Dartmouth College
Department of Psychological & Brain Sciences
HB 6207 Moore Hall
Hanover, NH 03755

October 26, 2022

To the editors of *Software X*:

We have enclosed our manuscript entitled `davos`*: a Python package "smuggler" for constructing lightweight reproducible notebooks* to be considered for publication as an *Article*.

Our manuscript describes a new Python package, `davos`. When used in combination with a notebook-based Python project, the `davos` library provides tools for specifying and automatically installing the correct versions of the project's dependencies. Our library also ensures that the correct versions of those dependencies are in use any time the notebook's code is executed. This enables researchers to share a complete reproducible copy of their code within a single Jupyter notebook (.ipynb) file.

Broadly, we designed the `davos` library to target a "sweet spot" along a continuum of existing approaches to facilitating reproducible code-based research. At one end of this continuum, "lightweight" approaches entail simply sharing raw code (e.g., plain-text python scripts) or Jupyter notebooks (e.g., JSON files that comprise a mix of text, code, and embedded media). These lightweight solutions benefit from very low setup costs (which increase accessibility), but they typically do not make any attempt to manage or constrain the computing environment in which the shared code is executed. At best, when dependencies are missing on the end user's system, shared code may fail to run entirely. And when the *versions* of a project's dependencies differ across the original author's system and the end user's system, shared code may (at worst) behave in unexpected ways or even cause damage.

At the other end of this continuum, "heavyweight" approaches entail simulating or replicating, to varying depths, the original computing environment that the shared code was developed for. For example, virtual environments, containerized systems, and virtual machines reproduce (respectively) a complete Python environment, operating system, and/or full hardware simulation of the original environment. Each of these systems guarantees, to varying degrees, that shared code will behave as expected for the end user. A downside to these approaches is that they are often effort- and/or resource-intensive, since they require installing additional tools (e.g., Anaconda, Docker, machine emulators, etc.) and configuration files (e.g., environment configuration files, Dockerfiles, system images, etc.), alongside the to-be-shared code itself.

The `davos` library is lightweight in the sense that it does not require any setup beyond that required to run standard Jupyter notebooks. But `davos` also provides infrastructure for precisely controlling project dependencies in a way that can easily be embedded into standard notebooks. This provides a complete system for sharing reproducible code inside of a standard notebook file.

Beyond its intended primary role in facilitating reproducible research, `davos` is also useful in pedagogical settings (e.g., courses that involve programming in notebook-based environments), or when putting together lightweight notebook-based demonstrations.

Thank you for considering our manuscript, and we hope you will find it suitable for publication in *Software X*.

Sincerely,

Jeremy R. Manning