



Benchmark Development Tutorial: Overview

Dmitry Kalinkin

University of Kentucky

Goals for benchmarking

Setting up analysis software to be automatically periodically run should allow us to:

- » Detect software regressions – allow for faster development, ease software upgrades
- » Collaborative development – publish things to avoid unnecessary duplicate work
- » Up to date calibration constants, ML artifacts
- » Analysis archival and preservation, validation against beam test results
- » Synergy with the simulation campaign production

There are some upfront costs in setup of the automation.

Testing and benchmarking in ePIC

- » Development tests for epic geometry and for EICrecon
- » Detector benchmarks
- » Physics benchmarks

required contribution,
strict standards,
early feedback
low statistics



voluntary contribution,
relaxed standards,
late feedback,
large statistics

ePIC benchmarks on eicweb

Now available for contribution on GitHub, without an eicweb account:

- » https://eicweb.phy.anl.gov/EIC/benchmarks/detector_benchmarks
https://github.com/eic/detector_benchmarks ← contribute here
- » https://eicweb.phy.anl.gov/EIC/benchmarks/physics_benchmarks
https://github.com/eic/physics_benchmarks ← contribute here

We need more and better benchmarks to be added. We hope you can help!

Defining automation

The user experience on eicweb is not ideal. Defining analysis in `.gitlab-ci.yml` is a bit involved and doesn't allow to test changes locally.

Pilot project is to use **Snakemake** for analysis workflow definition.

- » allows to run small and large workflows **locally**
- » can submit **batch jobs** on computing grids (HTCondor, Slurm, ...)
- » less confusing than shell scripts
- » caches intermediate steps – ideal for **quick iteration** development

Try it in Exercise 2 and let us know what you think!

The Tutorial

<https://eic.github.io/tutorial-developing-benchmarks/>

00:00

1. [Excercise 1: Setting up your first benchmark](#)

00:20

2. [Excercise 2: Workflow management with Snakemake](#)

How does one share data analysis workflows?

Online support [~collab-mtg-jan24-tutorials](#) during Jan 9 2024,
[~Helpdesk](#) after that (make sure to refernce the tutorial)

Further work

An example of state of art benchmark with Snakemake and eicweb is https://github.com/eic/physics_benchmarks/tree/master/benchmarks/diffractive_vm

- » Capable of campaign processing, and works on CI with online simulation (a "smoke test")
- » You should be now able to understand how it's setup!

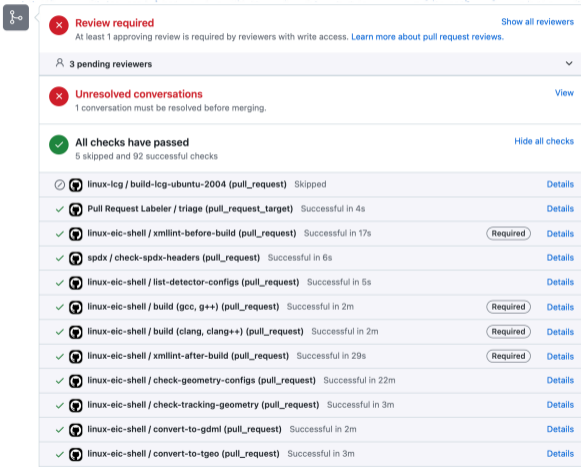
Not covered:

- » Artifacts upload (write user-facing output to `results/`) – visual results
- » `common_bench` - numerical benchmark results, pass/fail conditions

Backup

epic geometry tests

https://github.com/eic/epic/pulls



The screenshot shows a GitHub pull request interface. At the top, there is a red 'Review required' status with a sub-message: 'At least 1 approving review is required by reviewers with write access. Learn more about pull request reviews.' Below this, it indicates '3 pending reviewers'. A red 'Unresolved conversations' status follows, with the message: '1 conversation must be resolved before merging.' A green 'All checks have passed' status is shown next, with the message: '5 skipped and 92 successful checks'. The bottom section is a list of 14 CI/CD checks, each with a status icon, a name, a duration, and a 'Details' link. Some checks are marked as 'Required'.

Status	Check Name	Duration	Required
Skipped	linux-icg / build-icg-ubuntu-2004 (pull_request)	Skipped	No
Successful	Pull Request Labeler / triage (pull_request_target)	Successful in 4s	No
Successful	linux-eic-shell / xmllint-before-build (pull_request)	Successful in 17s	Yes
Successful	spdx / check-spdx-headers (pull_request)	Successful in 6s	No
Successful	linux-eic-shell / list-detector-configs (pull_request)	Successful in 5s	No
Successful	linux-eic-shell / build (gcc, g++) (pull_request)	Successful in 2m	Yes
Successful	linux-eic-shell / build (clang, clang++) (pull_request)	Successful in 2m	Yes
Successful	linux-eic-shell / xmllint-after-build (pull_request)	Successful in 29s	Yes
Successful	linux-eic-shell / check-geometry-configs (pull_request)	Successful in 22m	No
Successful	linux-eic-shell / check-tracking-geometry (pull_request)	Successful in 3m	No
Successful	linux-eic-shell / convert-to-gdml (pull_request)	Successful in 2m	No
Successful	linux-eic-shell / convert-to-tgeo (pull_request)	Successful in 3m	No

- » Compile with gcc and clang
- » Run TGeo and Geant4 overlap checks for all configurations
- » Run ACTS checks
- » Produce GDML, ROOT(TGeo) geometry files
- » Render dawn views
- » Trigger running of detector and physics benchmarks on eicweb, status is reported back

epic geometry tests

<https://github.com/eic/epic/pulls>

✓		eicweb/physics_benchmarks (epic_brycecanyon) — Succeeded! Fri Sep 1 06:02:32 PM EDT 2023	Required	Details
✓		eicweb/physics_benchmarks (epic_craterlake) — Succeeded! Fri Sep 1 05:48:16 PM EDT 2023	Required	Details
✓		eicweb/reconstruction_benchmarks (epic_brycecanyon) — Succeeded! Fri Sep 1 05:30:47 PM E...	Required	Details
✓		eicweb/reconstruction_benchmarks (epic_craterlake) — Succeeded! Fri Sep 1 05:27:34 PM EDT ...	Required	Details

EIC > benchmarks > physics_benchmarks > Pipelines > #70417

add diffractive_vmp

passed Wouter Deconinck triggered pipeline for commit [fd283781](#) finished 2 weeks ago

For [master](#)

60 71 Jobs 45 minutes 12 seconds, queued for 3 seconds

Pipeline Needs Jobs 71 Tests 0

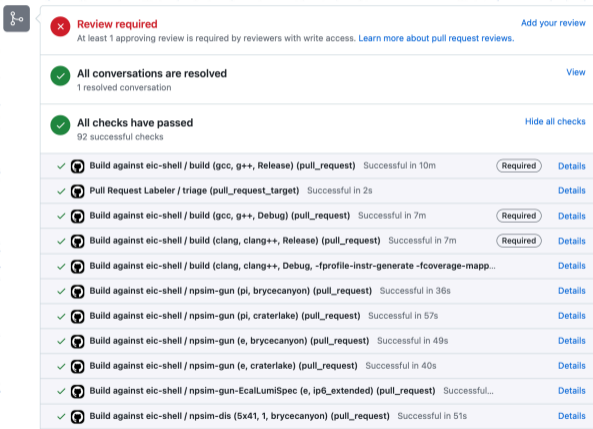
Group jobs by

simulate	reconstruct	analyze	collect
✓ backgrounds:synchrotron:simulate 2	✓ single:reconstruct 6	✓ single:analyze 6	✓ backgrounds:resu
✓ diffractive_vmp:simulate 2			✓ diffractive_vmp:res
✓ dis:simulate 11			✓ dis:results
✓ dvcs:simulate 2			✓ dvcs:results
✓ single:simulate 6			✓ single:results
✓ tcs:simulate 3			✓ tcs:results
✓ u_omega:simulate 2			✓ u_omega:results

- » Compile with gcc and clang
- » Run TGeo and Geant4 overlap checks for all configurations
- » Run ACTS checks
- » Produce GDML, ROOT(TGeo) geometry files
- » Render dawn views
- » Trigger running of detector and physics benchmarks on eicweb, status is reported back

EICrecon tests

<https://github.com/eic/EICrecon/pulls>



Review required [Add your review](#)
At least 1 approving review is required by reviewers with write access. [Learn more about pull request reviews.](#)

All conversations are resolved [View](#)
1 resolved conversation

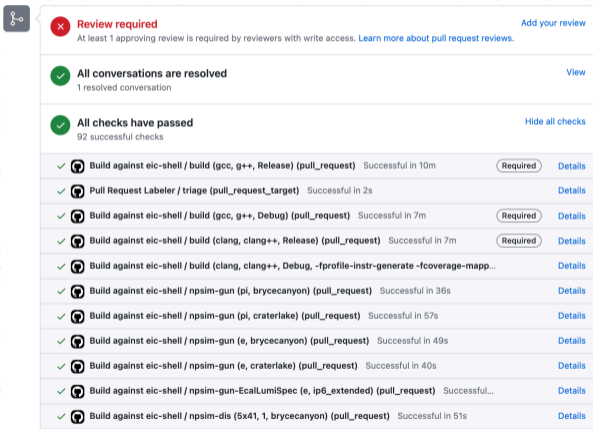
All checks have passed [Hide all checks](#)
92 successful checks

✓	Build against eic-shell / build (gcc, g++, Release) (pull_request)	Successful in 10m	Required	Details
✓	Pull Request Labeler / triage (pull_request_target)	Successful in 2s		Details
✓	Build against eic-shell / build (gcc, g++, Debug) (pull_request)	Successful in 7m	Required	Details
✓	Build against eic-shell / build (clang, clang++, Release) (pull_request)	Successful in 7m	Required	Details
✓	Build against eic-shell / build (clang, clang++, Debug, -fprofile-instr-generate -fcoverage-mapp...			Details
✓	Build against eic-shell / npsim-gun (pi, bryce canyon) (pull_request)	Successful in 36s		Details
✓	Build against eic-shell / npsim-gun (pi, craterlake) (pull_request)	Successful in 57s		Details
✓	Build against eic-shell / npsim-gun (e, bryce canyon) (pull_request)	Successful in 49s		Details
✓	Build against eic-shell / npsim-gun (e, craterlake) (pull_request)	Successful in 40s		Details
✓	Build against eic-shell / npsim-gun-EcallLumiSpec (e, ip6_extended) (pull_request)	Successful...		Details
✓	Build against eic-shell / npsim-dis (5x41, 1, bryce canyon) (pull_request)	Successful in 51s		Details

- » Compile with gcc and clang
- » Static analysis and code style (clang-tidy, IWYU)
- » With AddressSanitizer and UBSanitizer
- » Run unit tests
- » Run simulation and reconstruction for gun and DIS (100 events)
- » Run JANA-based benchmarks
- » Upload artifacts (EDM4hep sim, EDM4eic reco, jana factory parameters, janadot, coverage report, doxygen)
- » Compare to reco EDM4eic to artifact from the base branch

EICrecon tests

<https://github.com/eic/EICrecon/pulls>



Review required [Add your review](#)
At least 1 approving review is required by reviewers with write access. [Learn more about pull request reviews.](#)

All conversations are resolved [View](#)
1 resolved conversation

All checks have passed [Hide all checks](#)
92 successful checks

✓	Build against eic-shell / build (gcc, g++, Release) (pull_request)	Successful in 10m	Required	Details
✓	Pull Request Labeler / triage (pull_request_target)	Successful in 2s		Details
✓	Build against eic-shell / build (gcc, g++, Debug) (pull_request)	Successful in 7m	Required	Details
✓	Build against eic-shell / build (clang, clang++, Release) (pull_request)	Successful in 7m	Required	Details
✓	Build against eic-shell / build (clang, clang++, Debug, -fprofile-instr-generate -fcoverage-mapp...			Details
✓	Build against eic-shell / npsim-gun (pi, brycecanon) (pull_request)	Successful in 36s		Details
✓	Build against eic-shell / npsim-gun (pi, craterlake) (pull_request)	Successful in 57s		Details
✓	Build against eic-shell / npsim-gun (e, brycecanon) (pull_request)	Successful in 49s		Details
✓	Build against eic-shell / npsim-gun (e, craterlake) (pull_request)	Successful in 40s		Details
✓	Build against eic-shell / npsim-gun-EcallLumiSpec (e, ip6_extended) (pull_request)	Successful...		Details
✓	Build against eic-shell / npsim-dis (5x41, 1, brycecanon) (pull_request)	Successful in 51s		Details

- » Compile with gcc and clang
- » Static analysis and code style (clang-tidy, IWYU)
- » With AddressSanitizer and UBSanitizer
- » Run unit tests
- » Run simulation and reconstruction for gun and DIS (100 events)
- » Run JANA-based benchmarks
- » Upload artifacts (EDM4hep sim, EDM4eic reco, jana factory parameters, janadot, coverage report, doxygen)
- » Compare to reco EDM4eic to artifact from the base branch