Welcome to MCS 507

1. About the Course
   - content and organization
   - expectations of the course
2. SageMath, SciPy, Julia
   - the software system SageMath
   - Python's computational ecosystem
   - the programming language Julia
3. Language Agnostic Computing
   - the Jupyter notebook and JupyterLab

MCS 507 Lecture 1
Mathematical, Statistical and Scientific Software
Jan Verschelde, 21 August 2023

Welcome to MCS 507

1. About the Course
   - content and organization
   - expectations of the course
2. SageMath, SciPy, Julia
   - the software system SageMath
   - Python's computational ecosystem
   - the programming language Julia
3. Language Agnostic Computing
   - the Jupyter notebook and JupyterLab

Catalog Description

The design, analysis, and use of mathematical, statistical, and scientific software.

Prerequisite(s): the catalog lists “Grade of B or better in MCS 360 or the equivalent or consent of instructor.”
Examples of courses which could serve as “the equivalent”
are MCS 320 (introduction to symbolic computation)
and MCS 471 (numerical analysis).

- MCS 507 fits in an interdisciplinary computational science and engineering (CSE) curriculum.
- MCS 507 is on the Computational Science Prelim.
- MCS 507 prepares for MCS 572 (introduction to supercomputing).

Research Software and Software Research

The design, analysis, and use of mathematical, statistical, and scientific software.

In this course, *software is the object of our research.*

Examples of academic publication venues:

- The annual SciPy and Julia conferences (each summer).
- The International Congress on Mathematical Software has proceedings as a volume of Lectures Notes in Computer Science.
- ACM Transactions on Mathematical Software is a journal.
- Practice and Experience in Advanced Research Computing, the PEARC conference series is sponsored by the ACM.
- SIAM conference on Parallel Processing.
- SIAM conference on Computational Science & Engineering.

Content of the Course

books and core software

Two book sources:

- An introduction to computational science with Python is in *A Primer on Scientific Programming with Python*. Fifth Edition, Springer-Verlag, 2016, by Hans Petter Langtangen.
- Juan Nunez-Iglesias, Harriet Dashnow, and Stéfan van der Walt. *Elegant SciPy.* O'Reilly Media, 2017.

We consider

- SageMath bundles many free and open source software systems.
- Python's computational ecosystem: SciPy.
- Julia: a fresh approach to numerical computing.

a week-by-week list of topics

1. overview of the course and Python
2. numpy, sympy, scipy, matplotlib
3. arithmetic, roots, packaging
4. a fresh approach to numerical computing
5. parallel programming
6. cython, testing, documenting, packaging
7. graphical user interfaces, web interfaces, and web servers
8. solving ordinary differential equations, review for *midterm*
9. data retrieval, analysis, and machine learning
10. image processing, networks
11. computational geometry
12. statistics
13. groups, numbers, polynomials

Welcome to MCS 507

1. About the Course
   - content and organization
   - expectations of the course
2. SageMath, SciPy, Julia
   - the software system SageMath
   - Python's computational ecosystem
   - the programming language Julia
3. Language Agnostic Computing
   - the Jupyter notebook and JupyterLab

Organization and Expectations

Three parts in the course:

- Scientific scripting with Python and Julia.
- Introduction to SageMath and some of its components: Maxima, GAP, PARI/GP, R, Singular, etc.
- Software for your own research interests.

Activities throughout the semester:

- Several homework collections.
- The midterm exam could be take home.
- Three computer projects.

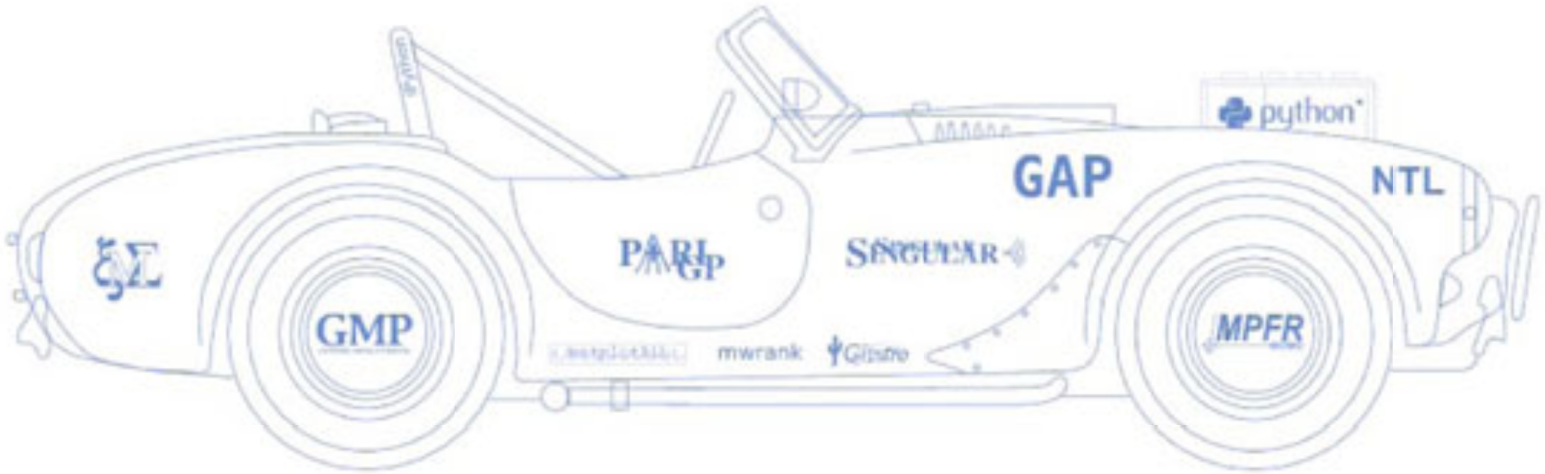
A final exam could be possible, but instead of a final exam, most likely the last lectures will be reserved for student project presentations.

Welcome to MCS 507

1. About the Course
   - content and organization
   - expectations of the course
2. SageMath, SciPy, Julia
   - the software system SageMath
   - Python's computational ecosystem
   - the programming language Julia
3. Language Agnostic Computing
   - the Jupyter notebook and JupyterLab

do not re-invent the wheel

metaphor: on building the car ...



B. Eröcal and W. Stein.

The Sage project: Unifying free mathematical software to create a viable alternative to Magma, Maple, Mathematica, and MATLAB. In K. Fukuda, J. van der Hoeven, M. Joswig, and N. Takayama, editors, *Mathematical Software - ICMS 2010*, vol. 6327 of *Lecture Notes in Computer Science*, pages 12–27. Springer, 2010.

SageMath

a free open source mathematical software system

Sage aims to be a viable free and open source alternative to the big M's (Magma, Mathematica, MATLAB, Maple).

Why Sage? At least four reasons:

1. bundles various leading software systems used for mathematical research and experimentation
2. Python as scripting language
3. active and large community of developers
4. Try it online! at `cocalc.com`

David Joyner and William Stein:

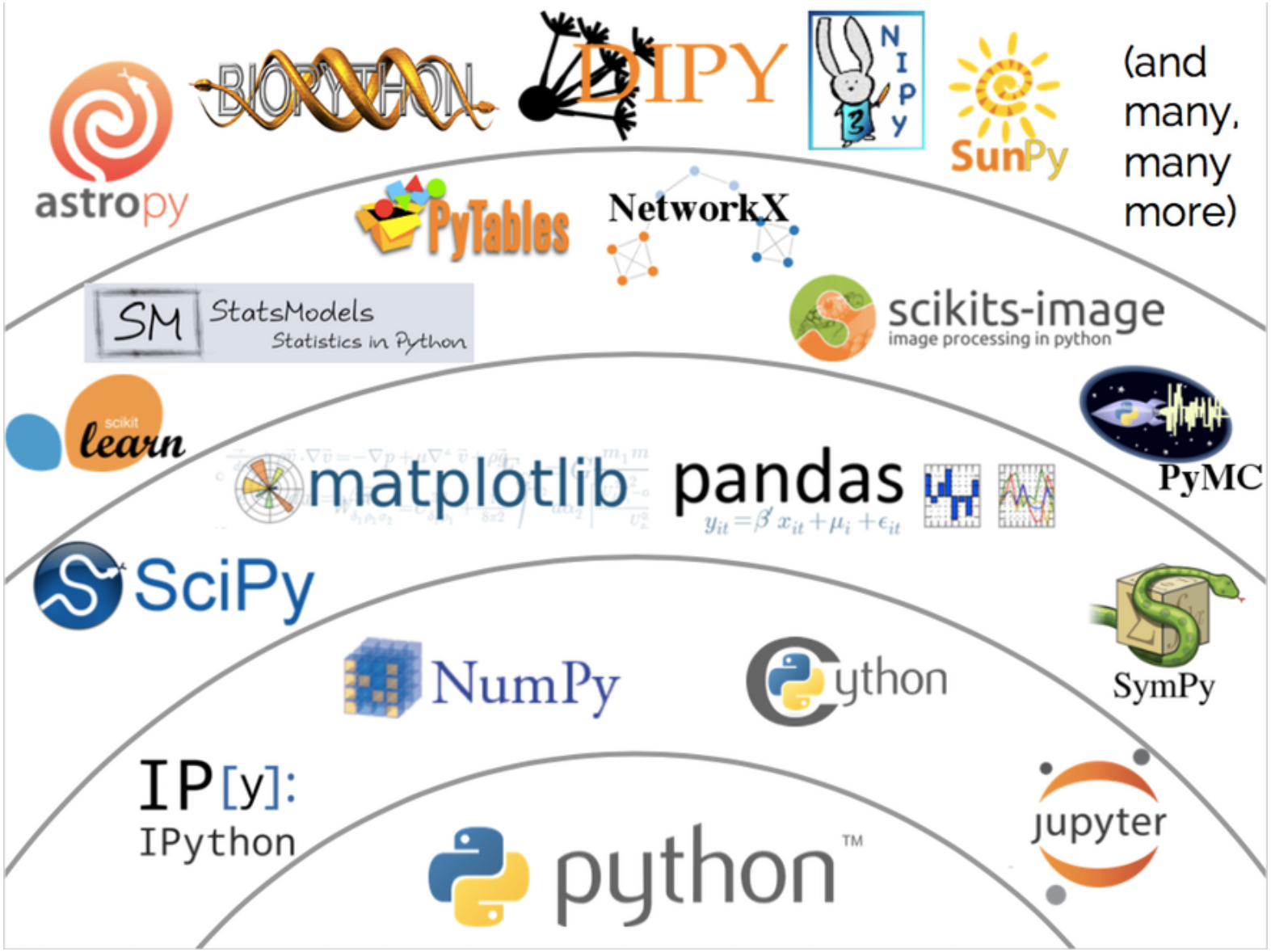
Open Source Mathematical Software. Opinion.
Notices of the American Mathematical Society 54(10):1279, 2007.

Welcome to MCS 507

1. About the Course
   - content and organization
   - expectations of the course
2. SageMath, SciPy, Julia
   - the software system SageMath
   - Python's computational ecosystem
   - the programming language Julia
3. Language Agnostic Computing
   - the Jupyter notebook and JupyterLab

the stack

picture from the slides of Jake VanderPlas



the SciPy community project

Pauli Virtanen, Ralf Gommers, Tyler Reddy, Anne Archibald, Andrew R. J. Nelson, Charles Harris, CJ Carey, Denis Laxalde, Eric Larson, Eric Moore, Eric Quintero, Evgeni Burovski, Jaime Fernández del Rio, Josef Perktold, Josh Wilson, Matthew Brett, Nikolay Mayorov, Warren Weckesser, Matt Haberland, Scott Sievert, Yu Feng, Antonio Horta Ribeiro, Ian Henriksen, K. Jarrod Millman, Stéfan J. van der Walt, and İlhan Polat.

SciPy 1.0 – Fundamental Algorithms for Scientific Computing in Python. `https://arxiv.org/abs/1907.10121`

Juan Nunez-Iglesias, Harriet Dashnow, and Stéfan van der Walt.

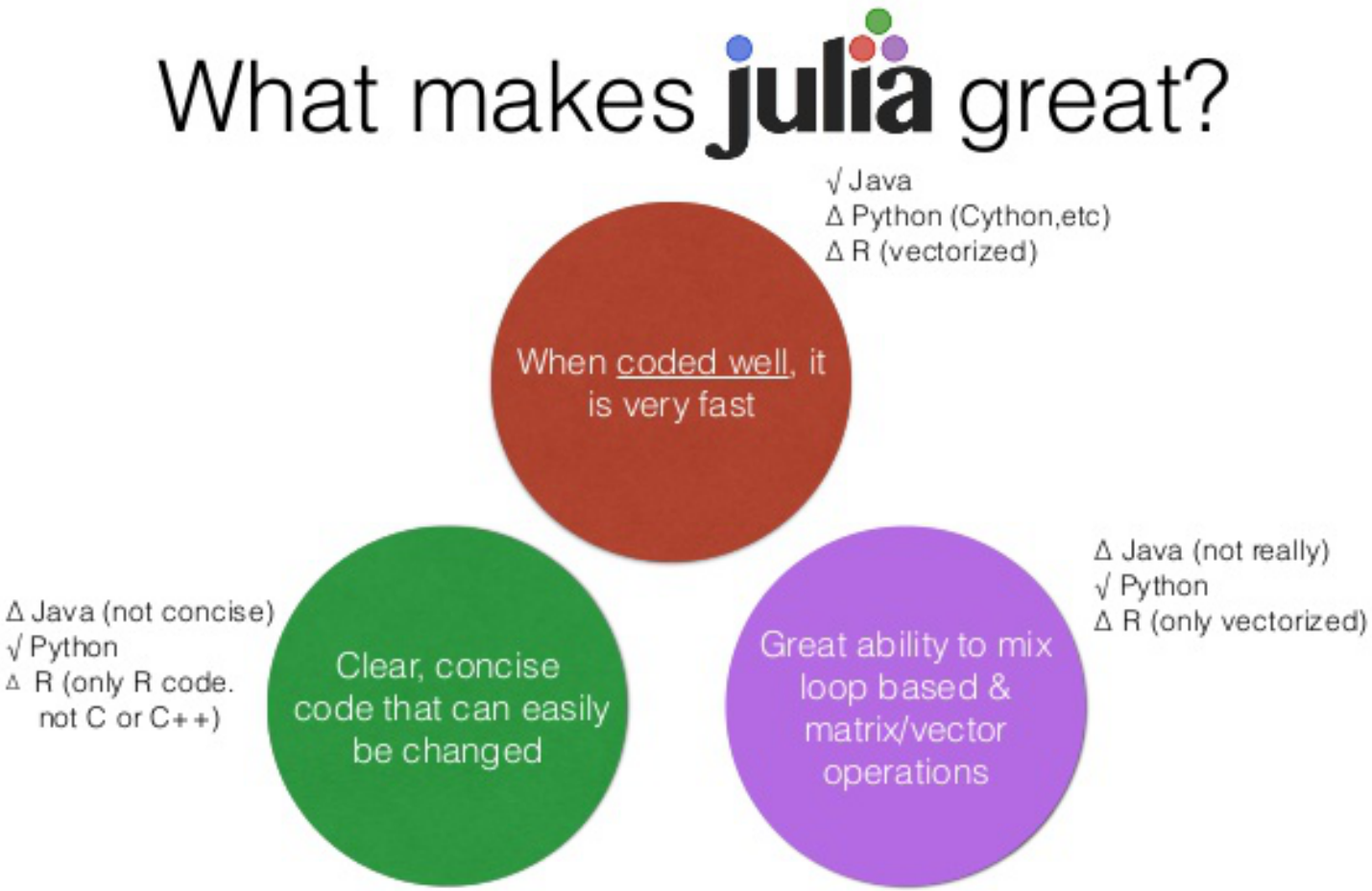
Elegant SciPy. O'Reilly Media, 2017.

Welcome to MCS 507

1. About the Course
   - content and organization
   - expectations of the course
2. SageMath, SciPy, Julia
   - the software system SageMath
   - Python's computational ecosystem
   - the programming language Julia
3. Language Agnostic Computing
   - the Jupyter notebook and JupyterLab

the Julia language

picture of Software Engineering Daily web site



high performance computing with a scripting language

Traditional mathematical software development in two steps:

1. prototype in computer algebra or scripting language,
2. rewrite high performance version in compiled language.

Julia aims to provide one environment

- to develop proof-of-concept implementations, and
- to write high performance code in the same language.

J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah.
Julia: A fresh approach to numerical computing.
SIAM Review, 59(1):65–98, 2017.

characteristics of modern software

Modern software systems offer the following solutions:

- interactive computing, scripts engage users
- cloud computing, available in a browser
- parallel computing: distributed, multicore, accelerated

Welcome to MCS 507

1. About the Course
   - content and organization
   - expectations of the course
2. SageMath, SciPy, Julia
   - the software system SageMath
   - Python's computational ecosystem
   - the programming language Julia
3. Language Agnostic Computing
   - the Jupyter notebook and JupyterLab

the Jupyter notebook and JupyterLab

A notebook allows to intermingle text and code:

- a well formatted notebook reads like a technical report;
- code cells can execute in sequence, as in a program.

Jupyter stands for Julia, Python, R, and many others ...

Language agnostic computing is a way of computing in which no particular language is promoted.

Demonstration of PSLQ algorithms in Jupyter notebook follows.

Summary + Exercises

We use Python, Julia, and SageMath to solve scientific problems.

Exercises:

1. Visit `www.sagemath.org` and install SageMath on your computer, or get familiar with running it online.
2. Visit `www.python.org` and install Python on your computer, or get familiar with running it online.
3. Visit `www.julialang.org` and install Julia on your computer, or get familiar with running it online.
4. Consider `r = 1.2599210498948732`. Apply the PSLQ algorithm to find the algebraic number that is closest to `r`.