ILLUSTRATION BY THE PROJECT TWINS

CONTAINERS IN THE CLOUD

Standardized platforms allow researchers to run each other's software – no installation required. **By Jeffrey M. Perkel**

Murphy's law for the digital age: anything that can go wrong, will go wrong during a live demonstration. For Ben Marwick, that happened in front of a roomful of landscape-archaeology students in Berlin. The topic: computational reproducibility using Docker.

Docker is a software tool that generates 'containers' – standardized computational environments that can be shared and reused. Containers ensure that computational analyses always run on the same underlying infrastructure, fostering reproducibility. Docker thereby insulates researchers from the challenges of installing and updating research software. However, it can be difficult to use.

Marwick, an archaeologist at the University of Washington in Seattle, had become proficient in migrating Docker configuration files ('Dockerfiles') from one project to the next, making minor tweaks and getting them to work. Colleagues in Germany invited him to teach their students how to follow suit. But because every student had a slightly different set of hardware and software installed, each one required a customized configuration. The demo "was a complete disaster", Marwick says.

Today, a growing collection of services allows researchers to sidestep such confusion. Using these services – which include Binder, Code Ocean, Colaboratory, Gigantum and Nextjournal – researchers can run code in the cloud without needing to install more software. They can lock down their software configurations, migrate those environments from laptops to high-performance computing clusters and share them with colleagues. Educators can create and share course materials with students, and journals can improve the reproducibility of results in published articles. It's never been easier to understand, evaluate, adopt and adapt the computational methods on which modern science depends.

William Coon, a sleep researcher at Harvard Medical School in Boston, Massachusetts, spent weeks writing and debugging an algorithm, only to discover that a colleague's containerized code could have saved a lot of time. "I could have just gotten up and running, using all of the debugging work that he had already done, at the click of a button," he says.

Scientific software often requires installing, navigating and troubleshooting a byzantine network of computational 'dependencies'

– the code libraries and tools on which each software module relies. Some have to be compiled from source code or configured just so, and an installation that should take a few minutes can degenerate into a frustrating online odyssey through websites such as Stack Overflow and GitHub. "One of the hardest parts of reproducibility is getting your computer set up in exactly the same way as somebody else's computer is set up. That is just ridiculously difficult," says Kirstie Whitaker, a neuroscientist at the Alan Turing Institute in London.

Easier evaluation

Docker reduces that to a single command. "Docker really provides reduced friction for that stage of the cycle of reproducing somebody else's work, in which you have to build the software from source and combine it with other external libraries," says Lorena Barba, a mechanical and aerospace engineer at George Washington University in Washington DC. "It facilitates that part, making it less error-prone, making it less onerous in researcher time."

Barba's team does most of its work in Docker containers. But that is a computationally savvy research group; others might find the process daunting. A text-based 'command-line' application, Docker has dozens of options, and building a working Dockerfile can be an exercise in frustration.

That's where the cloud-based services come in. Binder is an open-source project that allows users to test-drive computational notebooks – documents such as Jupyter or R Markdown notebooks, which blend code, figures and text. Colaboratory (free), Code Ocean, Gigantum and Nextjournal (the latter three have free and paid tiers) let users write code in the cloud as well and, in some cases, bundle it with the data to be processed. These platforms also allow users to modify the code and apply it to other data sets, and provide version-control features for reviewing changes.

Such tools make it easier for researchers to evaluate their colleagues' work. "With Binder, you have taken that barrier [of software installation] away," says Karthik Ram, a computational ecologist at the University of California, Berkeley. "If I can click that button, be dropped into a notebook where everything is installed, the environment is exactly the way you intended it to be, then you've made my life easier to go take a look and give you feedback."

Identifying required dependencies, and where to find them, varies with the platform. On Code Ocean and Gigantum, it's a point-and-click operation, whereas Binder requires a list of dependencies in a Github respository. Whitaker's advice: codify your computing environment as early as possible in a project, and stick with it. "If you try and do it at the end, then you are basically doing archaeology on your code, and it's really, really hard," she says. Ram developed a tool called Holepunch for projects that use the statistical programming language R. Holepunch distils the process of setting up Binder into four simple commands. (See examples of our code running on all five platforms at go.nature.com/2ps9se1.)

The easiest way to try Binder is at mybinder.org, a free, albeit computationally limited, website. Or, for greater power and security, researchers can build private 'BinderHubs' instead. The Alan Turing Institute has two, including one called Hub23 (a reference to Hut 23 at the Second World War code-breaking facility at Bletchley Park, UK), that provides greater computational resources and the ability to work with data sets that cannot be publicly shared, Whitaker says. The Pangeo community, which promotes open, reproducible and scalable geoscience, built a dedicated BinderHub so that researchers can explore climate-modelling and satellite data sets that can amount to tens of terabytes, says Joe Hamman, a computational hydroclimatologist at the National Center for Atmospheric Research in Boulder, Colorado. (Whitaker's team has published a tutorial on building a BinderHub at go.nature.com/349jscv.)

> **"Researchers can be confident that their code will remain usable, whichever platform they choose."**

Languages and clouds

Google's Colaboratory is basically a cross between a Jupyter notebook and Google Docs, meaning users can share, comment on and jointly edit notebooks, which are stored on Google Drive. Users execute their code in the Google cloud – only the Python language is officially supported – on a standard central processing unit (CPU), a graphics processing unit (GPU) or a tensor processing unit (TPU), a specialized chip optimized for Google's TensorFlow deep-learning software. "You can open up your notebook or someone else's notebook from GitHub, start playing around with it and then save your copy on Google Drive and work on it later," says Jake VanderPlas, a member of the Colaboratory team at Google in Seattle.

Nextjournal supports notebooks written in Python, R, Julia, Bash and Clojure, with more languages in development. According to Martin Kavalar, chief executive of Nextjournal, which is based in Berlin, the company has registered nearly 3,000 users since it launched the platform on 8 May.

Gigantum, a beta version of which launched last year, features a browser-based client that users can install on their own system or remotely, for cloud-based coding and execution in the Jupyter and RStudio coding environments. Coon, who uses Gigantum to run machine-learning algorithms in the Amazon cloud, says the service makes it easy for collaborators to hit the ground running. "[They] can read through my Gigantum notebooks and use this cloud-compute infrastructure to do the training and learning," he explains.

Then there's Code Ocean, which supports both notebooks and conventional scripts in Python, R, Julia, Matlab and C, among other languages. Several journals now use Code Ocean for peer review and to promote computational reproducibility, including titles from Taylor & Francis, De Gruyter and SPIE. In 2018, *Nature Biotechnology*, *Nature Machine Intelligence* and *Nature Methods* launched a pilot programme to use Code Ocean for peer review; *Nature*, *Nature Protocols* and *BMC Bioinformatics* subsequently joined the trial. More than 95 papers have now been involved in the trial, according to Erika Pastrana, editorial director of Nature Research's applied-science and chemistry journals, and more than 20 of those have been published.

Felicity Allen, a computer scientist at the Wellcome Sanger Institute in Hinxton, UK, co-authored one study in that trial, which analysed the types of mutation that can arise from CRISPR-based gene editing (F. Allen *et al. Nature Biotechnol.* **37**, 64–72; 2019). She estimates that it took a week to get the Code Ocean environment working. "The reviewers seemed to really like it," Allen says. "And I think it was really nice that it made an example that someone could just press 'go' on and it would run."

Although some worry about the long-term viability of commercial container-computing services, researchers do have options. Simon Adar, chief executive of Code Ocean, notes that Code Ocean 'compute capsules' are archived by the CLOCKSS project, which preserves digital copies of online scientific literature. And Code Ocean, Gigantum and Nextjournal allow Dockerfiles to be exported for use on other platforms. All of which means that researchers can be confident that their code will remain usable, whichever platform they choose.

Benjamin Haibe-Kains, a computational pharmacogenomics researcher at the Princess Margaret Cancer Centre in Toronto, Canada, adopted Code Ocean to respond quickly to critiques of an analysis he published in *Nature* (B. Haibe-Kains *et al. Nature* **504**, 389–393; 2013). For him, Code Ocean provides a way to ensure his code can be used and evaluated by his team, peer reviewers and the broader scientific community. "It's not so much that an analysis must be correct or wrong," he says. "Nothing is really fully correct in this world. However, if you're very transparent about it, you can always communicate efficiently in the face of criticism. You have nothing to hide; everything is there."

Jeffrey M. Perkel is technology editor at *Nature*.