Make your code count: Leveraging open-source tools in quantum technology

Galileo Galilei Institute – Summer School on Quantum Computing and Sensing – June 22nd, 2021

Nathan Shammah
nathan@unitary.fund

Creating a quantum technology ecosystem faster, better, and to benefit everyone.
Outline

● About Unitary Fund
  ○ Community activities
  ○ In-house research

● Quantum technology

● Open-source software (OSS)
  ○ An overview of scientific OSS
  ○ Make your code count
Unitary Fund

Non-profit helping create a quantum technology ecosystem faster, better, and to benefit everyone.

Our supporters

Our partners

https://unitary.fund/
Unitary Fund Team

**Team**

- **Will Zeng, PhD**  
  President. Head of Quantum Research at Goldman Sachs. Fmr. product/sw lead at Rigetti. Oxford quantum algorithms PhD.

- **Nathan Shammah, PhD**  
  CTO. Lead developer at QuTiP. Visiting scientist at RIKEN. PhD in quantum physics from Univ. of Southampton.

- **Sarah Kaiser, PhD**  
  Community Manager. Co-founder of Q# Community, Microsoft MVP, activist for open source and diversity in quantum. UWaterloo PhD in quantum computing.

- **Andrea Mari, PhD**  
  > 40 peer reviewed scientific publications. Contributor to Pennylane. Fmr. researcher at Xanadu. Univ. of Postdam PhD in quantum information.

- **Ryan LaRose**  
  NASA Fellowship PhD student at University of Michigan. Fmr at Alphabet X. Wrote first paper benchmarking quantum software packages.

- **Dan Strano**  
  Full stack web engineer. Lead developer on the qrack quantum simulator.

- **Vincent Russo, PhD**  
  Post-quantum security developer at ISARA. Lead dev on toqito quantum info package. PhD from UWaterloo.

**Advisory Board**

15 volunteer experts in quantum systems & software from:

- IBM
- Xanadu
- Microsoft
- University of Waterloo
- Oak Ridge National Laboratory
- Zapata
- Qosf
- NASA
- USC

**Community**

- 60+ grantees
- 21 countries

**Unitary Fund**

- Full-time
- Part-time
Unitary Fund Programs

Microgrant Program
Small microgrants => big impact

- 47 grants in 21 countries;
- 10+ publications;
- 30+ libraries with 1167 stars, 50+ contributors and ~6k commits
- Helped 12 folks get into quantum tech full time
- 2 new startups, 1 new non-profit

>$100k
Granted to date

Unitary Labs
- Open source research team
- Targeting impactful and open niches that aren’t monetizable by companies

QuTiP: Quantum Toolbox in Python
Unitary Fund is setting up governance and fiscal sponsorship for a critical open source quantum technology package
- >30k annual downloads, >2500 citations
- Sep 2019
downloads 152k total

Mitiq: Error mitigating quantum compiler

```
from qiskit import QuantumCircuit
from mitiq import mitigate_executor
qskt_noisy_sim = mitigate_executor(qs)
circ = QuantumCircuit(1, 1)
for __ in range(120): circ.x(0)
circ.measure(0, 0)

expectation = qskt_noisy_sim(circ)
print(f"Error is \{1 - expectation:.3f}\")
```

Error is 0.0582

Built on Original Research:

Microgrants: Building state of the art open software

**QRack**

SUPPORTED BY UNITARY FUND

an open source, comprehensive, GPU-accelerated framework for simulating universal quantum processors.

Better performance that industry options.

**QuNetSim**

SUPPORTED BY UNITARY FUND

To Stephen DiAdamo to develop the first full featured software stack for quantum network protocols.

**OLSQ**

SUPPORTED BY UNITARY FUND

To Daniel Tan to develop and open source the Optimal Layout Synthesizer for Quantum Computing, OLSQ. This compiler beats other benchmarks on optimal layout of computational qubits onto physical qubits.

---

![Graph](image.png)

**Table 4. Evaluation of QAOA-OLSQ**

<table>
<thead>
<tr>
<th>$M$</th>
<th>$t_{(ket)}$</th>
<th>Depth Swap</th>
<th>TB-OLSQ Depth Swap</th>
<th>Depth Reduction</th>
<th>SWAP Reduction</th>
<th>QAOA-OLSQ Depth SWAP</th>
<th>Depth Reduction</th>
<th>SWAP Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>16</td>
<td>7.3</td>
<td>6.9</td>
<td>7.3</td>
<td>56.7%</td>
<td>0</td>
<td>6.5</td>
<td>5.5</td>
</tr>
<tr>
<td>12</td>
<td>17.8</td>
<td>11.7</td>
<td>8.5</td>
<td>9.3</td>
<td>52.3%</td>
<td>20.4%</td>
<td>5.6</td>
<td>5.8</td>
</tr>
<tr>
<td>14</td>
<td>19.0</td>
<td>13.2</td>
<td>9.0</td>
<td>12.3</td>
<td>52.6%</td>
<td>6.8%</td>
<td>6.0</td>
<td>6.6</td>
</tr>
<tr>
<td>16</td>
<td>21.7</td>
<td>20.2</td>
<td>9.1</td>
<td>13.6</td>
<td>58.2%</td>
<td>32.7%</td>
<td>6.4</td>
<td>6.9</td>
</tr>
<tr>
<td>18</td>
<td>25.5</td>
<td>26.7</td>
<td>8.9</td>
<td>14.5</td>
<td>64.9%</td>
<td>45.7%</td>
<td>6.0</td>
<td>8.3</td>
</tr>
<tr>
<td>20</td>
<td>30.6</td>
<td>37.5</td>
<td>9.3</td>
<td>16.3</td>
<td>68.9%</td>
<td>57.7%</td>
<td>7.2</td>
<td>10.8</td>
</tr>
<tr>
<td>22</td>
<td>29.8</td>
<td>38.4</td>
<td>10.3</td>
<td>17.8</td>
<td>65.4%</td>
<td>53.6%</td>
<td>7.8</td>
<td>14.2</td>
</tr>
</tbody>
</table>

**Geometric Mean**

- $t_{(ket)}$: Time to compute the quantum state
- Depth: Number of quantum gates
- SWAP: Number of SWAP gates
- Reduction: Percentage reduction in quantum cost
Open-source scientific software: research at scale

PyZX: compress quantum programs efficiently with ZX calculus

Effective Compression of Quantum Braided Circuits Aided by ZX-Calculus
Michael Hanks, Marta P. Estarellas, William J. Munro, and Kae Nemoto
Phys. Rev. X 10, 041030 – Published 11 November 2020

We found it helpful to put several of the larger circuits through an initial round of automated reduction with PyZX, an open source Python library designed to reduce, validate, and visualize ZX-calculus diagrams [36]. PyZX applies a recursive, greedy algorithm [37]. Though the strategy of the PyZX library achieves significant reductions, it does not necessarily take into account the additional gate costs mentioned above (for instance, the reduced graphs of PyZX tend to have many Hadamard gates). Nonetheless, having reduced the overall graph size, it became feasible in isolation to tackle the $\pi/4$, $\pi/8$, and Hadamard gates by hand.
Quantum Tech: Quantum computing

Open-source and quantum technologies

Commercialize quantum technologies in five years
Masoud Mohseni, Peter Read, Hartmut Neven, Sergio Boixo, Vasil Denchev, Ryan Babbush, Austin Fowler, Vadim Smelyanskiy & John Martinis

03 March 2017

A Recent Review:
Mark Fingerhuth, Tomáš Babej, and Peter Wittek,
Open source software in quantum computing,
# Quantum Tech: Open Source Libraries

More open-source is empowering broad research in the field

<table>
<thead>
<tr>
<th>Library</th>
<th>Year</th>
<th>Creators</th>
<th>Institution</th>
<th>Language</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QuTiP</td>
<td>2012</td>
<td>Rob Johannson, Paul Nation, Franco Nori</td>
<td>RIKEN</td>
<td>Python</td>
<td>Simulation of open quantum systems; quantum optics, cavity QED.</td>
</tr>
<tr>
<td>QNet</td>
<td>2012</td>
<td>Nikolas Tezak, Michael Goerz, Hideo Mabuchi</td>
<td>Stanford</td>
<td>Python</td>
<td>Computer algebra package for quantum mechanics and photonic quantum networks</td>
</tr>
<tr>
<td>QuantumOptics.jl</td>
<td>2017</td>
<td>Sebastian Krämer et al.</td>
<td>U Innsbruck IQOQI</td>
<td>Julia</td>
<td>Quantum optics and open quantum systems framework inspired by the QO toolbox in Matlab and QuTiP</td>
</tr>
<tr>
<td>ProjectQ</td>
<td>2016</td>
<td>Damian S. Steiger, Thomas Häner, Matthias Troyer</td>
<td>ETH Zurich</td>
<td>Python</td>
<td>Hardware-agnostic framework with compiler and simulator with emulation capabilities.</td>
</tr>
<tr>
<td>OpenFermion</td>
<td>2017</td>
<td>Ryan Babbush et al.</td>
<td>Google (unofficial)</td>
<td>Python</td>
<td>Fermionic potential calculations for quantum chemistry</td>
</tr>
<tr>
<td></td>
<td>2018</td>
<td>Nathan Killoran et al.</td>
<td>Xanadu Inc</td>
<td>Python</td>
<td>Photonic quantum computing with continuous-variable optical circuits</td>
</tr>
</tbody>
</table>

Checkout more open-source projects at [https://qosf.github.io](https://qosf.github.io)
QuTiP: The Quantum Physics Simulator

The Quantum Toolbox in Python

Spin Lattices

Cavity QED

Quantum Optics

Condensed Matter

Quantum Error Correction

Superconducting Circuits

Ion Traps

Optomechanics

gutip.org
Open source for software, knowledge, research

"$5Bn in contributed development costs"*

*linuxfoundation.org/press-release/2015/09/
“Windows and MacOS are products, contrived by engineers in the service of specific companies.

Linux/Unix, by contrast, is not so much a product as it is a painstakingly compiled oral history of the hacker subculture.

It is our Gilgamesh epic.”

– Neal Stephenson
Discovery of nuclear fission

From Wikipedia, the free encyclopedia

Nuclear fission was discovered in December 1938 by chemists Otto Hahn and Fritz Strassmann and physicists Lise Meitner and Otto Robert Frisch. Fission is a nuclear reaction or radioactive decay process in which the nucleus of an atom splits into two or more smaller, lighter nuclei. The fission process often produces gamma rays and releases a very large amount of energy, even by the energetic standards of radioactive decay. Scientists already knew about alpha decay and beta decay, but fission assumed great importance because the discovery that a nuclear chain reaction was possible led to the development of nuclear power and nuclear weapons.

Hahn and Strassmann at the Kaiser Wilhelm Institute for Chemistry in Berlin bombarded uranium with slow neutrons and discovered that barium had been produced. They reported their findings by mail to Meitner in Sweden, who a few months earlier had fled Nazi Germany. Meitner and her nephew Frisch theorised, and then proved, that the uranium nucleus had been split and published their findings in *Nature*. Meitner calculated that the energy released by each disintegration was approximately 200 megar electronvolts, and Frisch observed this. By analogy with the division of biological cells, he named the process "fission". Hahn was awarded the 1944 Nobel Prize in Chemistry for the discovery.
Discovery

Q: "Bohr's liquid drop model had not yet been formulated, so there was no theoretical way to calculate whether it was physically possible for the uranium atoms to break into large pieces." The model was introduced in the previous section as Gamov's liquid drop model. When did Bohr formulate his model?

A: The previous section said: The current model of the nucleus was the liquid drop model first proposed by George Gamov in 1930. His simple and elegant model was refined and developed by Carl Friedrich von Weizsäcker and, after the discovery of the neutron, by Werner Heisenberg and Niels Bohr. Hawkeye7 (discuss) 03:28, 7 August 2020 (UTC)

Still, it kind of appears out of the blue because we, at that point, don't know when Bohr developed his own model. How about "and Niels Bohr (in 193x)."

Added "in 1936". Hawkeye7 (discuss) 05:57, 7 August 2020 (UTC)

Q: Last two paragraphs of "Objections" (why this title?) don't seem too relevant, should they be rather be in their respective biographies? The story is interesting, for sure, but the article is already too long.

A: It tells the reader how this team came together, and why Meitner and Hahn were engaged in digging in to this mystery instead of running the KWI for Chemistry. Meitner's later departure is also foreshadowed. (The title refers to objections to Fermi's claim to have discovered Hawkeye7 (discuss) 03:28, 7 August 2020 (UTC)

It's a biographic digression that I found a bit too long. Interesting, but long. No reason to fail GAN, though.

Q: Last few sentences in "Eureka" - it's unclear if all these dates were in 1939. Frisch conducted exp. in February, but mailed papers (about what?) in January.

A: Added "1939". Hawkeye7 (discuss) 05:57, 7 August 2020 (UTC)
Talk:Discovery of nuclear fission: Revision history

Filter revisions

External tools: Find addition/removal · Find edits by user (Alternate) · Page statistics · Pageviews · Fix dead links

For any version listed below, click on its date to view it. For more help, see Help:Page history and Help:Edit summary. (cur) = difference from current version, (prev) = difference from preceding version, m = minor edit, → = section edit, ← = automatic edit summary

Compare selected revisions

- (cur l prev) 15:54, 22 June 2021 Philoserf (talk | contribs) . . (5,763 bytes) (-622) . (OneClickArchiver archived A Commons file used on this page or its Wikidata item has been nominated for deletion to Talk:Discovery of nuclear fission/Archive 1) (undo)
  - (cur l prev) 15:54, 22 June 2021 Philoserf (talk | contribs) . . (6,385 bytes) (-956) . (OneClickArchiver archived Commons files used on this page or its Wikidata item have been nominated for deletion to Talk:Discovery of nuclear fission/Archive 1) (undo)
- (cur l prev) 15:54, 22 June 2021 Philoserf (talk | contribs) . . (7,341 bytes) (+1) . (Assessment: banner shell (Rater)) (undo)
- (cur l prev) 15:13, 14 May 2021 Wehwalt (talk | contribs) . . (7,340 bytes) (+23) . (Select as TFA for 22 June 2021) (undo)
- (cur l prev) 17:01, 16 April 2021 Community Tech bot (talk | contribs) . . (7,317 bytes) (+622) . (Files used on this page or its Wikidata item are up for deletion) (undo)
- (cur l prev) 18:46, 18 February 2021 BattyBot (talk | contribs) . . (6,695 bytes) (+34) . (→top: Added Template:WikiProject banner shell and other General fixes) (undo) (Tag: AWB)
- (cur l prev) 00:33, 17 November 2020 Morgan695 (talk | contribs) . . (6,661 bytes) (+11) . (undo)
- (cur l prev) 23:26, 1 November 2020 SaintGermie (talk | contribs) . . (6,259 bytes) (+79) . (Template:talk page deletion and formatting Template date)
Open source for open science: COVID-19

An aligned vision

https://nextstrain.org/ncov/global
Open Source
A new era for open source

Open-Source Basics
Read. Download. Deploy.
Definition: You can read the source code (open-source ≠ free).
Examples of open-source: Linux, Android, FireFox, MySQL, LibreOffice, Python.
Open-source deployment is accelerating many end-industries applications.
Open Source
A new era for open source

Open-Source Basics
Read. Download. Deploy.
Definition: You can read the source code (open-source ≠ free).
Examples of open-source: Linux, Android, FireFox, MySQL, LibreOffice, Python.
Open-source deployment is accelerating many end-industries applications.

Features for Developers
You can learn by reading the code and become a better developer.
You can edit the source code (licenses apply) and collaborate to existing projects.
You can submit fixes to bugs, propose improvements.
Open Source
A new era for open source

Open-Source Basics
Read. Download. Deploy.
Definition: You can read the source code (open-source ≠ free).
Examples of open-source: Linux, Android, FireFox, MySQL, LibreOffice, Python.
Open-source deployment is accelerating many end-industries applications.

Features for Developers
You can learn by reading the code and become a better developer.
You can edit the source code (licenses apply) and collaborate to existing projects.
You can submit fixes to bugs, propose improvements.

Open-source for Businesses
Source is generally free but companies can charge for additional services.
Red Hat acquired by IBM for $34 bln (2019).
Machine learning (‘AI’) is driving fast, pervasive adoption of open-source libraries.
Open source and open science
Aligned vision

Open Source Coding
Open Science Research
Open source and open science

Aligned vision

- Allow **access** to the research/project results, sharing knowledge.
- **Collaboratively** advance the field, building upon others’ results.
- Coordinate large and delocalized **teams** working remotely.
- Make supporting data and code available for fast **reproducibility**.
The steady growth of Python
Empowered by a large open-source ecosystem

Projections of future traffic for major programming languages

Future traffic is predicted with an STL model, along with an 80% prediction interval.

Source: David Robinson
Python’s strengths
A community-based programming language
Python’s strengths
A new community-based programming language
Python’s strengths
A modular architecture for well-maintained libraries

Resources

NumPy
Python’s strengths

Interactive Jupyter notebooks run an IDE in your browser

![Python logo](image)

Below we give basic examples on the use of qutip.qip. In the first example the incoherent emission of $N$ driven TLSs is considered. In a two-level system ensemble is a subsystem coupled to another subsystem, a bosonic cavity. Similar considerations apply to the coupling to subsystems (a single qubit, another two-level system ensemble).

```python
In [1]:
import matplotlib.pyplot as plt
import matplotlib as mpl
from matplotlib import cm

from qutip import *
from qutip.pigs import *

import matplotlib.animation as animation
from IPython.display import HTML
from IPython.core.display import Image, display
```

1. $N$ Qubits Dynamics

We study a driven ensemble of $N$ TLSs emitting incoherently,

$$H_{TLS} = \hbar \omega_0 J_z + \hbar \omega J_z$$

$$\dot{\rho} = D_{TLS}(\rho) = -\frac{i}{\hbar}[H_{TLS}, \rho] + \sum_j \frac{\gamma_j}{2} \mathcal{L}_{J_j}(\rho)$$
Open Source
From reproducible data to reusable code.

One of the scikit-learn core developers.
Open Source
From reproducible data to reusable code.

Beyond Reproducibility

https://dx.doi.org/10.6084/m9.figshare.7140050
Reproducibility in Science

A bold stance...

Reproducibility: Online

The public BinderHub instance

your GitHub repo

mybinder.org

interactive browser with your code and computational environment

repo2docker  Jupyter  kubernetes

Google Cloud

Figure from @TuringInst: Kirstie Whitaker, Why you need a reproducible computing environment

https://zenodo.org/record/2598530#.XQDf8tP7Q5g
Make your code count

A Guide to building your open-source scientific computing project in Python.

SUPPORTED BY UNITARY FUND

A cheatsheet to develop a scientific open-source library from scratch.

0: Open  
1: Code  
2: Develop  
3: Test  
4: Pack  
5: Document  
6: Distribute  
7: Publish  
8: Host

Open, Code, Develop  
Test, Pack, Document  
Distribute, Publish, Host

https://github.com/nathanshammah/make-your-code-count
Make your code count

A Guide to building your open-source scientific computing project in Python.

SUPPORTED BY UNITARY FUND

A cheatsheet to develop a scientific open-source library from scratch.

0: Open  3: Test  6: Distribute
1: Code  4: Pack  7: Publish
2: Develop  5: Document  8: Host

Open, Code, Develop  Test, Pack, Document  Distribute, Publish, Host

python  git  Travis CI  python Package Index
GitLab  GitHub  SPHINX  Conda
Read the Docs  zenodo  binder

https://github.com/nathanshammah/make-your-code-count
Open Science through Open Source: Making code count

The tools of open source

**Code & Testing**
- GitHub
- GitLab
- Travis CI

**Documentation**
- Sphinx
- Read the Docs

**Publication**
- Conda
- Zenodo

An OpenAIRE + CERN project
Python: Function, Class, Objects

```python
def num_dicke_states(N):
    """
    Calculate the number of Dicke states.
    
    Parameters
    ----------
    N: int
        The number of two-level systems.
    
    Returns
    ------
    nds: int
        The number of Dicke states.
    """
    nds = (N/2 + 1)**2 - (N % 2)/4
    return int(nds)

class Dicke(object):
    """
    A Dicke class defining an ensemble of spins
    
    Parameters
    ----------
    N: int
        Number of two level systems
    """
    def __init__(self, N, H=None, emission=0.):
        self.N = N
        self.H = H
        self.emission = emission

    def liouvillian(self):
        """
        Calculate the Liouvillian superoperator.
        """
```

```
>> from pigs import num_dicke_states
>> print(num_dicke_states(30))
256

>> from pigs import Dicke
>> ensemble = Dicke(30, emission=0.1)
>> L = ensemble.liouvillian()
```
Git and GitHub

- Version control
- Collaborative code
- Online and open-source

```
paper.tex
paper_suggestions2.tex
paper2.tex
  paper_deleted_sec1.tex
paper_paul_details.tex
  paper_final.tex
  paper_final2.tex
```

```
<table>
<thead>
<tr>
<th>COMMENT</th>
<th>DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATED MAIN LOOP &amp; TIMING CONTROL</td>
<td>14 HOURS AGO</td>
</tr>
<tr>
<td>ENABLED CONFIG FILE PARSING</td>
<td>9 HOURS AGO</td>
</tr>
<tr>
<td>MISC BUGFIXES</td>
<td>5 HOURS AGO</td>
</tr>
<tr>
<td>CODE ADDITIONS/EDITS</td>
<td>4 HOURS AGO</td>
</tr>
<tr>
<td>MORE CODE</td>
<td>4 HOURS AGO</td>
</tr>
<tr>
<td>HERE HAVE CODE</td>
<td>4 HOURS AGO</td>
</tr>
<tr>
<td>AAAAAAAAA</td>
<td>3 HOURS AGO</td>
</tr>
<tr>
<td>ADKFJ5LDFJSKFLF</td>
<td>3 HOURS AGO</td>
</tr>
<tr>
<td>MY HANDS ARE TYPING WORDS</td>
<td>2 HOURS AGO</td>
</tr>
<tr>
<td>HAAAAAAANDS</td>
<td>2 HOURS AGO</td>
</tr>
</tbody>
</table>

AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.
```
Git: Setup

Online repository → Your fork (copy) on Github

Local copy

>> git clone github.com/paul/qutip
>> git remote add upstream github.com/qutip/qutip
Introduction

Permutational Invariant Quantum Solver (PIQS)

PIQS is an open-source Python solver to study the exact Lindbladian dynamics of open quantum systems consisting of identical qubits.

In the case where local processes are included in the model of a system’s dynamics, numerical simulation requires dealing with density matrices of size $2^N$. This becomes infeasible for a large number of qubits. We can simplify the calculations by exploiting the permutational invariance of indistinguishable quantum particles which allows the user to study hundreds of qubits.

Integrated with QuTiP
Documentation

Auto generate with Sphinx

```
>> sphinx-quickstart
|-- doc/
   |-- MakeFile
   |-- make.bat
   |-- source
      |-- conf.py
      |-- conf.py
      |-- index.rst
      |-- intro.rst
```

Edit configurations

```
doc/source/conf.py
# -- Project information
project = 'piqs'
#
# The short X.Y version
version = ''
#
# The full version
release = '1.0'
```

Generate documentation

```
>> make html
```

Read the docs - host online

---

**Introduction**

**Permutational Invariant Quantum Solver (PIQS)**

PIQS is an open-source Python solver to study the exact Lindbladian dynamics of open quantum systems consisting of identical qubits.
Distributing package: pip, conda

**PyPI**  [Icon]

**conda-forge**  [Icon]

---

**Add setup information**

**piqs/setup.py**

```python
from setuptools import setup
setup(name='piqs',
      version='0.1',
      description='Permutational Invariant ...
```

**meta.yml (recipe)**

```yaml
{% set name = "piqs" %}
{% set version = "1.0" %}
package:
  name: {{ name|lower }}
```

---

**Make a package and upload**

```sh
>> python setup.py register sdist upload

Install from pip

>> pip install piqs

Add to a personal channel or conda-forge

>> conda build

>> conda install -c www.github.com/paul
```
Publishing online: Release

github.com/nathanshammah/piqs
Permutational Invariance Quantum Solver for Lindblad open quantum system dynamics

142 commits  7 branches  1 release  2 contributors

Draft a new release
Publishing online: Zenodo

1. Flip the switch
Select the repository you want to preserve, and toggle the switch below to turn on automatic preservation of your software.

2. Create a release
Go to GitHub and create a release. Zenodo will automatically download a zip-ball of each new release and register a DOI.

3. Get the badge
After your first release, a DOI badge that you can include in GitHub README will appear next to your repository below.

DOI 10.5281/zenodo.8475
(example)
Thank you!

discord.unitary.fund

nathan@unitary.fund