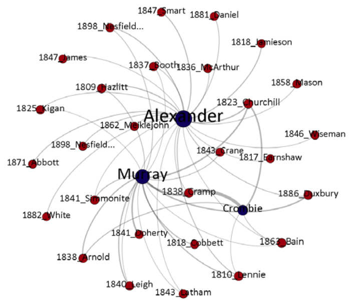
→ Threshold

DTM (csv)

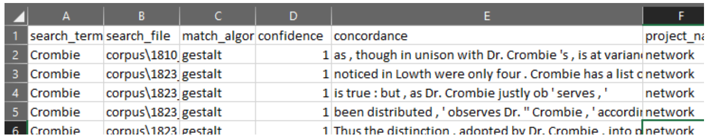


	A	B	C	D	E	F	G
1	Search Term	1802_Cromb	1809_Hazlitt	1810_Lennie	1817_Earnsh	1818_Cobbet	1818_Ja
2	Crombie	0	0	1	0	0	
3	Murray	8	4	9	0	8	
4	Alexander	7	8	5	2	0	
5							

Graph (GEXF, GraphML)



Concordances + Confidence Levels



	A	B	C	D	E	F
1	search_term	search_file	match_algor	confidence	concordance	project_na
2	Crombie	corpus\1810_	gestalt	1	as , though in unison with Dr. Crombie 's , is at varian	network
3	Crombie	corpus\1823_	gestalt	1	noticed in Lowth were only four . Crombie has a list c	network
4	Crombie	corpus\1823_	gestalt	1	is true : but , as Dr. Crombie justly ob ' serves , '	network
5	Crombie	corpus\1823_	gestalt	1	been distributed , ' observes Dr. " Crombie , ' accordi	network
6	Crombie	corpus\1823_	gestalt	1	Thus the distinction , adopted by Dr. Crombie , into n	network

HeidelGram
Specific Required Functionalities

2. Automatically extract a set of forms which possibly correspond to a particular function (e.g. Verbal Hygiene (Cameron [1995] 2012)**).**

Challenges:

- How can we identifiy patterns/n-grams related to a specific function?
- How can we identify (word-)forms related to a specific function which have not been previously identified as such?

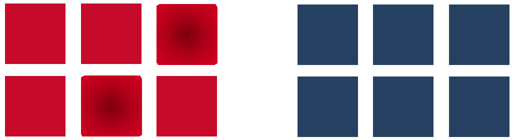
HGSimpleCorpusNetwork
SemtagToForm

In & Out

via USAS or HTST API

Corpus

(containing **misreadings**) + Annotation (Semantic)



Semtags / Seed

e.g.

A5

02.02.07

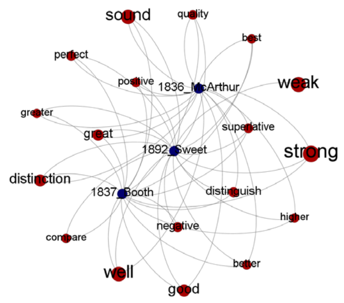
greatly erred

1. Look for seed terms
2. Find forms based on the semtags
3. Build a Word2Vec model
4. Find similar forms to the ones already found

List of Forms (for each document)

civilised	deteriorate
sad	level
invigorate	improved
sharp	defective
overrate	…

Graph (GEXF, GraphML)



Trained Word2Vec model

HGSimpleCorpusNetwork
Lessons Learned

- Software decay/entropy → constant refactoring (expensive, time-consuming)
- External dependencies (e.g. APIs) are a risk → e.g. *UcrelSemTaggerSoapService* going down
- Don’t reinvent the wheel, especially if good models and approaches are available (e.g. tokenization, string matching)
- Modeling parts of the theory in software leads to new insights into both theory and analysis
- Developing project-specific software forces you to have extremely good knowledge of your data and methodology
- Rapidly prototyping new (computational) approaches to analysis fosters creativity (this requires you to have a modular framework and good pipelines)

Do You Need to Develop Project-Specific Software?

Do think about developing software if …

- the skillset/capabilities are available to you and/or your team.
- your developer(s) have (enough) domain-specific knowledge.
- your project is big/important enough to spend the extra time and money.
- the new software potentially could benefit others as well. (= generalizability)
- the existing solutions are (persistently) intransparent and/or inaccessible.
- the existing solutions do not meet academic standards.
- the existing solutions are truly limiting your analysis.

and *vice versa*.

In short: ***Don't develop for the sake of developing!***

Development
Best Practices

tl;dr: *Carefully follow established industry standards and general best practices.*

- **Open Source Software** (OSS) + **DOI** + **Archive repository**
- Strong **version control** (e.g. Git) + branching → documentation of the process
- Testing / **Test-Driven Development** (Unit Tests, Integration Tests, Regression Tests)
- Run **simulations** and test against (reasonable) ranges and means → testing fuzzy returns
- **Continuous integration** + **Sanity/coverage** checks before pushing to remote
- Good (and complete) **documentation** (instructions, examples, self-documenting code, automated API documentation)
- **Reproducible runtimes** (e.g. Docker)
- **Limited reliance** on external (and specifically proprietary) modules/libraries

Valuable Guidelines: Software Sustainability Institute | *also see* Harpole 2017

HeidelGram

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

http://heidelgram.uni-heidelberg.de

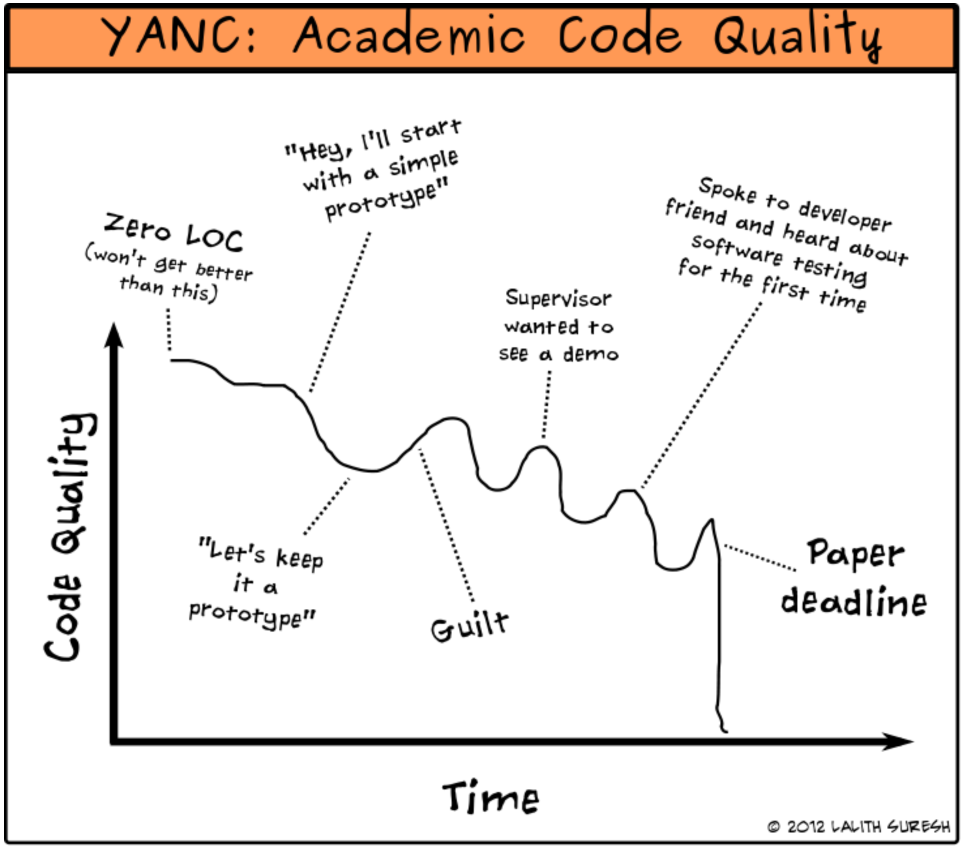
@BeatrixBusse @KleiberIngo

A big thank you also goes to Lyuba Dimitrova who constantly fights off bugs and keeps the code tidy.

Gefördert durch DFG Deutsche Forschungsgemeinschaft

Academic Code Quality

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386


YANC: Academic Code Quality
Zero LOC (won't get better than this)
"Hey, I'll start with a simple prototype"
"Let's keep it a prototype"
Guilt
Supervisor wanted to see a demo
Spoke to developer friend and heard about software testing for the first time
Paper deadline
Code Quality
Time
© 2012 LALITH SURESH

Works Cited I

- **Anthony**, Laurence. 2013. “A Critical Look at Software Tools in Corpus Linguistics.” *Linguistic Research* 30 (2): 141–61.
- **Busse**, Beatrix, and Ingo **Kleiber**. 2018. “A Network of Evaluative Terms in 19th-Century British Grammars: Methodological Challenges and Practical Solutions.” ICAME 39, Tampere, Finnland, 2018.
- **Busse**, Beatrix, Kirsten **Gather**, and Ingo **Kleiber**. 2018. “Assessing the Connections between English Grammarians of the Nineteenth Century: A Corpus-Based Network Analysis.” In *Grammar and Corpora 2016*, edited by Eric Fuß, Marek Konopka, Beata Trawiński, and Ulrich H. Waßner, 435–42. Heidelberg: Heidelberg University Publishing.
- **Busse**, Beatrix, Ingo **Kleiber**, and Kirsten **Gather**. 2019. “Paradigm Shifts in 19th-Century British Grammar Writing: A Network of Texts and Authors.” In *Norms and Conventions in the History of English*, edited by Birte Bös and Claudia Claridge, 49–71. Current Issues in Linguistic Theory. Amsterdam: John Benjamins.
- **Busse**, Beatrix, Kirsten **Gather**, and Ingo **Kleiber**. forth. “A Corpus-Based Analysis of Grammarians’ References in 19th-Century British Grammars.” In *Variation in Time and Space: Observing the World through Corpora*, edited by Anna Cermakova and Markéta Malá. Diskursmuster - Discourse Patterns 20. Berlin: De Gruyter.

Works Cited II

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

- **Cameron**, Deborah. (1995) 2012. *Verbal Hygiene.* The politics of language. London: Routledge.
- **Harpole**, Alice. 2017. “Sustainable Scientific Software Development.” europython 2017, Rimini, July 12. https://ep2017.europython.eu/conference/talks/sustainable-scientific-software-development.
- **Mason**, Oliver. 2000. *Programming and Corpus Linguistics.* Edinburgh: Edinburgh University Press.
- **Mason**, Oliver. 2008. “Developing Software for Corpus Research.” *International Journal of English Studies* 8 (1): 141–56.
- **Murray**, Gerald. 1847. *The Reformed Grammar or Philosophical Test of English Composition: Written for the Assistance of Teachers and Satisfaction of Learners.* London: Darton and Co. Holborn Hill.