

The Software Gardening Almanack

Dave Bunten¹, Will Davidson¹, Faisal Alquaddoomi¹, Vincent Rubinetti¹, Gregory P. Way¹

¹Department of Biomedical Informatics, University of Colorado Anschutz Medical Campus



Anschutz

I. Fragile code & sustainability debt

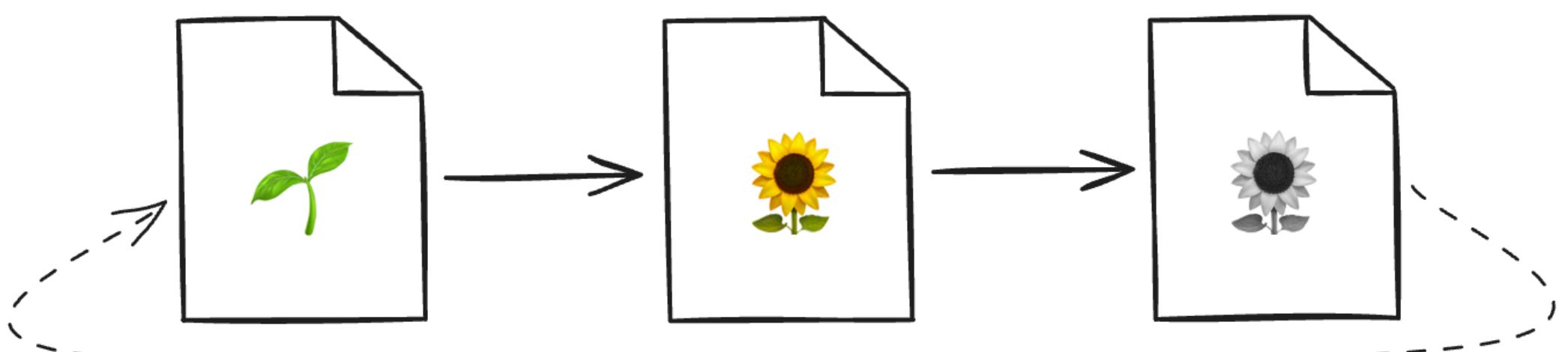


Figure 1: “Software grows and decays.” Repositories accumulate complexity, missing docs, and other issues that erode maintainability over time. The Almanack treats software like a living ecosystem and provides measurements + checks to slow decay and guide care.

Scientific software often becomes brittle: documentation gaps, weak tests, and ad-hoc practices combine with time to create **sustainability debt**. This debt hides defects, slows research, and undermines reproducibility. Existing guidance is scattered and qualitative. Teams need **measurable signals** and **actionable checks** that map to better long-term outcomes.

II. Handbook + Python package

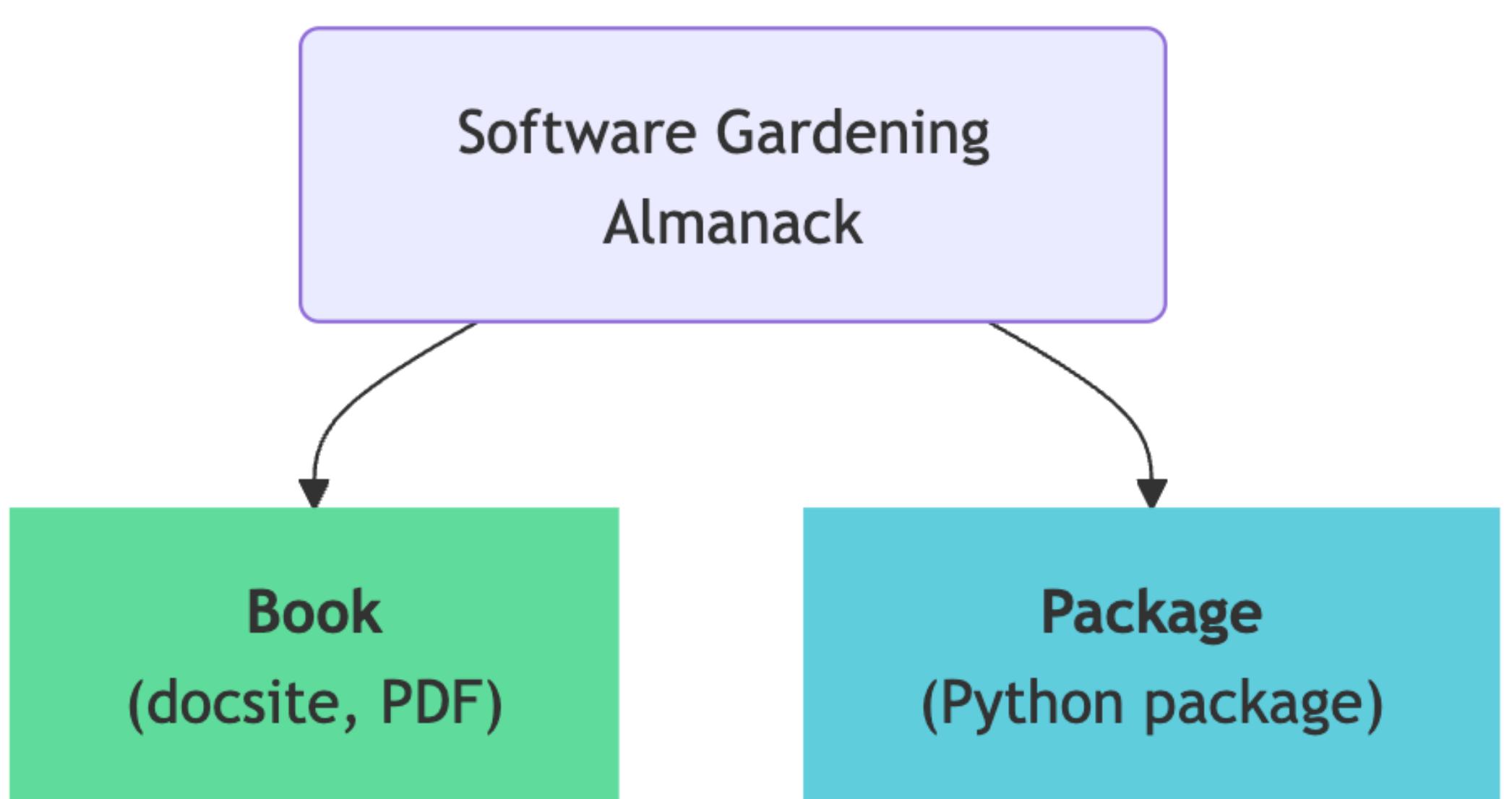


Figure 2: Two parts: a public handbook (education, examples, and rationale) and a Python package that generates metrics and runs checks on any repo. Integrate as CLI, pre-commit hook, or Python API to surface sustainability risks early.

We present The Software Gardening Almanack, an open-source project with two complementary components:

- **Handbook (Jupyter Book):** teaching the “why” and “how”—concepts, examples, and tutorials. The handbook provides a portable foundation for learning sustainable practices.
- **almanack Python package:** the “do”—compute metrics (including software information entropy), build reports, and run lint-style **sustainability checks** through Python API, CLI, or pre-commit CI.

Together, they help scientific developers quantify maintenance risk, prioritize fixes, and cultivate code that lasts. The Almanack fills an educational technology gap which is endemic in scientific software development today.

III. Learning with the Handbook

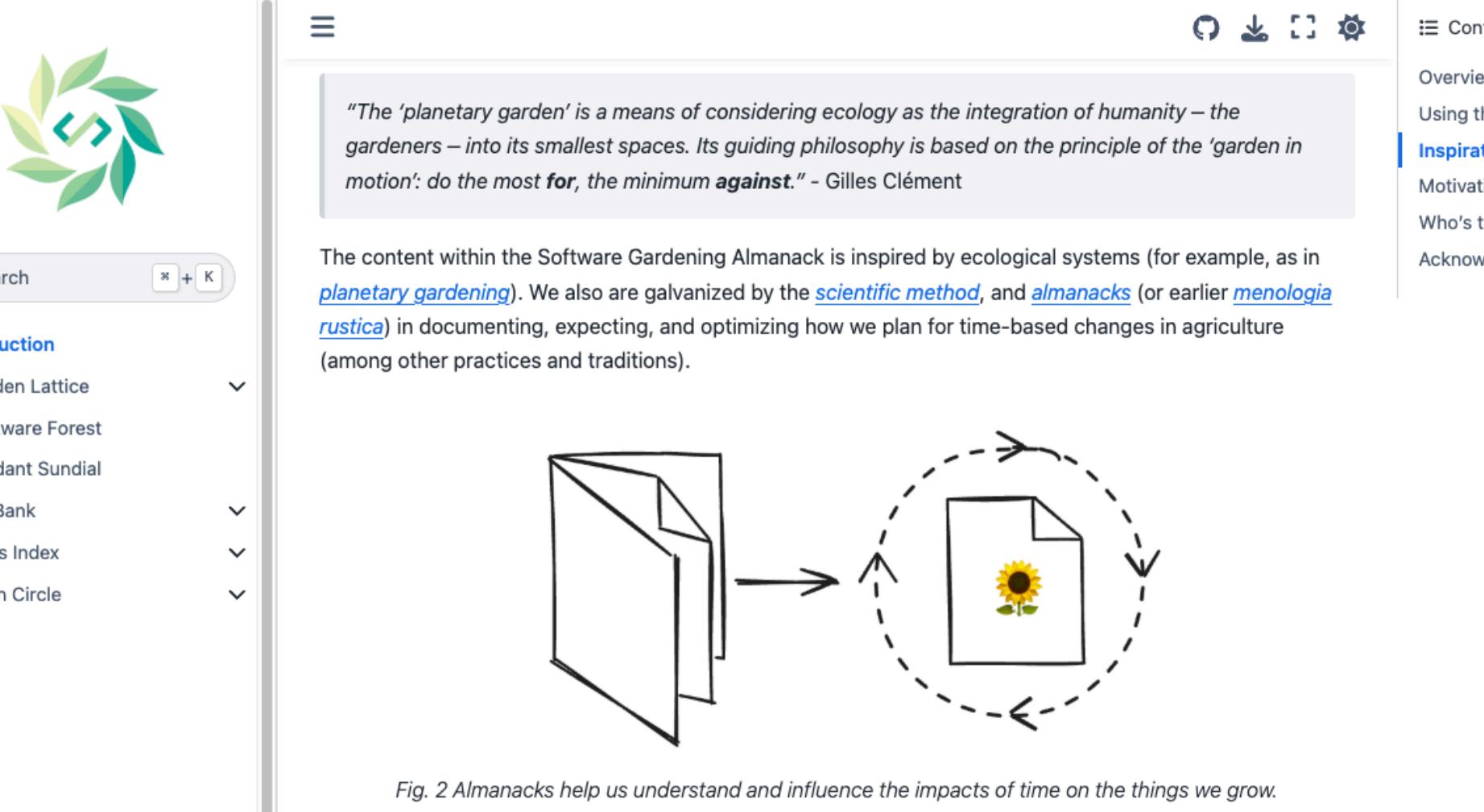


Figure 3: The Almanack handbook provides a learning avenue and reference manual for sustainable software development.

Educational material is provided for reference and general training through a Jupyter Book for the Almanack. This content covers the philosophical foundations and references to specific guidance for various checks.

IV. Using the almanack package

```
import almanack
import json
import pandas as pd

# gather the almanack table using the almanack repo as a reference
almanack_table = almanack.metrics.data.get_table("almanack")
pd.DataFrame(almanack_table)
```

Figure 4: The Almanack package allows you to gather raw metrics for deep analysis or perform a lint-style check on repositories.

a) Install with pip

```
# install from PyPI
pip install almanack
```

b) Use as a shell CLI

```
# run lint-style sustainability checks
almanack check https://github.com/an-org/your-repo
# gather all metrics about a project
almanack table path/to/repository
```

c) Pre-commit integration

```
# .pre-commit-config.yaml
# use the almanack directly through pre-commit
- repo: https://github.com/software-gardening/almanack
  rev: v0.1.1
  hooks:
    - id: almanack-check
```

V. What the metrics capture

The almanack package gathers, for example, the following metrics:

- **Repository hygiene:** README, license, citation, contribution guide, code of conduct, docs location.
- **Process health:** CI configuration, build status, test presence, coverage parsing (e.g., Python).
- **Community signals:** contributors, tags/releases, issue tracker availability.
- **Evolution/complexity:** information entropy at file and repo levels to highlight hotspots and decay risk.

a) Entropy as a sustainability signal

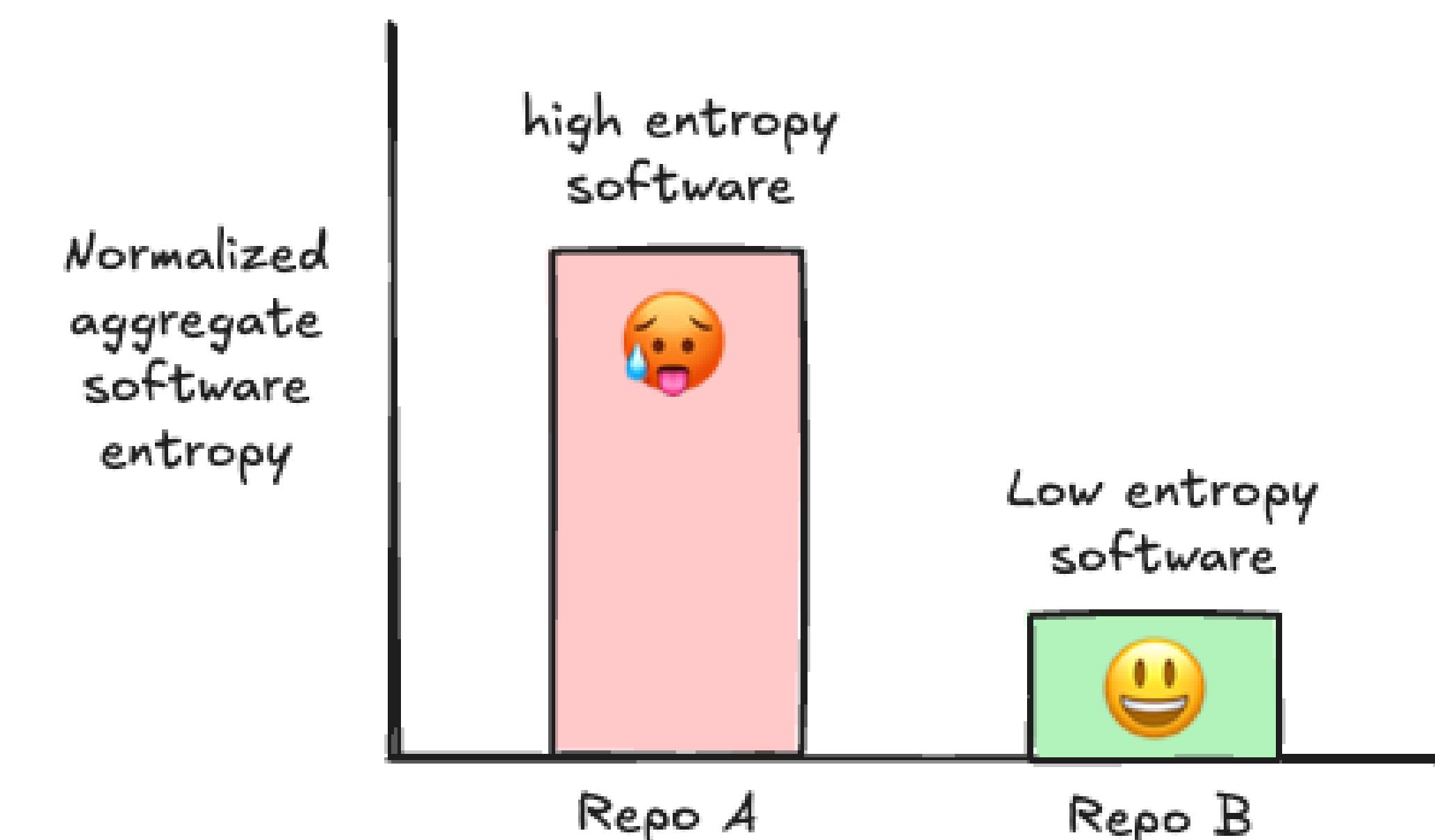


Figure 5: High entropy reflects irregular, unpredictable modification patterns—raising maintenance cost and failure risk. Low entropy suggests consistent, structured development and fewer surprises.

We investigated Shannon entropy as a metric computed on code changes across a project’s history (originally proposed in *Predicting faults using the complexity of code changes*, Hassan, 2009). Entropy is calculated at the file level across commits and normalize by lines of code to compare projects of different sizes.

VI. PubMed GitHub repository patterns

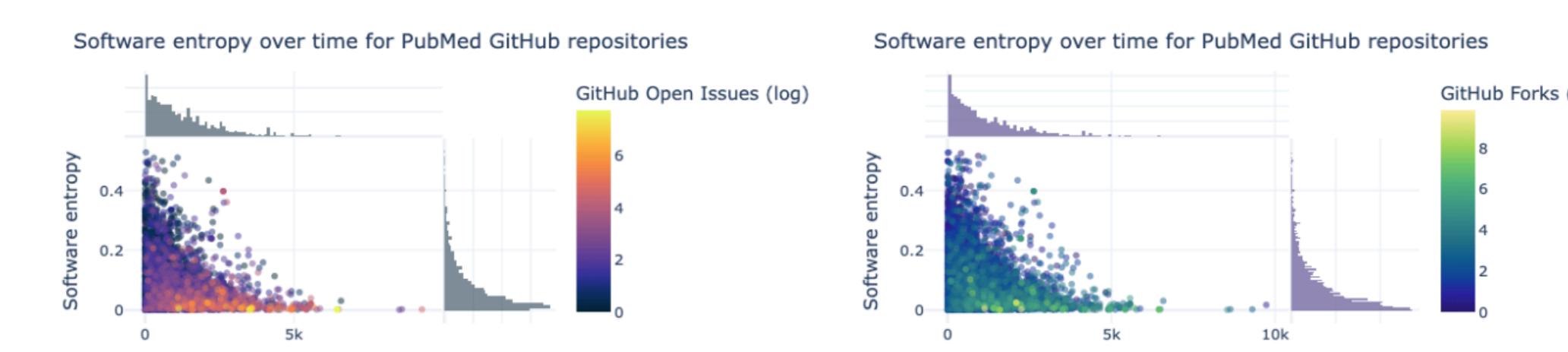


Figure 6: Higher entropy correlated with shorter lifespans and instability, while lower entropy with stronger community engagement aligned with more sustainable projects.

Analyzing ~10,000 PubMed-linked GitHub repositories (since 2011), we paired normalized, file-level entropy with repository metadata. An Almanack Seed Bank notebook shows how spotting entropy hotspots guides targeted refactoring to curb decay and extend longevity.

VII. Pathways in the Garden

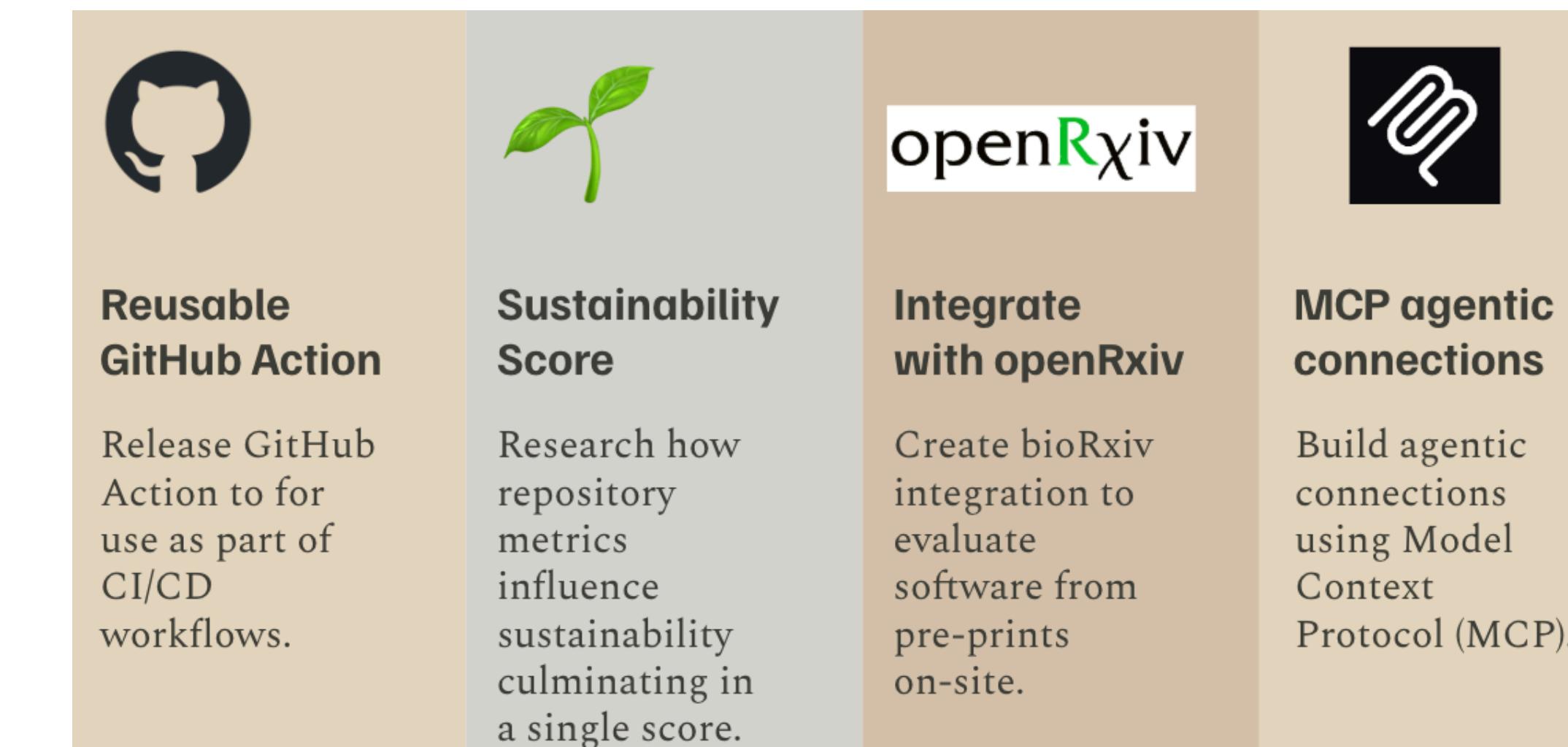


Figure 7: The Software Gardening Almanack will expand sustainability scores, add a GitHub Action, integrate with openRxiv, and open MCP endpoints to find growth through connection.

Looking ahead, the Almanack is poised to broaden its impact (Figure 7). We plan to introduce new metrics, including a **sustainability score**, alongside technical features such as a **GitHub Action** and integration tools for embedding with scientific communication platforms like **openRxiv**. In parallel, we are exploring an **MCP** (**Model Context Protocol**) set of **endpoints** designed to bring the Almanack’s toolset into large language models and agentic workflows, expanding its reach into next-generation research software ecosystems. Together, these innovations will place the Almanack directly in the hands of researchers, developers, and educators shaping the future of scientific software.

VIII. Acknowledgements

This work was supported by the Better Scientific Software Fellowship Program, a collaborative effort of the U.S. Department of Energy (DOE), Office of Advanced Scientific Research via ANL under Contract DE-AC02-06CH11357 and the National Nuclear Security Administration Advanced Simulation and Computing Program via LLNL under Contract DE-AC52-07NA27344; and by the National Science Foundation (NSF) via SHI under Grant No. 2327079.

We thank contributors and supporters of the Almanack effort including the following:

- Milton Pividori and members of the Pividori Lab
- Members of the Way Lab at the University of Colorado Anschutz Medical Campus
- Aditi Gopalan, Jineta Banerjee and The Multi-Consortia Coordinating (MC²) Center for Cancer Biology managed by Sage Bionetworks
- Jonathan Starr and collaborators of The Map of Open Source Science (MOSS) managed by NumFocus (<https://www.opensource.science/moss>)
- Better Scientific Software (BSSw)
- Sustainable Horizons Institute
- Department of Biomedical Informatics within the School of Medicine at the University of Colorado Anschutz Medical Campus