Computational Math / Science

A short report on the course: “Introduction to Programming with Scientific Applications”

Gerth Stølting Brodal

Department of Computer Science

Computational Thinking Seminar, Center for Computational Thinking and Design (CCTD), Aarhus University, May 16, 2018

Background

- Summer 2017 major Study Reform at Science and Technology @ AU:
  - 4 quarters replaced by 2 semesters (5 ECTS → 10 ECTS courses)
- Previously many non-computer science (CS) students were required to follow the CS introduction to programming course in Java
  - need expressed by other educations to have a more specifically target course
- Together with mathematics department (Niels Lauritzen) defined new programming course, targeted towards math students
  - Python
  - Some project work / applications
  - Dynamic programming (+ basic sorting and binary search)
  - Basic understanding of differences/similarities between Python and Java

Course content

Basic programming
Advanced / specific python
Libraries & applications

1. Introduction to Python	10. Functions as objects	19. Linear programming
2. Python basics / if	11. Object oriented programming	20. Generators, iterators, with
3. Basic operations	12. Class hierarchies	21. Modules and packages
4. Lists / while / for	13. Exceptions and files	22. Working with text
5. Tuples / comprehensions	14. Doc, testing, debugging	23. Relational data
6. Dictionaries and sets	15. Decorators	24. Clustering
7. Functions	16. Dynamic programming	25. Graphical user interfaces (GUI)
8. Recursion	17. Visualization and optimization	26. Java vs Python
9. Recursion and Iteration	18. Multi-dimensional data	27. Final lecture

27 lectures (2 x 45 min) + 14 exercise sessions (3 hours) + 5 hours studie café / week + 10 handins + PeerWise + MentiMeter + 1 final project (1 month, 25% of grade) + MCQ exam (75%, 2 hours)

https://blackboard.au.dk/webapps/blackboard/content/listContentEditable.jsp?content_id=_1639577_1&course_id=_110424_1

Personal goal

At the end of the course the students should...

- master basic programming concepts
- know and have used more advanced programming features (recursive functions & data types, OO, λ, decorators)
- have basic knowledge of some common Python packages
  - numpy, matplotlib, pandas, tkinter, scipy, Jupyter, doctest, ...
- be able to navigate in the Python ecosystem

Population (realized one week before the course)

- Mathematics (25, 2nd year)
- Mathematics-Economics (26, 2nd year)
- Chemistry (22, elective, 3rd – 4th year)
- Minor in Mathematics (20, ~ 4th year) - primary user of the study café

Binomial Coefficient
Dynamic programming using decorator

Slide from course on dynamic programming

- Use a decorator (@memoize) that implements the functionality of remembering the results of previous function calls

bionomial_decorator.py

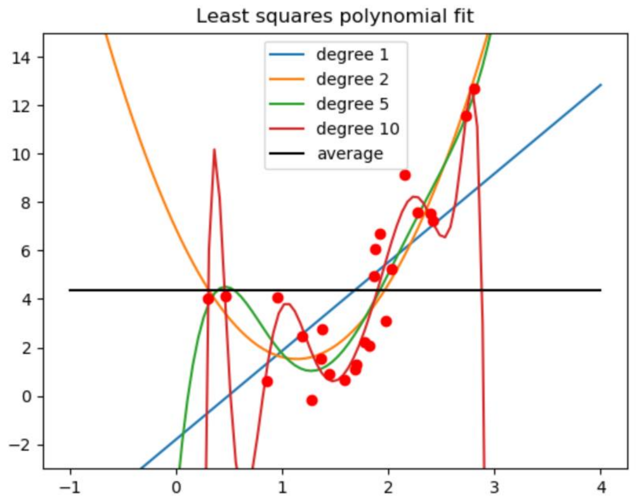
```
def memoize(f):
    # answers[args] = f(*args)
    answers = {}

    def wrapper(*args):
        if args not in answers:
            answers[args] = f(*args)
        return answers[args]

    return wrapper
```

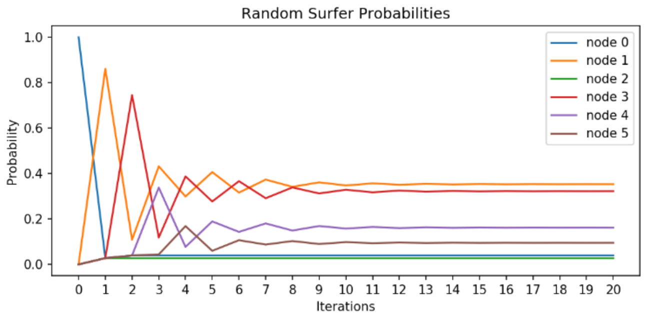
```
@memoize
def binomial(n, k):
    if k==0 or k==n:
        return 1
    else:
        return binomial(n-1, k) + binomial(n-1, k-1)
```

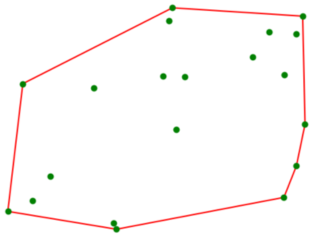
www.python-course.eu/python3_memoization.php

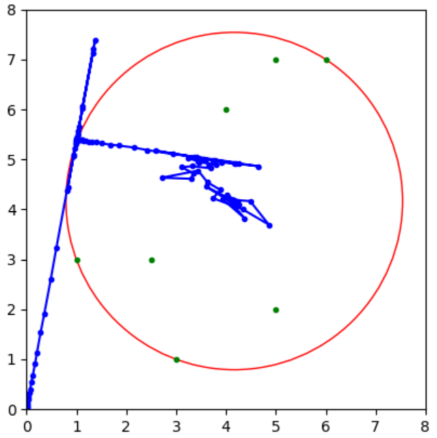
Math / scientific concepts covered

- Recursion recurring theme (recursive functions, recursive data types, recursive objects, recursive OO method calls, handins on comparing phylogenetic trees)
- Dynamic programming (recursion + decorator) and recurrences
- Plot of data (matplotlib.pyplot)
- Matrices and multidimensional data (numpy)
- Least squares fit (numpy.polyfit)
- Linear programming (scipy.optimize.linprog)
- Maximum flow problems (vha scipy.optimize.linprog)
- Eigenvector, PageRank (numpy.linalg.eig)
- Minimum of functions (scipy.optimize.minimize)
  - minimum enclosing circle, comparison with Matlab
- Jupyter notebooks



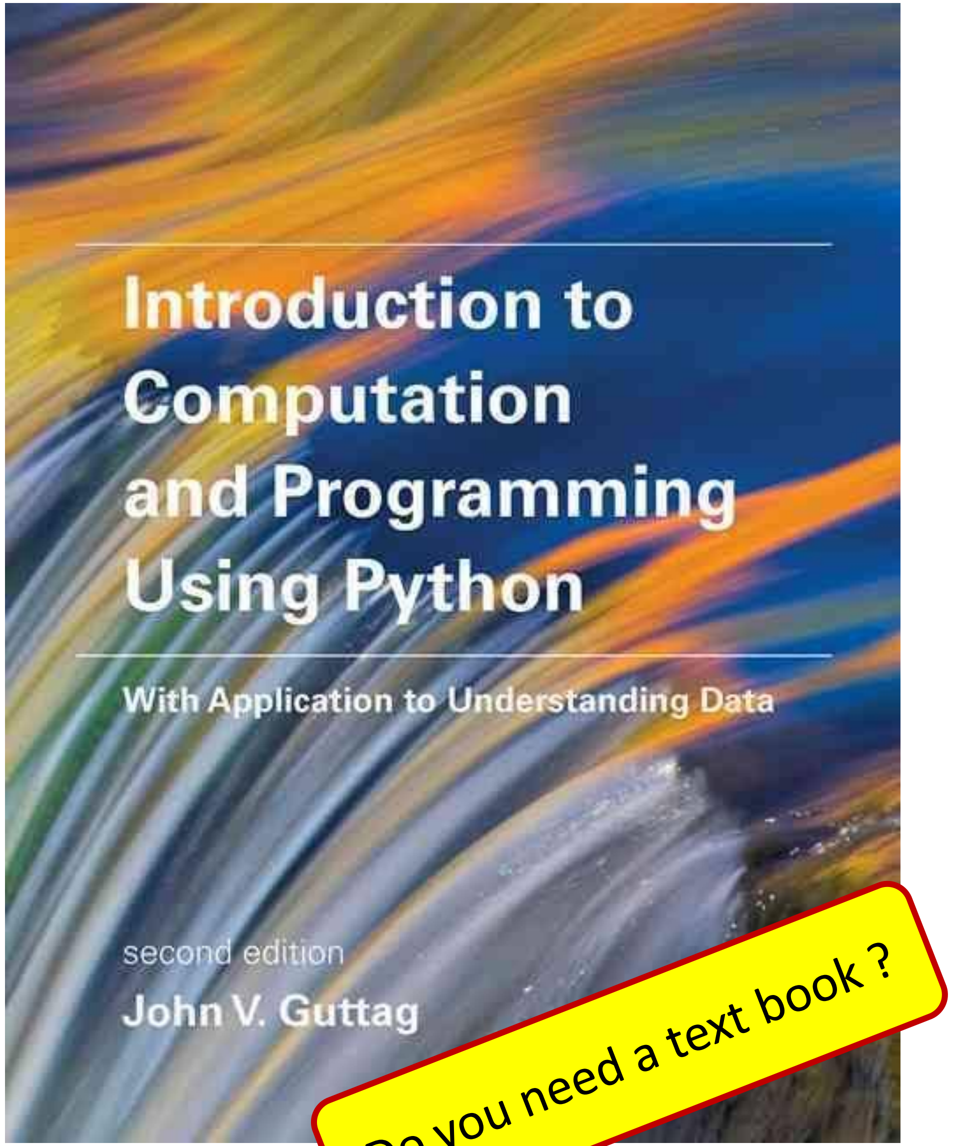






Course book

- Course book followed a little bit in random order – and only covered partially – but gives a good introduction to most important Python concept in a few pages and with many (perhaps too) mathematically oriented examples
- Primary course material are lecture slides (made available last minute...)
- A central competence for the students to acquire is to be able to Google relevant information (e.g. Python libraries)



Introduction to Computation and Programming Using Python

With Application to Understanding Data

second edition

John V. Guttag

Do you need a text book ?