TOOLBOX

PICK UP PYTHON

A powerful programming language with huge community support.

ILLUSTRATION BY THE PROJECT TWINS

BY JEFFREY M. PERKEL

Last month, Adina Howe took up a post at Iowa State University in Ames. Officially, she is an assistant professor of agricultural and biosystems engineering. But she works not in the greenhouse, but in front of a keyboard. Howe is a programmer, and a key part of her job is as a 'data professor' — developing curricula to teach the next generation of graduates about the mechanics and importance of scientific programming.

Howe does not have a degree in computer science, nor does she have years of formal training. She had a PhD in environmental engineering and expertise in running enzyme assays when she joined the laboratory of Titus Brown at Michigan State University in East Lansing. Brown specializes in bioinformatics and uses computation to extract meaning from genomic data sets, and Howe had to get up to speed on the computational side. Brown's recommendation: learn Python.

Among the host of computer-programming languages that scientists might choose to pick up, Python, first released in 1991 by Dutch programmer Guido van Rossum, is an increasingly popular (and free) recommendation. It combines simple syntax, abundant online resources and a rich ecosystem of scientifically focused toolkits with a heavy emphasis on community.

HELLO, WORLD

With the explosive growth of 'big data' in disciplines such as bioinformatics, neuroscience and astronomy, programming know-how is becoming ever more crucial. Researchers who can write code in Python can deftly manage their data sets, and work much more efficiently on a whole host of research-related tasks — from crunching numbers to cleaning up, analysing and visualizing data. Whereas some programming languages, such as MATLAB and R, focus on mathematical and statistical operations, Python is a general-purpose language, along the lines of C and C++ (the languages in which much commercial software and operating systems are written). As such, it is perhaps more complicated, Brown says, but also more capable: it is amenable to everything from automating small sets of instructions, to building websites, to fully fledged applications. Jessica Hamrick, a psychology PhD student at the University of California, Berkeley, has been ▶

programming in Python since 2008 and uses it in all phases of her research. In a study investigating how people manipulate geometric objects in their minds, for instance, she used the language (as well as JavaScript) to generate different shapes, present those to study participants, record their choices and analyse the data.

Despite its general-purpose power, Python is considered less painful for beginners to learn than other options. That accessibility is a function of both the language itself and the resources that have been built up around it (see 'A Python toolkit'). For example, software execution can be interactive — type a command, get a response — whereas in C, a compilation step is required to translate the code into an executable file, which complicates the process for neophytes. The language is also generally easier to handle; users do not have to predefine whether a variable will hold numbers or text, for instance. The classic programming exercise of printing 'Hello, world!' to the screen is as simple as it can be in Python — just type `print("Hello, world!")` at a Python prompt and hit Enter. "It's easier to teach novice programmers how to get things done in Python than in C++ or C," says Brown, now at the University of California, Davis. Python is in fact a popular choice for introductory programming classes in general.

The community aspect is particularly important to Python's growing adoption. Programming languages are popular only if new people are learning them and using them in diverse contexts, says Jessica McKellar, a software-engineering manager at the file-storage service Dropbox and a director of the Python Software Foundation, the non-profit organization that promotes and advances the language. That kind of use sets up a "virtuous cycle", McKellar says: new users extend the language into new areas, which in turn attracts still more users.

The community seems especially dedicated to encouraging women, Brown notes. There are numerous women-centric resources available, including workshops offered by the Hackbright Academy in San Francisco, the non-profit organization Ladies Learning Code in Toronto, Canada, and the global mentorship group PyLadies. As a master's student at McGill University in Montreal, Canada, Emily Irvine picked up Python to help her make sense of neuronal electrophysiology data. She was attracted to the language because of its "simple syntax" and "massive amount of online support". But just as important was the wider Python community, says Irvine, who will start a PhD in neuroscience at Dartmouth College in Hanover, New Hampshire, this autumn. At the PyCon conference last April in Montreal, "they just had such a welcoming atmosphere, especially towards women and scientists".

Educational resources also abound. The Software Carpentry Foundation runs a series of two-day workshops that focus on scientific programming, and many of its educational resources are available online. Online classes are also available through Coursera in Mountain View, California, and Edx in Cambridge, Massachusetts, as are do-it-yourself tutorials, such as those hosted by Codecademy in New York City. (Because Python is named in honour of Monty Python, these tutorials often work references to the British comedy troupe into their exercises: one Codecademy exercise, for example, is to capitalize and calculate the length of the phrase 'the ministry of silly walks'.)

Irvine taught herself to code using online courses and a healthy dose of the programming Q&A site stackoverflow.com. Today, she says, she considers herself somewhere between a beginner and an intermediate Python programmer, or 'pythonista', as they are sometimes called.

THE FULL MONTY

Of course, user-friendliness is meaningless if researchers cannot write the software they need. That is where Python's packages, which extend the language with new functionality, come into play. "Python was developed as a language with a philosophy that it was 'batteries included'," McKellar says — it has built-in capabilities that make it easy to get started right out of the box. But, "it also has a very mature package ecosystem around it. Anything that you could possibly write code to solve, people have written libraries to make that easier for you."

Scientific programmers, irrespective of their discipline, routinely use a small set of core packages: NumPy (mathematical arrays), SciPy (linear algebra, differential equations, signal processing and more), SymPy (symbolic mathematics), matplotlib (graph plotting) and Pandas (data analysis). Another popular tool, Cython, addresses Python's relatively slow execution speed. Cython optimizes certain aspects of Python code, such as 'for' loops (used to instruct a program to repeatedly run a specific block of code) that are notoriously slow, essentially by converting them into C. "You can get speed-ups that are up to 1,000 times faster than standard Python," says Paul Nation, a theoretical physicist at Korea University in Seoul.

The IPython Notebook is another popular package — Howe terms it "a coder's lab notebook" — that allows users to interleave data, code and explanatory text in a single browser-based page, rather than in separate files (see *Nature* **515,** 151–152; 2014).

Beyond the core packages, software packages exist for just about every scientific discipline, including scikit-Learn for machine learning, Biopython for bioinformatics, PsychoPy for psychology and neuroscience and Astropy for astronomers. Thomas Robitaille, a coordinator of the Astropy project and a researcher at the Max Planck Institute for Astronomy in Heidelberg, Germany, says that Astropy was created to reduce duplicated effort between research groups. It gives users a core set of abilities, such as ways to convert coordinates from one astronomical mapping system to another, and a unified interface for reading and writing different data file formats, manipulating images and carrying out cosmological calculations. QuTip, another Python package, enables researchers working on quantum mechanics to define a system and then simulate how it behaves. The project was launched in 2010 by Nation and Robert Johansson, a postdoctoral fellow in RIKEN's Interdisciplinary Theoretical Science Research Group in Wako, Japan, to adapt into Python a MATLAB package that Nation was using.

Such packages are key enablers of McKellar's 'virtuous cycle'. But researchers could probably do their work using any language, provided they put in the time to learn it. (Indeed, in many languages, including Python, it is possible to run algorithms written in a different language, thereby allowing researchers to reuse their old code.) The difficult part of learning to program lies with the fundamentals, says Brown — once a researcher has those nailed down, adapting to a new language is just a matter of syntax. What matters most in the early stages is having a good support network. "Pick the programming language based on what people around you are using," Brown advises. Increasingly, that language is Python. ■

Jeffrey M. Perkel *is a writer based in Pocatello, Idaho.*

A PYTHON TOOLKIT

How to get started

- Install Python through Anaconda or Enthought Canopy and find documentation at the Python Software Foundation
- Lessons for beginners can be found at Software Carpentry; Learn Python the Hard Way; Codecademy; and Think Python
- Other online resources on Python programming include a course from the Massachusetts Institute of Technology in Cambridge, lecture notes from Thomas Robitaille at the Max Planck Institute for Astronomy in Heidelberg, Germany, and a widely recommended essay from Google's head of research, Peter Norvig
- Open-source packages are available through SciPy.org
- Guides to programming and community support are available through Ladies Learning Code and Stack Overflow. PyCon.org lists conferences around the world.

Links to these resources can be found at go.nature.com/x2pzh1

NATURE.COM
For more on scientific software, apps and online tools, visit: nature.com/toolbox