

Luke's Software Blob

PittPACC Neutrino Generator Workshop
2024

TOC

- Container Image Contents
- Using Containers
- Using Containers For Development
- Using This Container
- NUISANCE3

Container Image Contents

The Container

Find the image here:

<https://hub.docker.com/repository/docker/picker24/pittpaccbox/general>

Software included:

Tools: ROOT (6.30), HepMC3 (3.3.0), LHAPDF (6.4.1), ProSelecta, NUISANCE2, NUISANCE3, Prob3++, Jupyterlab

Generators: NEUT (5.8.0), GENIE (3.04.02, ReWeight 1.04), NuWro (21.09.02), GiBUU (release2023, rootTuple, HepMC3 patch), Achilles (14eb544)

Experimental Frameworks: nusystematics (v2_00_02), T2KReWeight (24.05)

Inputs: GENIE Splines (AR23_20i_00_000, G18_10a_02_11a, G21_11a_00_000), BUUInput2023, NUISANCE2 data (including experimental fluxes).

Architectures: linux/amd64 and linux/arm64

Why Distributing via Containers?

- Ideal for software and dependency distribution:
 - I build the binaries so you don't have to.
- No need to build tagged versions of software:
 - No environment-specific issues for running software, runs the same on your laptop, my laptop, gpvms, lxplus, where ever...
- Can be used as a known build environment for developing software:
 - Since we know that the tagged versions built in the container, we should expect all the dependencies of the software exist in the container and development should be seamless.
 - I'll give an example of what I mean here, as this often confuses people.

Caveats with Containers

- They are ephemeral:
 - When you exit the main container process, the everything is thrown away.
 - Don't expect changes you make to the container image to persist in different containers from the same image. There are very natural ways to persist container artefacts, see later.
- Fiddly to set up X11-based UIs (TBrowser etc...)
 - I will recommend Jupyter, which works within the container
 - Use a version of ROOT installed on the host if you need a GUI (TBrowser, etc...)
- Some shared resources exhibit teething problems with using singularity to run container images. In my experience these are easily work-aroundable.
 - I will give an example of using this image on lxplus
 - I strongly recommend that you are working on your own laptop and not using shared resources this week unless it is really required.

Self-aware Workflow Commentary

- It is not lost on me that suggesting people change their workflow 'live' at a workshop is not likely to work for everyone. **WORK!**
- These tools are shared 'as is'. If you have your own system that you are confident with, feel free to continue using that.
- If you experience bugs/issues/confusion with these tools, please do flag them, I can try and fix some 'live' and other can be taken on board to improve these tools in the future.

Using Containers

What Is a Container?

- A way to 'abstract' away the host software distribution/OS from the dependencies of software you want to run
- A way to distribute software with its dependencies in a way that will run anywhere*
- A tarchive of a linux software distribution with a bit of JSON metadata specifying environment variables
- A low overhead way to isolate process execution (like a virtual machine)
- A simple way to seamlessly run linux software on windows/macOS
- <your definition here>

* anywhere that has a compatible container runtime... luckily they're standardised

Very Strong Recommendation

- For this week, I expect working on a local machine will be an order of magnitude more efficient than working on shared resources (gpvms, lxplus, university clusters).
 - You can throw 1M NuWro events in a minute or two on a single CPU core, we will during this talk!
- I will go through most of these examples assuming you are using docker desktop (or orbstack on Apple Silicon: M1, M2, M3).
 - If you are working on shared resources, you will be using apptainer/singularity, the syntax for starting the container is slightly different, but once you're in a container, everything should be the same
 - But good luck using Jupyter from shared resources...

How Do I Use a Container?

- If you are on linux, I would recommend using podman or docker, they are equivalent. podman lets you run containers without root access, docker requires root, both should be available from system package manager.
- If you are on macos, I would recommend [orbstack](#) or docker. orbstack is especially good on apple silicon (arm64v8) machines.
- You should think about it very similarly to using a software distribution system on shared resources:
 - `docker run -it --rm docker.io/picker24/pittpaccbox`
 - Should be thought of very similarly to:
 - `module load genie/3.04.00 neut/5.8.0 nuwro/21.09.02 nuisance/2 nuisance/3...`

How Do I Use a Container?

```
docker run -it --rm -v ${HOME}:/root -w /root docker.io/picker24/pittpacbox
```

How Do I Use a Container?

```
docker run -it --rm -v ${HOME}:/root -w /root docker.io/picker24/pittpacbox
```

docker run: start a new container, with options, from an image location

How Do I Use a Container?

```
docker run -it --rm -v ${HOME}:/root -w /root docker.io/picker24/pittpacbox
```

docker run: start a new container, with options, from an image location

-it --rm: attach the container to the current shell and when the entrypoint process exits, tear down the container

How Do I Use a Container?

```
docker run -it --rm -v ${HOME}:/root -w /root docker.io/picker24/pittpacbox
```

docker run: start a new container, with options, from an image location

-it --rm: attach the container to the current shell and when the entrypoint process exits, tear down the container

-v \${HOME}:/root -w /root: Mount your home directory on the host into the container as the root user's home directory. The user inside my container is root, but the runtime handles mapping uid/gids from the container to the host, so your permissions are translated into/out of the container as expected.

How Do I Use a Container?

```
docker run -it --rm -v ${HOME}:/root -w /root docker.io/picker24/pittpacbox
```

`docker run`: start a new container, with options, from an image location

`-it --rm`: attach the container to the current shell and when the entrypoint process exits, tear down the container

`-v ${HOME}:/root -w /root`: Mount your home directory on the host into the container as the root user's home directory. The user inside my container is root, but the runtime handles mapping uid/gids from the container to the host, so your permissions are translated into/out of the container as expected.

`docker.io/picker24/pittpacbox`: The image location. Will pull from dockerhub container registry if a copy doesn't exist locally, or it will use a copy from the local registry.

How Do I Use a Container? 1xp1us edition

```
export APPTAINER_CACHE_DIR=/tmp/${USER}/.apptainercache
apptainer pull docker://picker24/pittpaccbox:alma9
apptainer shell --cleanenv picker24_pittpaccbox.sif
```

How Do I Use a Container? 1xp1us edition

```
export APPTAINER_CACHE_DIR=/tmp/${USER}/.apptainercache
apptainer pull docker://picker24/pittpaccbox:alma9
apptainer shell --cleanenv picker24_pittpaccbox.sif
```

--cleanenv: apptainer by default will take bits of the host shell environment, which can cause weird interactions with the container, this option means that the container uses the environment shipped with the image.

docker_run

- aptainer has some nice defaults where it mounts your home directory inside the container and moves you to the same mounted working directory if possible. This makes it feel even more like just a 'software environment', rather than an 'isolated virtual machine'.
- I use a shell function `docker_run` which emulates this for docker/orbstack.
- Run and paste it into your `~/.bashrc` or `~/.zshrc` and restart your shell.
`docker run --rm docker.io/picker24/pittpacabox say_docker_run`
- I will use `docker_run` from here, which automatically mounts directories and passes `-it --rm`.

Using Containers For Development

Containers as Dev Environment

- In my opinion, containers are **best** used as a dev environment. I don't see many people using this workflow, but I find it amazingly efficient. So I'll bore you with it.
 - Bit-for-bit relocatable dev environments!
- The basic idea:
 - Mount somewhere from the host that is persistent
 - Clone the code that you want to develop
 - Edit it with your favorite editor on the host
 - build/run the code 'inside' the container.
 - Profit.
- The binary artefacts won't be executable outside of the container as they are built against the s/w distribution in the container, but that's not a problem. Spin up as many containers as you want. One for building one for running.

Rebuilding GENIE Example

**** it, we'll do it live...

Rebuilding GENIE Example

**** it, we'll do it live...

Rebuilding GENIE Example

**** it, we'll do it live...

```
$ docker_run picker24/pittpacabox
# cd ${HOME}; mkdir -p pittpacc_tut; cd pittpacc_tut
# git clone --depth 1 --branch R-3_04_02 \
    https://github.com/GENIE-MC/Generator.git GENIE-Generator
# cd GENIE-Generator
# export GENIE=$(pwd)
# ./configure --enable-lhapdf6 && make -j 10
# export PATH=${GENIE}/bin:${PATH}
# export LD_LIBRARY_PATH=${GENIE}/lib:${LD_LIBRARY_PATH}
```


Using This Container

How Do I Use This Container?

```
$ docker_run picker24/pittpaccbox halp
$ docker_run picker24/pittpaccbox halp neut
$ docker_run picker24/pittpaccbox halp nuwro
$ docker_run picker24/pittpaccbox halp genie
$ docker_run picker24/pittpaccbox halp gibuu
$ docker_run picker24/pittpaccbox halp achilles
$ docker_run picker24/pittpaccbox halp nusystematics
$ docker_run picker24/pittpaccbox halp t2kreweight
$ docker_run picker24/pittpaccbox halp nuisance2
$ docker_run picker24/pittpaccbox halp nuisance3
```

Generate Some NuWro Events

```
$ docker_run picker24/pittpaccbox  
# cd ${HOME}; mkdir -p pittpacc_tut; cd pittpacc_tut  
# nuwro -i $NUWRO/data/params.txt -p ${NUISANCE}/data/flux/
```

Generate Some Other Events

```
$ nuis flux help
```

NUISANCE2 Generation Helpers

NO WARRANTY AT ALL, THESE ARE PROVIDED AS IS. LEARN TO RUN THE GENERATOR DIRECTLY AND PLEASE DON'T COMPLAIN ABOUT THESE SCRIPTS TO THE GENERATOR AUTHORS!

```
$ nuis gen <generator> -E <experiment> -t <target> -P <probe> -n  
<numevents> -o <outputfile.root>
```

NUISANCE2 'Flat Trees'

```
$ nuisflat -i NuWro:eventsout.root -o nuwro.flat.root
```

Looking at nuwro.flat.root

Screenshot of one shell with container and one with root prompt.

A Word About Flux Histograms

GENIE expects bin content to be the neutrino rate, NEUT/NuWro
expect bin content to be rate density.

NUISANCE3

Why NUISANCE3?

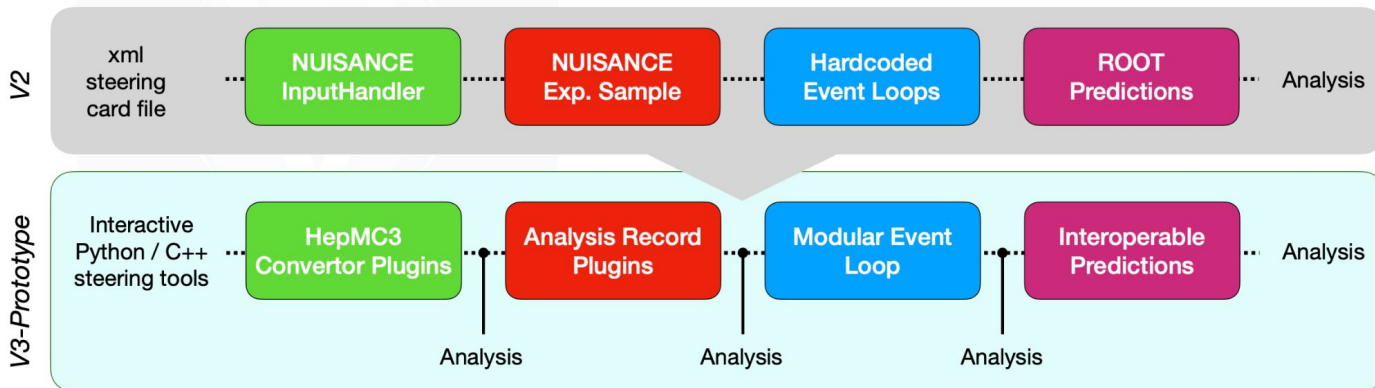
Patrick @ HSF

PROTOTYPING EFFORTS : nuisance3

- ✦ Exploring how the structure of NUISANCE could be rewritten to make it more accessible.
- ✦ Move away from prior monolith structure to modular interfaces.
- ✦ Enable external processing/tuning libraries to interface at different stages in the analysis chain through C++ or python.







Full C++ to Python Bindings



NUISANCE3

- NUISANCE is undergoing a ground-up rewrite
- All new 'tech':
 - HepMC3 (/NuHepMC)
 - HepData
 - ProSelecta
 - C++17
 - Natural python bindings
 - ... many more
- I don't necessarily expect you to use NUISANCE3 this week... but I'd be very happy if you did, and if you do, give feedback. We think it is ready for users, even if it doesn't yet have all of the NUISANCE2 features.
 - It has many more features than most users use from NUISANCE2. The rest of this talk will be a whistle-stop tour on it's features.

State of the v-tion: MCEG Format Edition

Generator Name	Vector Format	Comments
NEUT 	object TTree, 'flat' TTree	<ul style="list-style-type: none"> - Mostly F77, Pythia5, CERNLIB... - T2K/HK stacks built on NEUT 'flat' TTree - Event reweighting, flux/det. geom tools
Achilles 	HepMC3, NuHepMC	- C++17, interfaces with LHC MCEGs, e.g. SHERPA
GENIE 	object TTree, 'flat' TTree, XML	<ul style="list-style-type: none"> - FNAL v stacks built on GENIE object TTree - Event reweighting, flux/det. geom tools
NuWro	ROOT 'flat' tree	- Provides flux/geometry drivers
GiBUU	LHA XML, 'flat' TTree	- Sophisticated treatment of hadron transport
Marley 	NuHepMC	- Argon-target 'low-energy' MCEG

NuHepMC

- Much more info [here](#)
- Common event format based on HepMC3
 - No extra code **required**, can be read/written as 'pure' HepMC3.
 - Helper repo for adhering to extra NuHepMC requirements and recommendations, [here](#).
- Key features:
 - **Fully self describing vectors:** All interaction channels, FSI processes, models can be described and referenced in the vector itself.
 - **Native cross-section information:** multiple specifications for how to store flux-averaged cross section information needed to compare to cross section measurements. No longer an expert task!
- Implementation Status:
 - **Generators:** NEUT5: converter, NuWro: converter, GENIE: native (branch), Achilles: native, Marley: native, GiBUU: native (branch).
 - **Tools:** NUISANCE2: native input, NUISANCE3: native internal format, WCSim: native

HepData

- Goal is to offload the vast majority of the NUISANCE data store to an external repository.
- Real aim is **fully** automated MC–Data comparisons:
 - Foundational tech is separate from NUISANCE, available for use in your own comparison analyses.
- Need to be able to specify experimental parameters and measurement details
 - ProSelecta is a library based on cling and HepMC3 for interpreting and executing analysis steps from C++ snippets provided with data releases (stored on HepData).
- See an example [here](#)

Upload New Files

Download All

Filter 21 data tables

cross_section-pn

covariance-pn

smearing-pn

cross_section-alpha3d

covariance-alpha3d

smearing-alpha3d

cross_section-phi3d

covariance-phi3d

smearing-phi3d

cross_section-pn_para

covariance-pn_para

cross_section-pn

License: CC0

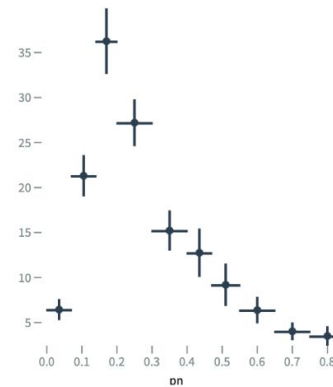
select	MicroBooNE_CC0Pi_GKI_nu_SelectSignal
project:pn	MicroBooNE_CC0Pi_GKI_nu_pn
target	Ar
probe_species	numu
variable_type	cross-section-measurement
covariance	covariance-pn
smearing	smearing-pn
pn	cross_section [cm² c/GeV /Nucleon]
0.0 - 0.07	6.4406 ±1.1679 total
0.07 - 0.14	21.314 ±2.2968 total
0.14 - 0.2	36.266 ±3.6505 total
0.2 - 0.3	27.206 ±2.6118 total
0.3 - 0.4	15.223 ±2.2399 total
0.4 - 0.47	12.758 ±2.6894 total

<https://www.hepdata.net/recc>



JSON

Visualize



Sum errors Log Scale (X) Log Scale (Y)

Deselect variables or hide different error bars by clicking on them.

Variables

Hep

- Go
- ext
- Re
 -
- Ne
- de
 -
- Se

analysis.cxx [10.17182/hepdata.1713917002.v1/r2](https://doi.org/10.17182/hepdata.1713917002.v1/r2)

License: CC0

Selection and projection function examples. Can be executed in the ProSelecta environment v1.0.

```
int MicroBooNE_CC0Pi_GKI_nu_SelectSignal(HepMC3::GenEvent const &ev) {
    static auto apply_mom_cut =
        [](std::vector<HepMC3::ConstGenParticlePtr> &parts, double mom_minimum,
           double mom_maximum) {
            parts.erase(std::remove_if(parts.begin(), parts.end(),
                                       [&](auto const &p) {
                                           double p3mod = p->momentum().p3mod();
                                           return (p3mod < mom_minimum) ||
                                                  (p3mod >= mom_maximum);
                                       }),
                       parts.end());
        };

    auto bpart = ps::sel::Beam(ev, 14);
    if (!bpart) {
        return 0;
    }
    auto cclep = ps::sel::PrimaryCCLepForNu(ev, bpart);
    if (!cclep) {
        return 0;
    }

    auto p_mu = cclep->momentum();
}
```

Download via DOI: `curl -OJLH "Accept: text/x-c++src" https://doi.org/10.17182/hepdata.1713917002.v1/r2`

Jupyter Time

NUISANCE3 Status

- Core interfaces are designed, but room for changes if users identify issues with them:
 - eventinput
 - eventframe/histframe
 - pyNUISANCE
 - ProSelecta/HepData (not really discussed here)