HEP Packaging Coordination: Reproducible reuse by default

Matthew Feickert
University of Wisconsin-Madison

Reinterpretation Forum Workshop
February 27th, 2025

FAIROS-HEP

Goals for computational environment preservation

- For analysis preservation and reinterpretation we care not only about preserving the data and the source code, but also the **full** computational environment.
- Ideal way to achieve this is by having an **environment manifest**
  - 💡File that specifies the **known software requirements** of your analysis
- …and an **environment lock file**
  - 💡Hash level list of the exact environment actually “solved”
- Software products should be **publicly accessible**, long term **archivable**, and hash-level **verifiable**
  - You shouldn’t have to guess or ask if you have things right. This should be programmatic and automatic.
  - Requires long term public infrastructure

Quick setup for demo later

There's going to be a demo later on in this talk! :D

To follow along all you need to do is install `pixi` and then restart your shell.

- Linux/macOS

  ```
  curl -fsSL https://pixi.sh/install.sh | bash
  ```

- Windows

  ```
  powershell -ExecutionPolicy ByPass -c "irm -useb https://pixi.sh/install.ps1 | iex"
  ```

`pixi` is a single securely signed Rust binary, it won't hurt you, and you can trivially remove it from your computer later if you want.

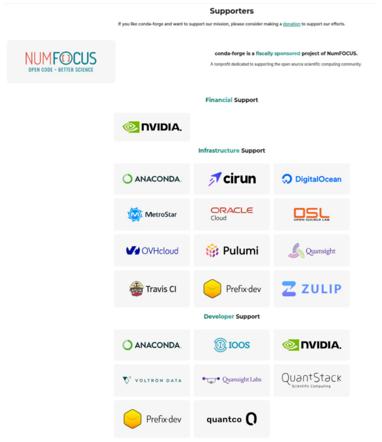
If you don't believe me you can also use a Linux container:

```
docker run --rm -ti ghcr.io/prefix-dev/pixi:latest
```

What is conda-forge?

- Open source global community software **build infrastructure** and **package registry** system
- Support building arbitrary software for **Linux** (x86_64, aarch64, ppc64le), **macOS** (x86_64, arm64), **Windows** (x86_64, arm64) as “conda” package
  - Conda packages are zipfiles of tar.zst directory trees to be unpacked with additional environment magic.
- Built distributions are publicly and permanently hosted on Anaconda.org on the `conda-forge` channel
  - Conda-forge is immutable after upload. Can be marked as “broken” but not removed (unless legally required)
- Software artifacts on conda-forge are installable by anything that knows how to interact with a conda package: `conda`/`mamba`/`pixi`
- Co-founded by (now) CERN’s [Philip Elson]!
  - c.f. Phil’s SciPy 2016 talk [Community-Powered Packaging with conda-forge]

CONDA-FORGE



Supporters

If you like conda-forge and want to support our mission, please consider making a donation to support our efforts.

conda-forge is a fiscally sponsored project of NumFOCUS.

A nonprofit dedicated to supporting the open source scientific computing community.

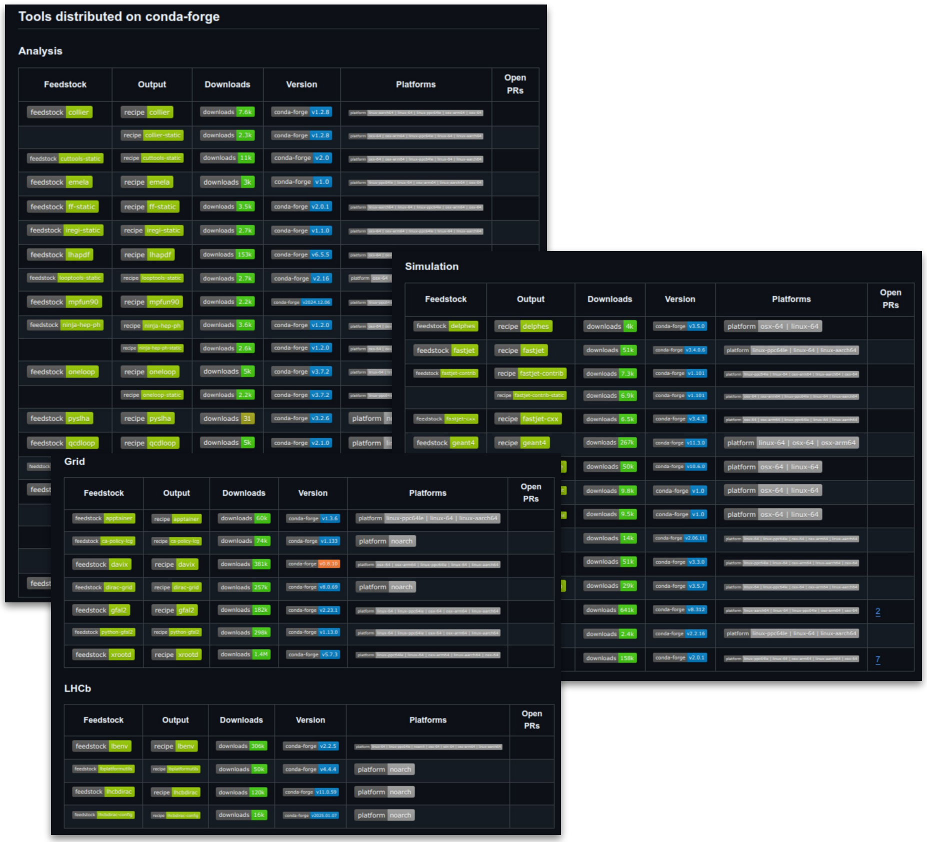
Financial Support

Infrastructure Support

Developer Support

HEP Packaging Coordination

- Community project on GitHub to get **as much HEP software as possible on conda-forge**
    - High quality research software should be trivial to install and provide platform specific optimized binary builds by default
    - Building from source should be an option for development and debugging, not the default
- In the **context of reinterpretation**, if you want to preserve your analysis environment, you preserve the whole thing, not just the source code.



Tools distributed on conda-forge

Analysis

Feedstock	Output	Downloads	Version	Platforms	Open PRs
feedstock collier	recipe collier	downloads 7.6k	conda-forge v1.2.8	platform linux-aarch64 | linux-64 | linux-ppc64le | osx-arm64 | osx-64	
	recipe collier-static	downloads 2.3k	conda-forge v1.2.8	platform osx-64 | osx-arm64 | linux-ppc64le | linux-64 | linux-aarch64	
feedstock cuttools-static	recipe cuttools-static	downloads 11k	conda-forge v2.0	platform osx-64 | osx-arm64 | linux-ppc64le | linux-64 | linux-aarch64	
feedstock emela	recipe emela	downloads 3k	conda-forge v1.0	platform linux-ppc64le | linux-64 | osx-arm64 | linux-aarch64 | osx-64	
feedstock ff-static	recipe ff-static	downloads 3.5k	conda-forge v2.0.1	platform linux-aarch64 | linux-64 | linux-ppc64le | osx-arm64 | osx-64	
feedstock iregi-static	recipe iregi-static	downloads 2.7k	conda-forge v1.1.0	platform osx-64 | osx-arm64 | linux-ppc64le | linux-64 | linux-aarch64	
feedstock lhapdf	recipe lhapdf	downloads 153k	conda-forge v6.5.5	[illegible]	
feedstock looptools-static	recipe looptools-static	downloads 2.7k	conda-forge v2.16	[illegible]	
feedstock mpfun90	recipe mpfun90	downloads 2.2k	conda-forge v2024.12.06	[illegible]	
feedstock ninja-hep-ph	recipe ninja-hep-ph	downloads 3.6k	conda-forge v1.2.0	[illegible]	
	recipe ninja-hep-ph-static	downloads 2.6k	conda-forge v1.2.0	[illegible]	
feedstock oneloop	recipe oneloop	downloads 5k	conda-forge v3.7.2	[illegible]	
	recipe oneloop-static	downloads 2.2k	conda-forge v3.7.2	[illegible]	
feedstock pyslha	recipe pyslha	downloads 31	conda-forge v3.2.6	[illegible]	
feedstock qcdloop	recipe qcdloop	downloads 5k	conda-forge v2.1.0	[illegible]	

Simulation

Feedstock	Output	Downloads	Version	Platforms	Open PRs
feedstock delphes	recipe delphes	downloads 4k	conda-forge v3.5.0	platform osx-64 | linux-64	
feedstock fastjet	recipe fastjet	downloads 51k	conda-forge v3.4.0.6	platform linux-ppc64le | linux-64 | linux-aarch64	
feedstock fastjet-contrib	recipe fastjet-contrib	downloads 7.3k	conda-forge v1.101	platform linux-ppc64le | linux-64 | osx-arm64 | linux-aarch64 | osx-64	
	recipe fastjet-contrib-static	downloads 6.9k	conda-forge v1.101	platform osx-64 | osx-arm64 | linux-ppc64le | linux-64 | linux-aarch64	
feedstock fastjet-cxx	recipe fastjet-cxx	downloads 6.5k	conda-forge v3.4.3	platform osx-64 | osx-arm64 | linux-ppc64le | linux-64 | linux-aarch64	
feedstock geant4	recipe geant4	downloads 267k	conda-forge v11.3.0	platform linux-64 | osx-64 | osx-arm64	
[illegible]	[illegible]	downloads 50k	conda-forge v10.6.0	platform osx-64 | linux-64	
[illegible]	[illegible]	downloads 9.8k	conda-forge v1.0	platform osx-64 | linux-64	
[illegible]	[illegible]	downloads 9.5k	conda-forge v1.0	platform osx-64 | linux-64	
[illegible]	[illegible]	downloads 14k	conda-forge v2.06.11	platform linux-64 | linux-ppc64le | osx-64 | osx-arm64 | linux-aarch64	
[illegible]	[illegible]	downloads 51k	conda-forge v3.3.0	platform linux-ppc64le | linux-64 | osx-arm64 | linux-aarch64 | osx-64	
[illegible]	[illegible]	downloads 29k	conda-forge v3.5.7	platform linux-64 | linux-ppc64le | osx-64 | osx-arm64 | linux-aarch64	
[illegible]	[illegible]	downloads 641k	conda-forge v8.312	platform linux-aarch64 | linux-64 | linux-ppc64le | osx-arm64 | osx-64	2
[illegible]	[illegible]	downloads 2.4k	conda-forge v2.2.16	platform linux-ppc64le | linux-64 | linux-aarch64	
[illegible]	[illegible]	downloads 158k	conda-forge v2.0.1	platform linux-ppc64le | linux-64 | osx-arm64 | linux-aarch64 | osx-64	7

Grid

Feedstock	Output	Downloads	Version	Platforms	Open PRs
feedstock apptainer	recipe apptainer	downloads 60k	conda-forge v1.3.6	platform linux-ppc64le | linux-64 | linux-aarch64	
feedstock ca-policy-lcg	recipe ca-policy-lcg	downloads 74k	conda-forge v1.133	platform noarch	
feedstock davix	recipe davix	downloads 381k	conda-forge v0.8.10	platform osx-64 | osx-arm64 | linux-ppc64le | linux-64 | linux-aarch64	
feedstock dirac-grid	recipe dirac-grid	downloads 257k	conda-forge v8.0.69	platform noarch	
feedstock gfal2	recipe gfal2	downloads 182k	conda-forge v2.23.1	platform linux-64 | linux-ppc64le | osx-64 | osx-arm64 | linux-aarch64	
feedstock python-gfal2	recipe python-gfal2	downloads 298k	conda-forge v1.13.0	platform linux-64 | linux-ppc64le | osx-64 | osx-arm64 | linux-aarch64	
feedstock xrootd	recipe xrootd	downloads 1.4M	conda-forge v5.7.3	platform linux-ppc64le | linux-64 | osx-arm64 | linux-aarch64 | osx-64	

LHCb

Feedstock	Output	Downloads	Version	Platforms	Open PRs
feedstock lbenv	recipe lbenv	downloads 306k	conda-forge v2.2.5	platform linux-64 | linux-ppc64le | noarch | osx-64 | win-64 | osx-arm64 | linux-aarch64	
feedstock lbplatformutils	recipe lbplatformutils	downloads 50k	conda-forge v4.4.4	platform noarch	
feedstock lhcbdirac	recipe lhcbdirac	downloads 120k	conda-forge v11.0.59	platform noarch	
feedstock lhcbdirac-config	recipe lhcbdirac-config	downloads 16k	conda-forge v2025.01.07	platform noarch	

https://github.com/hep-packaging-coordination

HEP Packaging Coordination

The Road to Differentiable and Probabilistic Programming in Fundamental Physics, Munich, 2023
F(X)
F'(X)
F''(X)

[Matthew Feickert]
(ATLAS, IRIS-HEP)

[Chris Burr]
(LHCb, conda-forge core)

[Giordon Stark]
(ATLAS)

[Henry Schreiner]
(IRIS-HEP, PyPA)

…and anyone else! Open community project, looking for volunteers.

Other existing packaging projects (why another?)

There have been ~similar efforts in the past

An incomplete list:

- Gentoo sci-physics
- HEPrpms
- Homebrew-hep
- Build system managers (HPCs, builds from source)
  - Spack
  - EasyBuild

Critical difference (why you should care)

Conda-forge is:

- Multi-platform
  - Linux (x86_64, aarch64, ppc64le)
    macOS (x86_64, arm64)
    Windows (x86_64, arm64)
- Global scale community software
  - Contribute to maintenance/effort, not our sole responsibility
- Fully automated
  - Automatic ABI migrations and rebuilds without losing past distributions

Thanks to Alexander Puck Neuwirth for making me aware of most of these

The role of Linux containers

Different solutions for different problems:
Linux containers are distribution methods, not packaging technologies.

Linux containers for when you want to

- Deploy an instantiated, bespoke environment as a single executable on a known platform
- Sandbox an environment from the rest of the operating system

Powerful and necessary: Note that conda-forge itself heavily uses Linux containers!

For ATLAS RECAST, Linux containers are critical and have enabled large successes (I think they're great!)

- c.f. ATLAS Run 2 pMSSM analysis

However,

- Most users find them confusing
- Composing environments is cumbersome
- Sometime you get "the kitchen sink", not something well scoped/designed

The role of Linux containers

Realistically, different amounts of work between different solutions.

conda-forge

CONDA-FORGE

- First build effort:
  - PyPI (trivial)
  - Bespoke C++/Fortran (some)
  - Heavily patched and vendored custom software (large)
- Next build and rebuilds
  - “Easy” to “push button”
- Multi-platform: support is easy
- Composition: Trivial
  - We’ll see this in a moment with `pixi`

Linux containers

- First build effort:
  - Same as every other build (depends)
- Next build and rebuilds
  - Same as every other build (depends)
- Multi-platform: Complex
  - requires doing cross-builds to other platforms with emulation or doing separate builds and then joining manifests. All “by hand”.
- Composition: Complex

What do I (someone who develops tools) need to do?

- This is all **opt in**
  - Advantages are **worth it** in my experience
- Can you **write a build script**?
  - If so, you are ~90% of the the way to having your software be packaged on conda-forge.
- Have all your **dependencies on conda-forge** already (bootstrap system)
  - Might have to talk to some people (or package them yourself)
- Have your **source code** come from a **stable and static official source**
  - No go: Distributions on website that may disappear without warning with no long term archive. **Don't do this!**
  - Not great: Tarballs on a distribution website with no long term preservation
  - Better: Public version controlled system with 1-to-1 release tags
  - Best: "Better" solution with Zenodo archive for "permanent" archive access
- **Ask for help** from people in HEP Packaging Coordination.
  - Some things will not be inherently obvious, but those things are also usually quality of life things that you didn't think to expect (e.g. selectors, multi-platform environment variables, recipe linting, validation checks)

What do I (a user of tools) need to do?

- Very little
  - Know what your **software requirements** are
  - Know what **platforms** you want your analysis to run on
  - Have **pixi installed** (lives in user space so you can do this yourself anywhere)
    - Though talk to your cluster admins for more efficient caching use
- This brings us to the demo!

Live Demo

On any Linux or macOS machine:

```
$ mkdir rif-workshop-example

$ cd rif-workshop-example

$ pixi init

$ time pixi add cxx-compiler fortran-compiler lhapdf pythia8
smodels

$ pixi shell
```

Alternatively go to:

https://github.com/matthewfeickert-talks/talk-reinterpretation-forum-workshop-2025

What just happened? (What is pixi?)

- Pixi is what I call an “environment manager” rather than a “package manager”
  - Different philosophy of managing your *project* environment rather than a globally active environment of tools
  - Written in Rust (very fast)
  - Intelligent caching
  - Can talk to/install from any **conda package index** (e.g. conda-forge) and any **Python package index** (e.g. PyPI)
- Has **high level semantics** ✅ intended for scientists
- **High level manifest** ✅ that supports multiple environments, task runner
- Automatically and non-optionally generates a **hash level lock file**✅

(Known) Limitations

- Entire environment needs to be provided so far
  - **Normally a feature**, but you can imagine situations in which you want to extend an existing experiment specific environment (e.g. LCG view or ATLAS Analysis Release) that is hardcoded to certain binaries (e.g. Python).
  - Chris Burr + others have done a lot of work in LHCb to make this a better experience. Can learn from them.
- Builds are official versions, but LHC experiments realistically use heavily patched releases.
  - **Normally a feature**, but this means you aren't getting official production versions of tools used by e.g. ATLAS/CMS. That's probably fine for most use cases, and also what rest of the community expects from tools as well.
- Lots of small files in user space on distributed systems / HPCs can be problematic
  - These systems optimized for reading large input files.
  - Workaround: Distribute environment in Linux container image to system (e.g. execute with Apptainer)

Where to go from here?

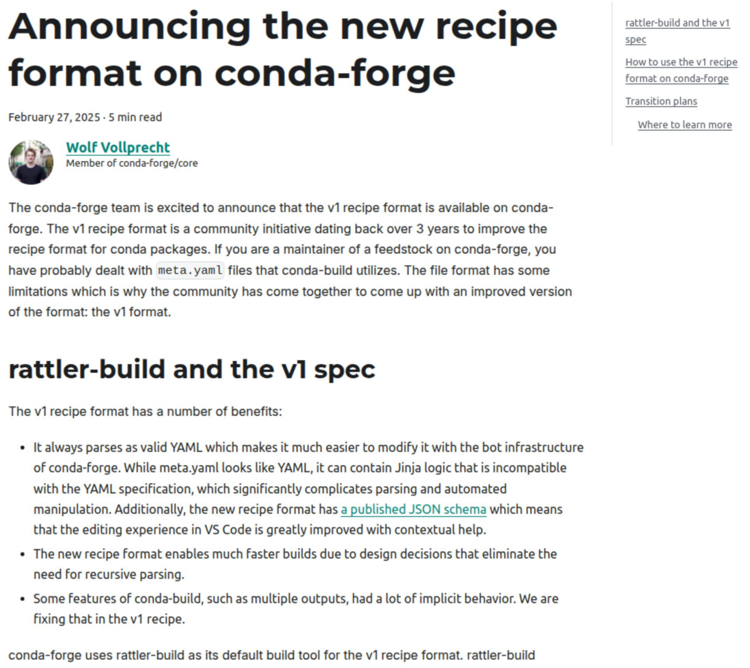
- FAIROS-HEP is sponsoring a HEP Packaging Hackathon in 2025
- Members of HEP Packaging Coordination and conda-forge core team will lead tutorials on packaging HEP software for conda-forge and use tutorials with pixi
- Upcoming and details TBD, so stay tuned, but you can get started *now*.

Summary

- For **analysis reinterpretation and reuse**, want **full computational environment preservation**, but also for it to be
  - easy to create
  - with good semantics
  - fully reproducible
- Packaging tools on conda-forge gets us the **distribution**, **automation, and community support**
  - For machine learning, also have the full CUDA stack on conda-forge thanks to the NVIDIA open source team
- Using pixi gets us the **scientist level semantics**, with **cross-platform manifest and lock file**

Backup

New! conda-forge recipe spec v1 release announcement



Announcing the new recipe format on conda-forge

February 27, 2025 · 5 min read

Wolf Vollprecht
Member of conda-forge/core

The conda-forge team is excited to announce that the v1 recipe format is available on conda-forge. The v1 recipe format is a community initiative dating back over 3 years to improve the recipe format for conda packages. If you are a maintainer of a feedstock on conda-forge, you have probably dealt with `meta.yaml` files that conda-build utilizes. The file format has some limitations which is why the community has come together to come up with an improved version of the format: the v1 format.

rattler-build and the v1 spec

The v1 recipe format has a number of benefits:

- It always parses as valid YAML which makes it much easier to modify it with the bot infrastructure of conda-forge. While meta.yaml looks like YAML, it can contain Jinja logic that is incompatible with the YAML specification, which significantly complicates parsing and automated manipulation. Additionally, the new recipe format has a published JSON schema which means that the editing experience in VS Code is greatly improved with contextual help.
- The new recipe format enables much faster builds due to design decisions that eliminate the need for recursive parsing.
- Some features of conda-build, such as multiple outputs, had a lot of implicit behavior. We are fixing that in the v1 recipe.

conda-forge uses rattler-build as its default build tool for the v1 recipe format. rattler-build

https://conda-forge.org/blog/2025/02/27/conda-forge-v1-recipe-support/

Same day as this talk, which feel fortuitous!

Example: Process of packaging on conda-forge

Adding a package to conda-forge

1. Fork https://github.com/conda-forge/staged-recipes on GitHub
2. Make a new branch from `main` for your package's recipe.
3. Make a new folder in recipes for your package.
   a. Look at the example recipe, conda-forge documentation, and the FAQ for help, and ask HEP Packaging Coordination if you have any questions.
4. Add your package's recipe, patching if necessary
   a. If on PyPI can probably just do `pixi run pypi <PyPI package name>`
5. Lint the recipe with `pixi run lint`
6. Build the recipe locally to do a quick test for your OS with `pixi run build-linux` / `pixi run build-osx`
7. Push your changes to your fork and open a pull request to `main`, validate that the CI builds pass, follow checklist, request review.

Example: SModelS

Example: Process of packaging on conda-forge

Adding additional platform builds (read the docs!)

1. After feedstock creation and rerender, fork the feedstock repo
   a. All development must be done on **personal forks** (not **orgs** and not on the original)
2. Open a PR to the conda-forge-pinning-feedstock repository that add your feedstock to `arch_rebuild.txt` (for linux-aarch64, linux-ppc64le) and to osx_arm64.txt (for osx-arm64).
3. Can manually add these as well, but then you have to manually maintain these files.

Example: Process of packaging on conda-forge

Updating to new releases (read the docs!)

1. When a new release tag of your package's "upstream" source code is released (e.g. new tag on GitHub, tarball on HEPForge (legacy)) regro-cf-autotick-bot will automatically discover and pick this change up within 24 hours.
2. `regro-cf-autotick-bot` will then open up a PR with the new version, rerender with conda-smithy, and tag you for review.
3. If the PR needs adjustments, you can push directly to regro-cf-autotick-bot's fork's branch or close the PR and manually migrate it yourself in a new PR.

CVMFS

CVMFS is an interesting and successful HEP specific solution but as is in the name it is a **distributed file system**. It is **not** a global package index.

When you select software from CVMFS you are selecting entire **distributions** of software that you may or may not actually want (e.g. LCG views). There's no way reasonable and reproducible way to create long term bespoke distribution selections.

CVMFS works quite well for C++ libraries. It is a pretty miserable experience for Python, and there's no support for Julia or other modern languages that we're seeing in our ecosystem.