Natural Language Processing (2025), **31**, pp. 176–180
doi:10.1017/nlp.2024.8

CAMBRIDGE UNIVERSITY PRESS

Book Review

***Python for Linguists*, by Michael Hammond. Cambridge: Cambridge University Press, 2020. ISBN: 978-1-108-49344-4 (hb), 978-1-108-73707-4 (pb), 978-1-108-63898-2 (e-book), xii + 300 pages.** https://doi.org/10.1017/9781108642408

"Computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity,
and especially because it produces objects of beauty."
Donald E. Knuth, 1974

In a broad sense, this book review focuses on a volume that aims to explicate the main rationale of computational thinking using Python, primarily targeting readers with a background in linguistics. The book's core content is structured across 10 self-contained chapters (chs. 2–11), featuring an introduction to environments for interacting with the language, an appendix on the `NLTK package` (or *Natural Language Toolkit*) and an index with the terms introduced throughout the contribution. Each chapter also includes a gamut of exercises (of varying degrees of difficulty) allowing readers to practise the theoretical concepts brought in thus far. This book provides a further contribution to a set of textbooks that Hammond has put to press over the years for language scientists seeking practical computing knowledge. This collection includes titles on Java technology (Hammond, 2002) and Perl (Hammond, 2003). These programming languages, alongside Prolog (see, for instance, Nugues, 2014), C, Haskell or Lisp, have been extensively used by professionals and researchers in the field of Natural Language Processing (henceforth NLP), until the advent of Python some decades ago. Probably one of the reasons why this publication is eagerly anticipated by those unfamiliar with this coding language is its reputation as an "extremely powerful and versatile [language] while at the same time also easily understood and learned" (Lin *et al.*, 2022, p. 5).

The volume begins with an introductory exploration of important questions, such as 'Why is Python a good choice for linguistics?' or 'Why do linguists need to learn how to program?'. It also includes some remarks pinpointing the unique characteristics (the distinctive feature lies in its focus on readers with a linguistic expertise) that set this volume apart from others currently available in the market, as well as its distinctions from existing Python resources. Chapter 1, "Interacting with Python and basic functions", surveys different ways of using Python and how to invoke it. It could have been useful to include some information on how to install Python, particularly for readers working with Windows. The author primarily works with the interactive environment (i.e., a terminal window running a Python interpreter). This chapter also offers some guidelines for fundamental interactions with the language, and concludes with concise instructions on editing and running programs from the command line. The subsequent two chapters are of paramount importance for any *ab initio* learner of programming, as they cover the creation of variables as well as the use of control structures to manipulate them. Both variable assignment and control structures constitute, in Hammond's view, "the heart and soul of programming in Python" (p. 54).

Chapter 2, "Data types and variables", begins with some hints on naming variables and assigning values to them. The bulk of the chapter revolves around introducing fundamental data types and structures[a] and, more importantly, some built-in methods and functions that Python has already implemented to manipulate them. The last part delves into the concept of *mutability*, which broadly refers to the property of some data types being alterable after creation. Chapter 3, "Control structures", introduces Python's essential control structures, which are the *raison d'être* of "almost everything you need to know to write meaningful and useful programs" (p. 29). After a brief detour on the relevance of code grouping, the discussion shifts to conditional statements (`if`) and loops (`for` and `while`). The chapter also includes some observations on printing expressions, as well as loop control statements (`break` and `continue`) whose ultimate function is to alter execution from its default order. It culminates with a practical example illustrating how to create diverse syllables and sentences, particularly useful for generating stimuli in psycholinguistic research.

In Chapter 4, "Input-output", the author transitions to various mechanisms for populating programs with data from a range of sources. Five methods of inputting data are discussed in the book: *command-line*, *standard input*, *keyboard input*, *file input-output*, and *web*. The chapter concludes with a meticulously crafted program accompanied by comprehensive comments on conducting lexical statistics (i.e., correlation between word length and word frequency) using Lewis Carroll's *Alice's adventures in Wonderland (Alice* for short) as a running example, and guiding readers through each step. This excerpt is sourced from Project Gutenberg. Chapter 5, "Subroutines and modules", proffers some techniques to break down lengthy and repetitive programs into more straightforward and easily manageable code snippets, which is achieved by utilising functions and modules. Various types of functions are introduced progressively (i.e., simple functions, functions that return values, functions that take arguments, and recursive functions), before embarking on the creation of modules as a collection of functions and objects that can be shared across multiple programs. The chapter also explores the notion of *recursivity* (as applied to functions), which is essential for tasks like creating a phrase structure grammar. Additionally, the use of the `lambda` keyword to define unnamed functions is covered and further examined in Chapter 11. This contribution finishes with insights into code documentation via docstrings and comments, along with a brief demonstration of sentence analysis in *Alice*.

In Chapter 6, "Regular expressions", the author dedicates the entire section to regular expressions (for pattern matching), a crucial and valuable aspect of programming in linguistic research. The greater part of the chapter focuses on the functionalities of the `re` module and its related functions, including `search()`, `group()`, `start()`, `end()`, `span()` and `findall()`. The syntax associated with specifying patterns—such as *union*, *concatenation* and *Kleene star*[b]—is thoroughly explored. Notably, the text explores the intricacies of backreferences, a significant improvement with respect to conventional regular expressions. As the book describes, backreferences allow you to refer back to previously used parentheses in a pattern. The chapter also includes a case study demonstrating how regular expressions can be employed to identify word-initial consonant clusters in *Alice*.

Chapter 7, titled "Text manipulation", examines the tools that can be used for transforming "one string of letters into another" (p. 138). Given that string manipulation can involve a gamut of costly computational operations (depending on the specific objective), this section

[a]There are other data types, such as complex numbers, that are mentioned in passing, since they are of no particular interest for linguistic endeavours, at least at this stage.

[b]Notice that the use of Kleene star here departs sharply from the treatment typically encountered in the literature on syntax (to mark ungrammatical sentences) and historical linguistics (to showcase unattested forms in historical records). In the A* notation the device as it is used here alludes to a sequence of multiple items, all of which belong to the category A. It seems that "[t]he notation is borrowed from the mathematics of formal systems, in which it normally means a string of zero or more such categories, but many linguists use the notation to express a string of one or more such categories, which in mathematics would be expressed as A^{+}, since the notion 'zero or more' is only occasionally of linguistic use" (*DGTL*, *s.v. Kleene star*) (DGTL = Trask 1991).

https://doi.org/10.1017/nlp.2024.8 Published online by Cambridge University Press

introduces several libraries and resources such as `re.sub()`, `str.translate()`, `re.split()` and the `join()` method. While the majority of the chapter focuses on the fundamentals of Porter's stemming algorithm—a technique for extracting suffixes from (specific) stems, such as *running* (<*run*) or *obfuscation* (<*obfuscate*)—Hammond showcases its potential through illustrative use cases. Even though the instances provided are self-evident in most cases, some comments on the output of each function could have been useful to accompany the fragments of code. Not only does this case study explore the possibilities of text manipulation, but it also investigates concepts inducted from previous chapters.

Chapter 8, "Internet data", delves into the automatic retrieval of information from websites by means of the `urllib.request` module. Since this process requires knowing the format and encoding of web pages, some sections are entirely devoted to introducing the basic tenets of HTML using the BeautifulSoup (`bs4`) module, and encoding systems such as ASCII, Unicode, UTF-16 and, particularly, UTF-8 (the one that Python reads). This chapter also prefaces the utility of computer parallelism, by means of the `multiprocessing` module, when fetching data from several web pages all at once, and is rounded off with the implementation of a web crawler to collect texts in Welsh and save their content. It could perhaps have also included a discussion of the differences between *web scraping* and *web crawling*, however. The next three chapters explore different programming paradigms from the *procedural programming* one used until now.

Chapter 9, "Objects", introduces the logic of *object-oriented programming*, as opposed to sequences of commands. This coding approach enables the structured organisation of programs into classes and objects in a clearer fashion. It commences with the introduction and (ample) exemplification of key terms like *class*, *object* (or *instance of a class*), *class method*, *instance method*, *class variable* or *instance variable*. All this paves the way for exploring the nitty-gritty of the concept of *inheritance*. The chapter closes with a useful OO-driven case study on syllabification, where all programming strategies introduced thus far are employed.

In Chapter 10, "GUIs", Hammond presents *event-driven programming* using *graphical user interfaces* (GUIs) with `tkinter` (i.e., Python's built-in GUI toolkit). GUIs permit the user to interact with Python through input modes and, therefore, the major part of this contribution revolves around the presentation of *widgets* (or GUI elements) such as labels, buttons or pop-up menus. Not only does this chapter focus on their implementation, but also digs into the different packing options to organise widgets in a window; modify things like colour, spacing and font; display messages; and insert text. A case study implements Porter's stemmer (see ch. 7) using GUI programming. Finally, Chapter 11 on "Functional programming" elucidates the ideal governing *functional programming* (an approach rooted in lambda calculus). Despite its limitations, it advocates practices like avoiding global variables, immutable variable usage and embracing functions as arguments, often employing recursion and comprehensions to replace control structures. The appendix on the `nltk` module includes some important information on how to use some of the corpora available, how to tokenize a stretch of text, how to eliminate function words (i.e., stopwords), and how to part-of-speech tag a text.

All in all, Hammond, a renowned expert in (theoretical) morphology and prosodic phonology, has crafted an excellent introductory textbook on programming. This high-calibre resource enables language scientists to grasp the fundamental principles of Python code for managing basic language processing tasks. After absorbing and practising the content, readers will gain a solid command of the Python programming language. This competence should suffice for tackling typical linguistic daily challenges. Furthermore, it may empower readers to delve into more advanced Python-related topics (see Roth and Wiegand, 2021, p. 220).

Among the various avenues for further study opened up by this volume, it heightens awareness of the intricacies of the swiftly evolving field of NLP[c] (see, e.g., Allen (1995), Bird *et al.*

[c]Attempting to provide an *au courant* overview of NLP here would far exceed the goals set for a publication of this nature. There are, however, some useful references where the reader can find updated accounts of the field (*vid.* Agerri *et al.* (2021), probably the fullest available, or Orăsan & Mitkov (2022).

https://doi.org/10.1017/nlp.2024.8 Published online by Cambridge University Press

(2009), Eisenstein (2019), Jurafsky and Martin (2009), Manning and Schütze (1999) or Zhang and Teng (2021)). Another option is delving into cutting-edge algorithmic research, exemplified by the highly acclaimed volumes of Knuth's *The art of computer programming*, arguably the most comprehensive theoretical account of computer programming.[d] It also complements other books on programming that have been specifically published for linguists in the last few years, particularly Mason (2000), Hammond (2002, 2003) or Weisser (2009)[e].

The inclusion of exercises at the end of each chapter is one of the main strengths of the book since this affords potential readers to run through programming skills applied to linguistic problems. Although most of these exercises are of use, it would also have been profitable to cogitate about (mini-)projects directly linked to the computational processes inherent in the implementation of certain constructions attested, for example, in currently available grammars. One of these includes *The Cambridge grammar of the English language* (Huddleston, Pullum *et al*., 2002) (*cf*. Brew, 2003), arguably the fullest account of English grammar produced to date. It is also extremely useful to have all the material (including source code) covered in the textbook publicly accessible in the companion website[f], although there are no solutions available.

Acknowledgements. We are indebted to Idoia Berges, Juanan Pereira, Olatz Perez-de-Viñaspre and the book review editor of *Natural Language Processing* (previously known as *Natural Language Engineering)*, Diana Maynard, for useful comments and suggestions on a previous draft. Usual disclaimers apply.

Author contributions. Each author has made an equitable contribution to the composition of this book review.

Competing interest. The authors hereby affirm that there are no competing interest pertaining to this manuscript.

Pablo M. Tagarro
University of the Basque Country (UPV/EHU),
Donostia/San Sebastián, Spain
pablobec93@hotmail.es

Igor Rodriguez
RSAIT, University of the Basque Country (UPV/EHU),
Donostia/San Sebastián, Spain
igor.rodriguez@ehu.eus

Maite Oronoz
HiTZ Center - Ixa, University of the Basque Country (UPV/EHU),
Donostia/San Sebastián, Spain
maite.oronoz@ehu.eus

References

Agerri, **R.**, **Agirre**, **E.**, **Aldabe**, **I.**, **Aranberri**, **N.**, **Arriola**, **J. M.**, **Atutxa**, **A.**, **Azkune**, **G.**, **Casillas**, **A.**, **Estarrona**, **A.**, **Farwell**, **A.**, **Goenaga**, **I.**, **Goikoetxea**, **J.**, **Gojenola**, **K.**, **Hernaez**, **I.**, **Iruskieta**, **M.**, **Labaka**, **G.**, **Lopez-de-Lacalle**, **O.**, **Navas**, **E.**, **Oronoz**, **M.**, **Otegi**, **A.**, **Pérez**, **A.**, **Perez-de-Viñaspre**, **O.**, **Rigau**, **G.**, **Sanchez**, **J.**, **Saratxaga**, **I. and Soroa**, **A.** (2021). Report on the state of the art in language technology and language-centric AI. *European Language Equality,* D1.2: 1–65.

Allen, **J.** (1995). *Natural Language Understanding*. 2nd Edn. San Francisco, CA: Benjamin Cummings.

Baayen, **R. H.** (2008). *Analyzing Linguistic Data: A Practical Introduction to Statistics Using R*. Cambridge: Cambridge UP.

[d]Their contents, however, might be intimidating at first sight given the amount of mathematical knowledge and logical thinking presupposed, and additional understanding from related areas of computer science might be required.

[e]It should be noted that there are currently available other books on R, SPSS and JASP whose main objective, however, is the acquisition of statistical skills (i.e., collection, analysis and interpretation of data) for carrying out research in linguistics. Some of these include Baayen (2008), Eddington (2015), Levshina (2015), Gries (2017, 2021), Desagulier (2017), Winter (2019) or Loerts *et al*. (2020).

[f]https://hammondm.github.io/.

https://doi.org/10.1017/nlp.2024.8 Published online by Cambridge University Press

Bird, **S.**, **Klein**, **E. and Loper**, **E.** (2009). *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. Sebastopol, CA: O'Reilly.

Brew, **C.** (2003). Review of the *Cambridge grammar of the English language* by Rodney Huddleston & Geoffrey K. Pullum et al. *Computational Linguistics* **29**(1): 144–147.

Desagulier, **G.** (2017). *Corpus Linguistics and Statistics with R: Introduction to Quantitative Methods in Linguistics*. Cham: Springer.

DGTL = Trask, **R. L.** (1991). A dictionary of grammatical terms in linguistics. London: Routledge.

Eddington, **D.** (2015). *Statistics for Linguists: A Step-by-Step Guide for Novices*. Newcastle upon Tyne: Cambridge Scholars Publishing.

Eisenstein, **J.** (2019). *Introduction to Natural Language Processing*. Cambridge, MA: The MIT Press.

Gries, **S. Th.** (2017). *Quantitative Corpus Linguistics with R: A Practical Introduction*. 2nd Edn. London: Routledge.

Gries, **S. Th.** (2021). *Statistics for Linguistics with R: A Practical Introduction*. 3rd Edn. Berlin: De Gruyter.

Hammond, **M.** (2002). *Programming for Linguists: JavaTM Technology for Language Researchers*. Oxford: Blackwell.

Hammond, **M.** (2003). *Programming for Linguists: Perl for Language Researchers*. Oxford: Blackwell.

Huddleston, **R. and Pullum**, **G. K.** (in collaboration with L. Bauer, B. Birner, T. Briscoe, P. Collins, D. Denison, D. Lee, A. Mittwoch, G. Nunberg, F. Palmer, J. Payne, P. Peterson, L. Stirling and G. Ward). (2002). *The Cambridge Grammar of the English Language*. Cambridge: Cambridge University Press.

Jurafsky, **D. and Martin**, **J. H.** (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. 2nd Edn. Englewood Cliffs, NJ: Prentice-Hall. (A draft of the third edition, which is periodically being updated, is currently available in Jurafsky's website).

Levshina, **N.** (2015). *How to Do Linguistics with R: Data Exploration and Statistical Analysis*. Amsterdam: John Benjamins.

Lin, **J. W. B.**, **Aizenman**, **H.**, **Espinel**, **E. M. C.**, **Gunnerson**, **K. and Liu**, **J.** (2022). *An Introduction to Python Programming for Scientists and Engineers*. Cambridge: Cambridge University Press.

Loerts, **H.**, **Lowie**, **W. and Seton**, **B.** (2020). *Essential Statistics for Applied Linguistics using R or JASP*. 2nd Edn. London: Palgrave Macmillan.

Manning, **C. D. and Schütze**, **H.** (1999). *Foundations of Statistical Natural Language Processing*. Cambridge, MA: The MIT Press.

Mason, **O.** (2000). *Programming for Corpus Linguistics: How to Do Text Analysis with Java*. Edinburgh: Edinburgh University Press.

Nugues, **P. M.** (2014). *Language Processing with Perl and Prolog: Theories, Implementation, and Application*. 2nd Edn. Berlin: Springer.

Orăsan, **C. and Mitkov**, **R.** (2022). Recent developments in natural language processing. In R. Mitkov (ed.), *The Oxford Handbook of Computational Linguistics*. Oxford: Oxford University Press, pp. 1198–1241.

Roth, **B. and Wiegand**, **M.** (2021). Review of *Python for Linguists* by Michael Hammond. *Computational Linguistics* **47**(1): 217–220.

Weisser, **M.** (2009). *Essential Programming for Linguistics*. Edinburgh: Edinburgh University Press.

Winter, **B.** (2019). *Statistics for Linguists: An Introduction using R*. London: Routledge.

Zhang, **Y. and Teng**, **Z.** (2021). *Natural Language Processing: A Machine Learning Perspective*. Cambridge: Cambridge University Press.

https://doi.org/10.1017/nlp.2024.8 Published online by Cambridge University Press