Arguments for and against DIY corpus tools creation:
A debate about programming

Laurence Anthony
Center for English Language Education in Science and Engineering (CELESE),
Waseda University, Tokyo, Japan
anthony@waseda.jp
http://www.laurenceanthony.net/
Motivations


Overview

- The current state of corpus linguistics tools
  - four generations of corpus tools
  - a need for something new

- DIY Corpus Tools – The Debate
  - Arguments for learning to program
  - Arguments against learning to program

- Thoughts on the future of programming and tools in corpus linguistics research
  - Programming, tools, and statistics
  - Collaboration in project teams
The current state of corpus linguistics tools
four generations of tools
Current state of corpus linguistics tools:
A definition of corpus linguistics

- It is an *empirical* (experimental) approach
  - An analysis of actual patterns of use in target texts
- It uses a *corpus* of natural texts as the basis for analysis
  - Corpus = a representative sample of target language stored as an electronic database (plural = "corpora")
- It relies on computer software for analysis
  - Results are generated using automatic and interactive techniques
- It depends on both *quantitative* and *qualitative* analytical techniques
  - Observations are counted and results are interpreted

Biber, Conrad, and Reppen (1998)
Current state of corpus linguistics tools: From principled corpora to opportunistic corpora

- Google
- Corpus of Global Web-based English (GloWbE)
- The Oxford English Corpus
- Cambridge English Corpus
- British National Corpus

**Brown Corpus Sample**
A01 0010 The Fulton County Grand Jury said Friday an investigation
A01 0020 of Atlanta's recent primary election produced "no evidence" that
A01 0030 any irregularities took place. The jury further said in term-end
A01 0040 presentments that the City Executive Committee, which had over-all
A01 0050 charge of the election, "deserves the praise and thanks of the
A01 0060 City of Atlanta" for the manner in which the election was conducted.
A01 0070 The September-October term jury had been charged by Fulton
A01 0080 Superior Court Judge Durwood Pye to investigate reports of possible
A01 0090 "irregularities" in the hard-fought primary which was won by
A01 0100 Mayor-nominate Ivan Allen Jr. "Only a relative handful
Current state of corpus linguistics tools:
Four generations of corpus tools (McEnery & Hardie, 2012)

- 1st-generation (1960s-1970s)
  - run on mainframes, single function tools, ‘monolingual’ (ASCII-based), designed for tiny corpora (of the time)
    - e.g., A Concordance Generator (Smith, 1966)
    - e.g., Discon (Clark, 1966)
    - e.g., Drexel Concordance Program (Price, 1966)
    - e.g., Concordance (Dearing, 1966)
    - e.g., CLOC (Reed, 1978)
Current state of corpus linguistics tools:
Four generations of corpus tools (McEnery & Hardie, 2012)

- 2nd-generation (1980s-1990s)
  - run on PCs, Roman-script language support, limited functions, designed for ‘small’ corpora
  - e.g., *Oxford Concordance Program (OCP)* (Hockey, 1988)
  - e.g., *Longman Mini-Concordancer* (Chandler, 1989)
  - e.g., *Kaye concordancer* (Kaye, 1990)
  - e.g., *MicroConcord* (Scott & Johns, 1993)
Current state of corpus linguistics tools:
Four generations of corpus tools (McEnery & Hardie, 2012)

- 3rd-generation (2000s-present)
  - run on PCs, partial (or full) Unicode support, more functions, designed for ‘bigger’ corpora, more statistical measures, easy-to-use
  - e.g., *WordSmith Tools* (Scott, 1996-2014)
  - e.g., *MonoConc Pro* (Barlow, 2000)
  - e.g., *AntConc* (Anthony, 2004-2014)
Current state of corpus linguistics tools:
Four generations of corpus tools (McEnery & Hardie, 2012)

- 4th-generation (late 2000s-present)
  - run on a server (accessed via a browser), partial (or full) Unicode support, simple (or advanced) functions, designed for pre-installed (copyrighted) corpora, simple to advanced statistical measures, easy-to-use
  - e.g., corpus.byu.edu (Davies, 2011), CQPweb (Hardie, 2011), SketchEngine (Kilgariff, 2011), Wmatrix (Rayson, 2011)
Current state of corpus linguistics tools:
Most popular tools for analyzing corpora

"Which computer programs do you use for analysing corpora?"
International survey of corpus linguists. Responses: 891. (Tribble, 2012)
Current state of corpus linguistics tools:
Most popular tools for analyzing corpora

Download Statistics for AntConc (2004-2014)
The current state of corpus linguistics tools
a need for something new
Current state of corpus linguistics tools: A need for something new

Corpus Development Process

1. Choose a target area of language use
2. Review the literature on language features
3. Search for pre-built corpora in target area
4. Design your own corpus (DIY)
5. Decide a sampling procedure
6. Collect, clean, tag, annotate, process, save

Empirical investigation
Current state of corpus linguistics tools:
A need for something new (exploratory tools)

WordWanderer
http://wordwanderer.org/

GraphColl
http://www.extremetomato.com/projects/graphcoll/
Current state of corpus linguistics tools:
A need for something new (collecting, cleaning, tagging tools)

NotePad++
https://notepad-plus-plus.org/

BulkFileRenamer
http://www.bulkrenameutility.co.uk/Main_Intro.php
Current state of corpus linguistics tools:
A need for something new

AntLab Tools
www.laurenceanthony.net/software
Current state of corpus linguistics tools: A need for something new (exploratory tools)

ProtAnt
http://www.laurenceanthony.net/software/protant/

FireAnt
http://www.laurenceanthony.net/software/fireant/
Current state of corpus linguistics tools: A need for something new (collecting, cleaning, tagging tools)

EncodeAnt
http://www.laurenceanthony.net/software/encodeant/

TagAnt
http://www.laurenceanthony.net/software/tagant/
Current state of corpus linguistics tools: The power of R (and Python, Java, ...)

Text Visualization Browser
A Visual Survey of Text Visualization Techniques
Provided by ISOVIS group

Techniques displayed: 170
Search:
Time filter: 1976 - 2015

Analytic Tasks
Visualization Tasks

Text Visualization Browser
http://textvis.lnu.se/
In a search for ditransitive constructions...

“As a heuristic, we used a script (less than 60 lines) that recovered all verb tokens tagged as used ditransitively in the ICE-GB, looked up the lemmas for these tokens in a lemma list, looked up all the forms for these lemmas in the lemma list... and then outputted a concordance of all matches of those forms in the learner corpus”

“This is not perfect, but it is easy to see that no ready-made program could ever do this (especially not quickly)”

(Gries, 2011: 93-94)
DIY Corpus Tools – The Debate
Arguments for learning to program
DIY Corpus Tools – The Debate:
Arguments for learning to program

- Corpus linguists should learn to program...
- (Biber, Weisser, Gries, Davies, ....)
If you program ...

"you can do analyses not possible with concordancers ... you can do analyses more quickly and more accurately ... you can tailor the output to fit your own research needs ... you can analyze a corpus of any size"

(Biber et al., 1998, p. 256)
"when you use pre-configured corpus programs, you’re a little bit at the mercy of the company or individual selling them ...

One final big advantage of programming languages, therefore, is that you are in the driver’s seat."

(Gries, 2009, p. 11-12)
“Every corpus-linguistic researcher should have some programming skills”

Reason 1: software many people use is severely limited in terms of
- availability (OS, cost)
- functionality (only what is hardwired)
- user-control (at the mercy of developers)

Reason 2: “inflexible software creates inflexible researchers”

(Gries, 2011: 92-94)
“Here, I think, we must divide researchers into two camps – corpus users and corpus creators. Corpus users can often get by with stand-alone tools or web-based corpora...For corpus creators, however, I would say that some experience with programming is a necessity.”

(Davies, 2011: 77)
A caveat...

For corpus users...
“even a little knowledge of regular expressions will go a long way in helping with more complex queries.”

For corpus creators (in more simple cases)...
“perhaps regular expressions and a simple knowledge of semi-automated file handling would be sufficient.”

(Davies, 2011: 77-78)
DIY Corpus Tools – The Debate:
Arguments for learning to program

- Where to start with programming...
  - Pick a popular language
    - My languages of choice
      - Python (standalone tools), JavaScript/PHP (web programming)
    - Suggestions for learning programming (in order of preference)
      - Scratch (BYOB), Python, Java
    - Suggestions for learning corpus linguistics programming (in order of preference)
      - R, Python, JavaScript (PHP?, Java??, Perl???, Pascal????)
  - Read a programming book or online tutorial (or join a MOOC)
    - e.g. Teach Yourself Perl 5 in 21 Days
    - e.g. Learn Python The Hard Way (http://learnpythonthehardway.org/book/)
  - Join the one truly amazing programming forum
    - Stack Overflow (http://stackoverflow.com/)
DIY Corpus Tools – The Debate:
Arguments for learning to program

- The Scratch programming interface
DIY Corpus Tools – The Debate:
Arguments for learning to program

- A (Brown Corpus) word list tool in Python...

```python
import nltk
import collections
types = collections.Counter()

from nltk.corpus import brown
for word in brown.words():
    types[word] += 1
```

- A (Brown Corpus) KWIC Concordancer in Python...

```python
import nltk
from nltk.corpus import brown
for id, type in enumerate(brown.words()):
    if type == 'politician':
        print(' '.join(brown.words()[id - 5:id + 6]))
```
DIY Corpus Tools – The Debate
Arguments against learning to program
DIY Corpus Tools – The Debate: Arguments against learning to program

**Argument 1:**
- Most corpus linguists are corpus users (not corpus creators)
  - [We can ‘get by’ with current corpus tools]

**Rebuttal**
- Using ready-built tools ‘imprisons’ the corpus linguist preventing them from developing new methods, analyzing interesting data sets, and deriving novel interpretations of that data
  - “AntConc’s availability only in compiled form makes running it problematic” (AntConc user Feb. 10, 2016)
  - “Many ‘stand-alone’ programs to analyze corpora are not scalable enough to handle new, ‘super-sized’ corpora (Davies, 2011: 74)
  - “if the commercial software is not designed to produce the desired results, then the corpus linguist without programming experience either has to live with a potentially foul compromise or drop the project” (Gries, 2011: 94)
DIY Corpus Tools – The Debate: Arguments against learning to program

Argument 2:

- Researchers in many fields do not develop their own tools
  - e.g. astronomers, biologists, (doctors), ...

- Issac Newton (Optical) Telescope
- The Hubble (Optical) Telescope
- The Fermi Gamma-ray Space Telescope
- Jodrell Bank Radio Telescope
DIY Corpus Tools – The Debate:
Arguments against learning to program

- "How are new tools developed for astronomy?"

Wide-Field Infrared Explorer

James Webb Space Telescope

Professor Jim Wild, Lancaster University
Vice-President, Royal Astronomical Society
Arguments against learning to program

- "How are new tools developed for astronomy?"
  - Collaborations between astronomers and engineers
  - Massive funding for tool development
    - STFC (Science and Technology Facilities Council) national laboratories
      - ISIS, Diamond, Central Laser Facility
    - STFC technology centers
    - Specialist laboratories
      - Rutherford Appleton Laboratory
      - Space Magnetometer Laboratory (Imperial College)
      - Particle sensors laboratory (University College London)
Arguments against learning to program

- Why should we be hermits trying to develop corpus tools on our own?
DIY Corpus Tools – The Debate: Arguments against learning to program

- **Arts and Law**
  - English, Drama and American & Canadian Studies; History and Cultures; Languages, Cultures, Art History and Music; Birmingham Law School; Philosophy, Theology and Religion

- **Engineering and Physical Sciences**
  - Chemistry; Chemical Engineering; Civil Engineering; Computer Science; Electronic, Electrical and Computer Engineering; Mathematics; Mechanical Engineering; Metallurgy and Materials; Physics and Astronomy

- **Life and Environmental Sciences**
  - Biosciences; Geography, Earth and Environmental Sciences; Psychology; Sport and Exercise Sciences

- **Medical and Dental Sciences**
  - Cancer Sciences; Clinical and Experimental Medicine; Dentistry; Health and Population Sciences; Immunity and Infection

- **Social Sciences**
  - Birmingham Business School; Education; Government and Society; Social Policy

- **Liberal Arts and Sciences**
DIY Corpus Tools – The Debate:
Arguments against learning to program

- Argument 3:

  The reality for most corpus researchers, however, is that computer programming is in a completely different world ... without extensive training in programming ... it is likely that these [DIY] tools would be more restrictive, slower, less accurate and only work with small corpora.

  (Anthony, 2009, p. 95)
DIY Corpus Tools – The Debate:
Arguments against learning to program

- Where to start when using standard tools...
  - Decide your research question before selecting your tool/method

"Research should be led by the science not the tool."

Professor Jim Wild, Lancaster University
Vice-President, Royal Astronomical Society
DIY Corpus Tools – The Debate: Arguments against learning to program

- Where to start when using standard tools...
  - Decide your research question before selecting your tool/method
  - Learn to use a “good” text editor: Notepad++ (Win), TextWrangler (Mac)
    - Unicode support (reading/converting text encodings)
    - Batch file handling
    - Regular Expressions (Regex) search/replace
  - Read the user guide of your chosen tool
    - Can it handle Unicode data?
    - Can it perform Regular Expression (Regex) searches?
    - Can it output results that you can feed into other tools (e.g. Excel/SPSS)?
  - Be proactive in contacting software developers
    - Explain clearly want you want to do (not how you think you should do it)
    - Provide motivation for them to get involved (what will they get out of it?)
    - Treat them as part of the team (not just a technical staff member)
“Regardless of the project or the resources being used, researchers should attempt to understand (a) the limitations of the tools they are using and (b) what the alternatives are.”

(Davies, 2011: 77)
Thoughts on the future of programming and tools in corpus linguistics research

programming, tools, and statistics, project teams
Thoughts on the future of programming and tools in corpus linguistics research:

- Programming, tools, and statistics

  "We’re statisticians. We don’t program."
  — Anonymous statistician
  https://flowingdata.com/2011/10/18/statisticians-dont-program/

- Statistics has evolved enormously through the development of software (and hardware) tools
- But... not all statisticians are programmers
- Should statisticians program?

Andy C — October 19, 2011 at 2:21 pm
Clearly, the future belongs to analysts, data scientists and statisticians who code. I couldn’t imagine taking anyone on who couldn’t analyse _and_ code.
Thoughts on the future of programming and tools in corpus linguistics research:

- From hermit to team player...

  e.g. xyz
Thoughts on the future of programming and tools in corpus linguistics research:

- From team player to ...

  e.g. Sinclair

http://www.hermitary.com/archives/vaneyck.jpg

https://upload.wikimedia.org/wikipedia/commons/d/d6/St.-Jerome-In-His-Study.jpg
Conclusions

- Corpus linguistics research is rapidly changing in terms of corpus size, design, and applications
  - Many interesting corpus linguistics problems can only be solved with new and interesting tools
- Many corpus linguists struggle to collect, clean, tag, annotate and analyze their corpus in new and interesting ways
  - Developing a generation of corpus linguistics who understand basic text handling and processing is essential
- Future corpus tools development and research designs can be improved (most rapidly) through researcher interaction within and across disciplines
  - Creating successful project teams will need infrastructure and financial support by institutions, societies, and funding agencies