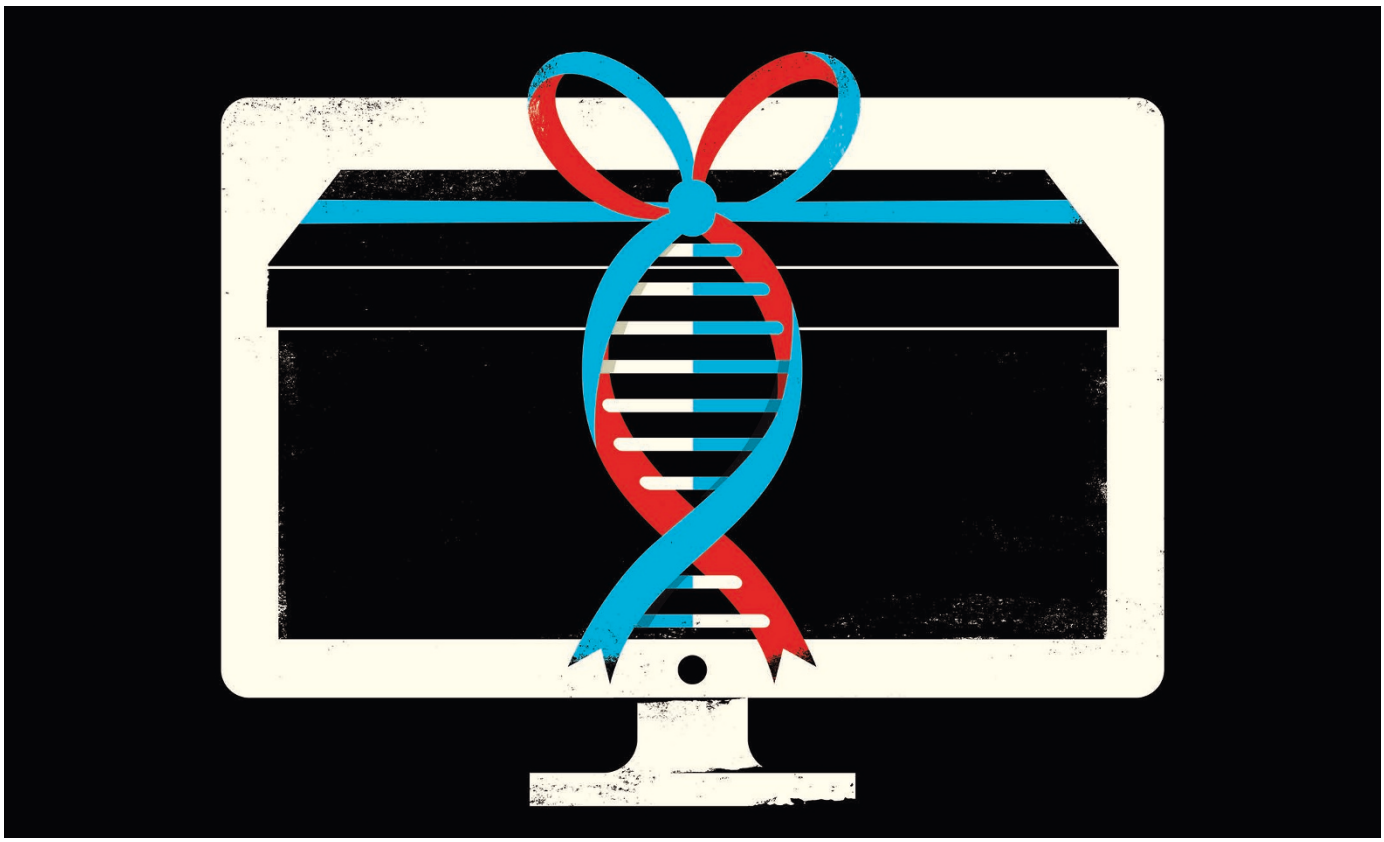
TOOLBOX

SOFTWARE SIMPLIFIED

Containerization technology takes the hassle out of setting up software and can boost the reproducibility of data-driven research.



PROJECT TWINS

BY ANDREW SILVER

In 2015, geneticist Guy Reeves was trying to configure a free software system called Galaxy to get his bioinformatics projects off the ground. After a day or two of frustration, he asked members of his IT department for help. They installed Docker, a technology for simulating computational environments, which enabled him to use a special version of Galaxy that came packaged with everything he needed — called a container. A slight tweak to the Galaxy settings, and he was "done before lunch".

Reeves, at the Max Planck Institute for Evolutionary Biology in Plön, Germany, is one of many scientists adopting containers. As science becomes ever more data intensive, more software is being written to extract knowledge from those data. But few researchers have the time and computational know-how to make full use of it. Containers, packages of software code and the computational environment to run it, can close that gap. They help researchers to use a wider array of software, accelerate experiments and promote reproducibility.

Containers are essentially lightweight, configurable virtual machines — simulated versions of an operating system and its hardware, which allow software developers to share their computational environments. Researchers use them to distribute complicated scientific software systems, thereby allowing others to execute the software under the same conditions that its original developers used. In doing so, containers can remove one source of variability in computational biology. But whereas virtual machines are relatively resource-intensive and inflexible, containers are compact and configurable, says C. Titus Brown, a bioinformatician at the University of California, Davis. Although configuring the underlying containerization software can be tricky, containers can be modified to add or remove tools according to the user's need — flexibility that has boosted their popularity, he says. "I liked the idea of having something that works out of the box," says Reeves.

WHAT'S INSIDE

Lab-built tools rarely come ready to run. They often take the form of scripts or programming source code, which must be processed and ▶

configured. Much of the software requires additional tools and libraries, which the user may not have installed. Even if users can get the software to work, differences in computational environments, such as the installed versions of the tools it depends on, can subtly alter performance, affecting reproducibility. Containers reduce that complexity by packaging the key elements of the computational environment needed to run the desired software, including settings and add-ons, into a lightweight, virtual box. They don't alter the resources required to run it — if a tool needs a lot of memory, then so too will its container. But they make the software much easier to use, and the results easier to reproduce.

Depending on the software used — Docker, Singularity and rkt are popular — containers can run on Windows, Mac OS X, Linux or in the cloud. They can package anything from a single process to a complex environment such as Galaxy. These tools can interact with each other, sharing data or building pipelines, for instance. Because each application resides in its own box, even tools that would ordinarily conflict with each other can run harmoniously.

Docker uses executable packages, called images, which include the tool to be contained as well as the developer's computational environment. To create a Docker image, a developer creates a configuration file with instructions on how to download and build all the required tools inside it. He or she then 'runs' the file to create an executable package. All the user then needs to do is retrieve the package and run it. Other tools can also generate images. The Reprozip program, for example, assembles Docker-compatible packages by watching as software tools run and tracing the input files and software libraries that the tool requires.

Deborah Bard, a computer scientist at the National Energy Research Scientific Computing Center in Berkeley, California, helps researchers to install their software on the lab's supercomputer. She recalls spending three or four days installing a complex software pipeline for telescope simulation and analysis. Using containers cut this time down to hours. "You can spend your time doing science instead of figuring out compiler versions," she says.

For Nicola Mulder, a bioinformatician at the University of Cape Town in South Africa, containers help her to synchronize a cross-border bioinformatics network she runs in Africa, called H3ABioNet. Not all African institutions have access to the same computational resources, she explains, and Internet connectivity can be patchy. Containers allow researchers with limited resources to access the tools that they otherwise might not be able to.

They also allow researchers with sensitive genomic data to collaborate and compare findings without actually sharing the underlying data, Mulder says. And, if researchers at one site obtain different results from their colleagues at another, the standardization the containers provide could eliminate one of the reasons why.

OBTAINING CONTAINERS

Although computer scientists have multiple options for container platforms, Docker, which is an open-source project launched in 2013, is perhaps the most popular among scientists. It has a large registry of prebuilt containers and an active online community that competitors have yet to match. But many administrators of high-performance computing systems preclude Docker use because it requires high-level administrative access privileges to run. This type of access may allow users to copy or damage anything on the system. An add-on to the fee-based enterprise edition allows users to sidestep that requirement, but it is not available with the free, community edition. They can, however, use a different containerization tool such as Shifter, which doesn't require full privileges, or root access, but still supports Docker images.

The requirement for root access is the biggest obstacle to widespread adoption of Docker, Brown explains. Many academics run bioinformatics tools on high-performance computing clusters administered by their home institutions or the government. "Of course, they don't have administrative privileges on most of those systems," he says. Brown spends about US$50,000 annually for cloud computing time on Amazon Web Services, but he says this represents just one-third of his computing work; the rest is carried out on a cluster at Michigan State University, where he lacks root-level access. As a result, Brown creates Docker containers of his tools for distribution, but can rarely use them himself.

Researchers can access Docker images either from the platform's own hosting service, Docker Hub, or from registries of containers such as BioContainers and Dockstore, which allow the sharing of tools vetted by other scientists. Brian O'Connor at the University of California, Santa Cruz, who was the technical lead for the Dockstore registry, recommends that scientists look through container registries to find a tool that works for their project instead of trying to reinvent something that already exists.

But actually getting the underlying Docker software to run properly can be challenging, says Simon Adar, chief executive of Code Ocean in New York, an online service that aims to simplify the process. "It's too technical, it was designed for developers to deploy complex systems." The service, launched in February, creates what Adar calls "compute capsules", which comprise code, data, results and the Docker container itself. Researchers upload their code and data, and then either execute it in a web browser or share it with others — no installation required. Adar likens the process to sharing a YouTube video. The company even offers a widget that enables users to embed executable code in web pages.

Shakuntala Baichoo, a computer scientist at the University of Mauritius in Moka, learned about containers at a communal programming event, called a hackathon, organized by H3ABioNet. Previously, she spent hours helping collaborators install her tools. In making the tools easier to install, she says, containers not only free up her time, but they might also encourage scientists to test them and provide feedback.

At CERN, the particle-physics laboratory near Geneva, Switzerland, scientists use containers to accelerate the publication process, says physicist Kyle Cranmer at New York University who works on CERN's ATLAS project, which searches for new elementary particles. When physicists run follow-up studies, they have to dig up code snippets and spend hours redoing old analyses; with containers, they can package ready-to-use data analysis workflows, simplifying and shortening the process.

ADVANCING REPRODUCIBILITY

Cranmer says that although much of the debate around reproducibility has focused on data and code, computing environments themselves also play a big part. "It's really essential," he says. One study of an anatomical analysis tool's performance in different computing environments, for example, found that the choice of operating system produced a small but measurable effect (E. H. B. M. Gronenschild *et al. PLoS ONE* **7**, e38234; 2012).

But containers are only as good as the tools they encapsulate, says Lorena Barba, a mechanical and aerospace engineer at George Washington University, Washington DC. "If researchers start stuffing their bad code into a container and pass it on, we are foredoomed to failure." And, says Brown, without pressure from funding agencies and journals, containers are unlikely to make researchers suddenly embrace computational reproducibility.

Indeed, few researchers are using containers, says Victoria Stodden, a statistician at the University of Illinois at Urbana–Champaign who studies computational reproducibility. In part that's because of a lack of need or awareness, but it is also because they might not have the computer skills needed to get going.

Behind the scenes, however, that could be changing. Companies such as Google and Microsoft already run some software in containers, says Jonas Almeida, a bioinformatician at Stony Brook University, New York. Large-scale bioinformatics projects may not be far behind. The cloud-based version of Galaxy will eventually run inside containers by default, says Enis Afgan, a computer scientist at Johns Hopkins University in Baltimore, Maryland, who works on Galaxy.

In 5–10 years, Almeida predicts, scientists will no longer have to worry about downloading and configuring software; tools will simply be containerized. "It's inevitable," he says. ■

Andrew Silver *is a freelance science writer in Berlin, Germany.*