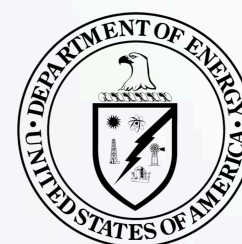


Datacards for ePIC

ePIC User Learning Workshop: Discoverability and Reusability for the pre-TDR and Beyond

Torri Jeske



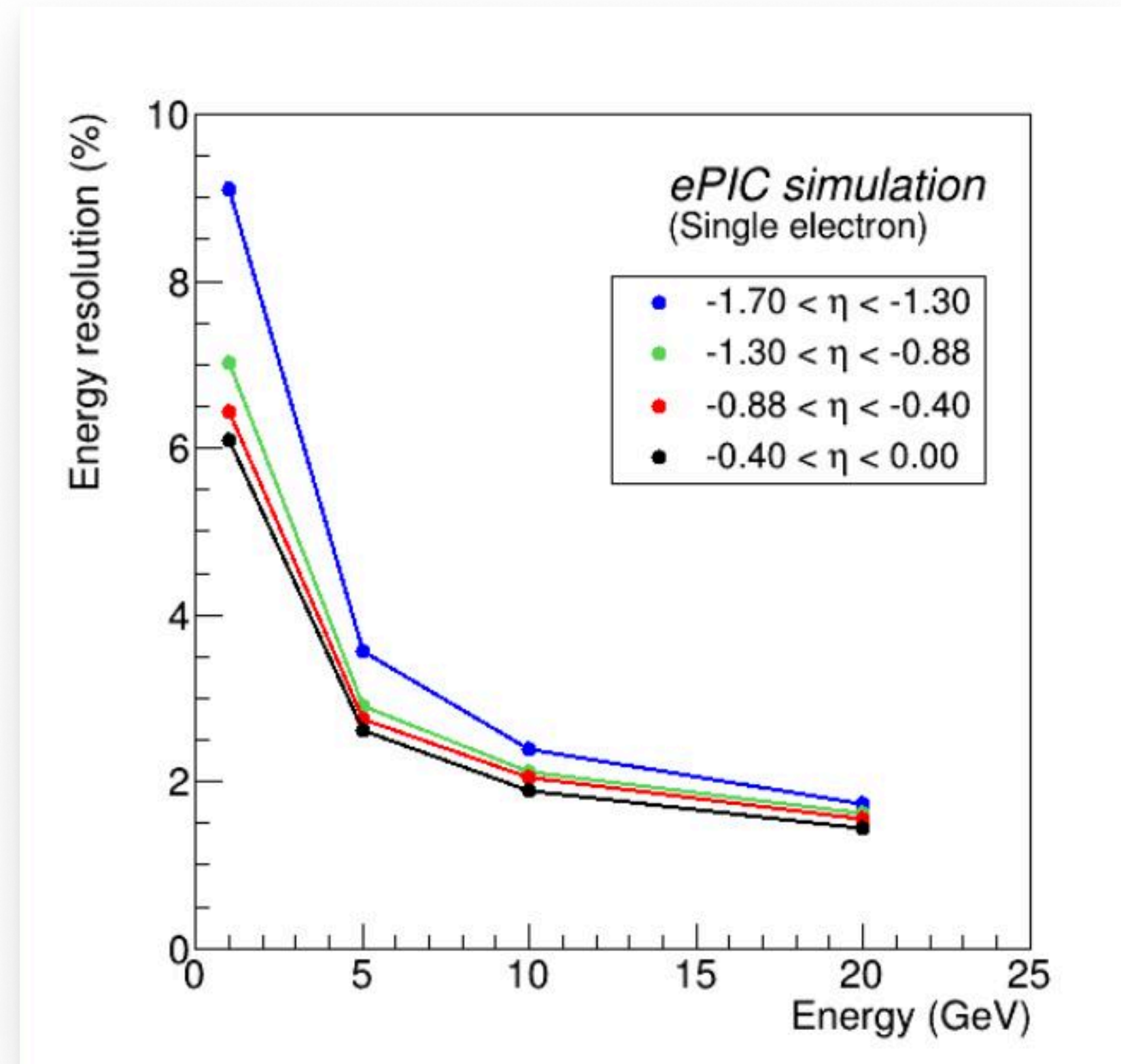
U.S. DEPARTMENT
of ENERGY

Reproducibility

TDR plots

ePIC currently generates image-based data used for QA analysis, pre-TDR studies, etc

Storing macros in a repository is a good start, but is not completely sufficient to reproduce a given plot



Reproducibility

BIC TDR Example

BIC Repository contains ROOT macros to generate plots incorporated into the TDR but:

- References directories on local user machine
- Doesn't indicate what campaign it is valid for
- Doesn't directly indicate which particular data set inside a given campaign is used
- Minho had to put up with about 8 different emails from me to regenerate 🗨️

The screenshot shows a Git repository interface for a user named 'mhkim-anl'. The repository name is 'Add materials for energy resolution plot'. The file tree on the left includes folders 'backup/data' and 'data', and files 'Eres_e-.png', 'Eres_gamma.png', 'README.md', 'getEres.C', and 'plotEres.C'. The main area displays the content of 'README.md', which is a ROOT macro. The macro defines a function 'crystalball' and a procedure 'getEres'. It sets parameters for a simulation, reads data from a file, and generates histograms for energy resolution. The code includes comments for parameters to be changed and instructions for running the macro.

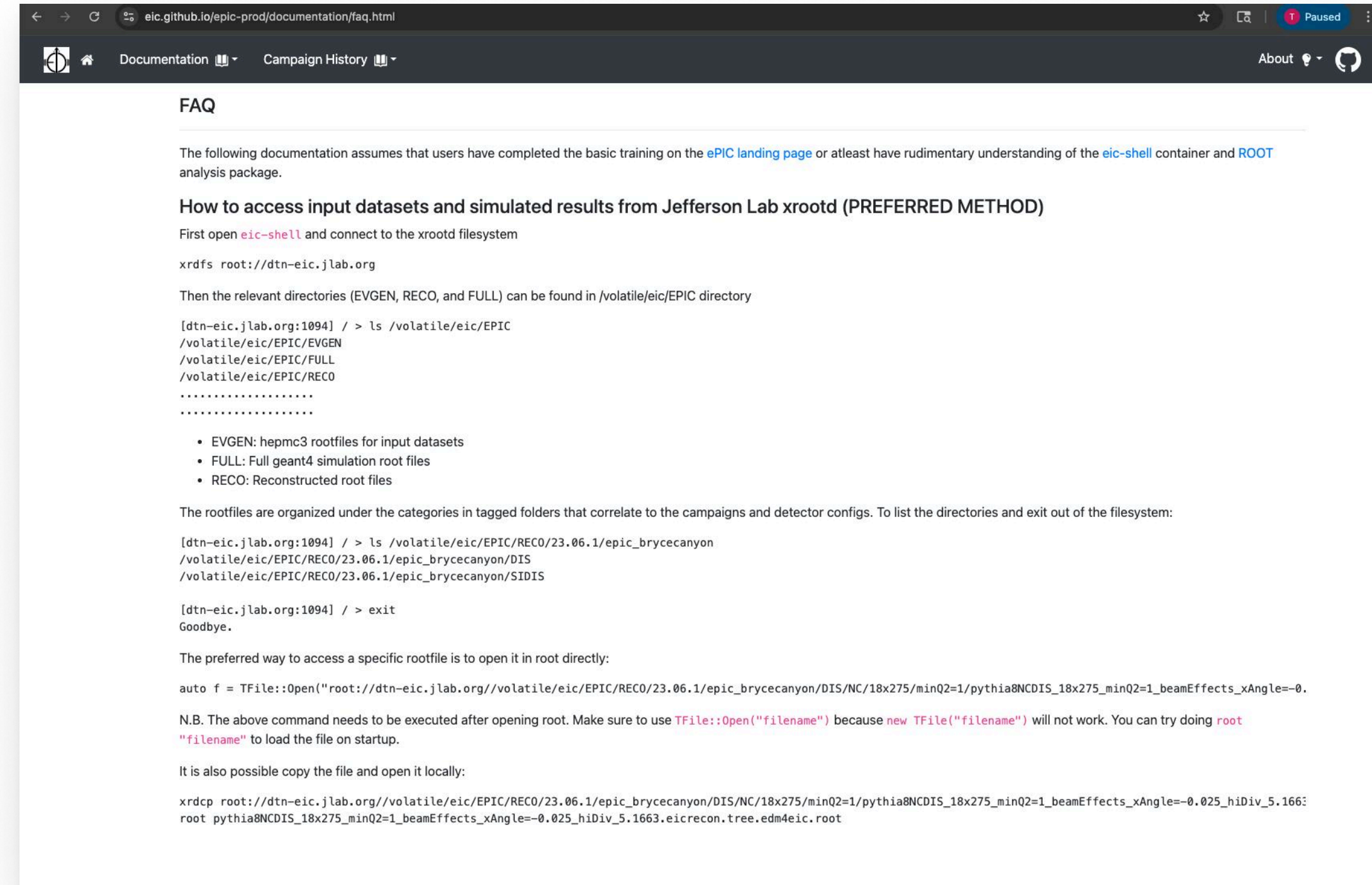
```
1 double crystalball(double *x, double* par){
2     return par[4]*ROOT::Math::crystalball_function(x[0], par[0], par[1], par[2], par[3]);
3 }
4
5 void getEres(){
6
7     // Parameters to be changed
8     //TString par = "gamma";
9     TString par = "e-";
10    const int mom0 = 20;
11    //TString ang = "45to135deg";
12    TString ang = "130to177deg";
13    TString path = "/home/minho.kim/eic/simulation/reco/data/e-/tmp";
14
15    // Input
16    TString infile = Form("%s/%s_%dGeV_%s.00*.eicrecon.tree.edm4eic.root",
17                        path.Data(), par.Data(), mom0, ang.Data());
18    TChain* ch = new TChain("events");
19    ch -> Add(infile);
20    TTreeReader reader(ch);
21
22    // Particle information
23    TTreeReaderArray<int> genStatus(reader, "MCParticles.generatorStatus");
24    TTreeReaderArray<float> mom_x(reader, "MCParticles.momentum.x");
25    TTreeReaderArray<float> mom_y(reader, "MCParticles.momentum.y");
26    TTreeReaderArray<float> mom_z(reader, "MCParticles.momentum.z");
27
28    // ScF1RecHits
29    TTreeReaderArray<float> scfi_Erec(reader, "EcalBarrelScF1RecHits.energy");
30
31    // Output
32    FILE* datRes;
33    if(ang=="130to177deg") datRes = fopen(Form("./data/%s/%dGeV/eta_-1.30_-0.88.dat", par.Data(), mom0), "w");
34    if(ang=="45to135deg") datRes = fopen(Form("./data/%s/%dGeV/eta_-0.88_-0.40.dat", par.Data(), mom0), "w");
35
36    // Eta bin and histograms
37    const int netab = 5;
38    const double etab[netab] = {-1.7, -1.3, -0.88, -0.4, 0};
39    TH1D* h1_Erec[netab-1];
40    double Emin = 0.6*mom0; double Emax = 1.2*mom0;
41    for(int i=0; i<netab-1; i++){
42        h1_Erec[i] = new TH1D(Form("h1_Erec_%d", i), "", 60, Emin, Emax);
43    }
44
45    // Histograms for check
46    TH1D* h1_eventType = new TH1D("h1_eventType", "", 10, 0, 10);
47
48    // Event loop
```

Reproducibility

Reproducing BIC plots

I have ROOT macro from GitHub, now:

- How do I access the simulation campaigns from JLab?
- Production Working Group maintains an up-to-date web page with a lot of useful information (if you know where to find it!)

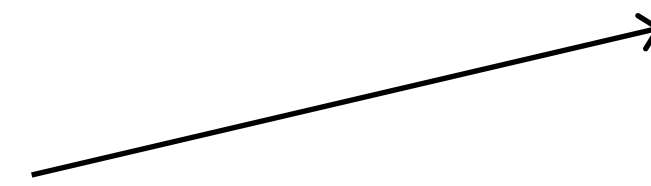


Reproducibility

Reproducing BIC plots

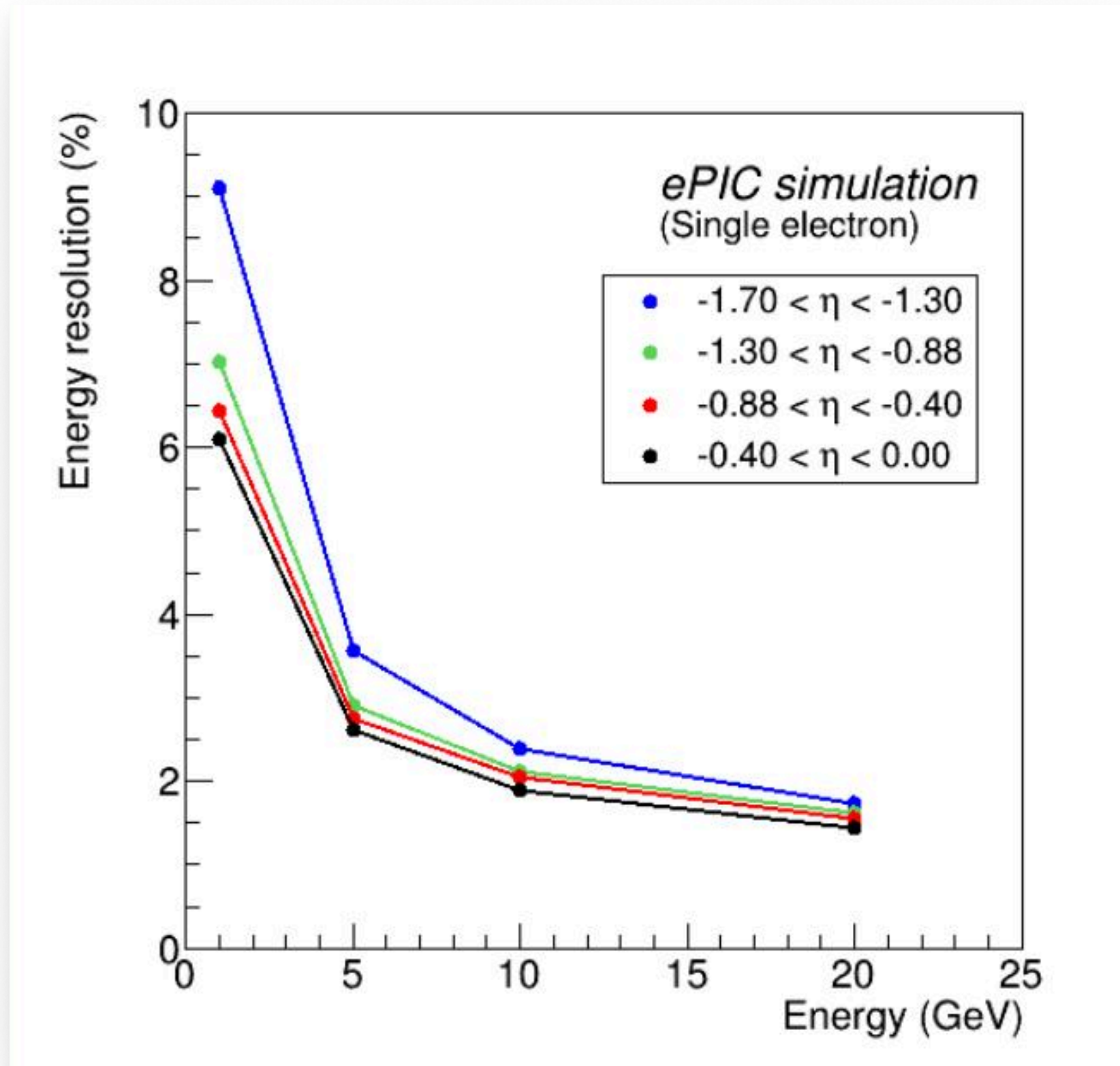
I have ROOT macro from GitHub, and the necessary campaign files now:

- What environment do I use (eic-shell)
- Where can I run this macro to produce the plots? (JLab, BNL, local machine)
- What I did: wrote a very basic script to submit jobs to JLab farm (clearly this doesn't scale)
- What we would like: Use a Snakemake workflow to be consistent with benchmarks

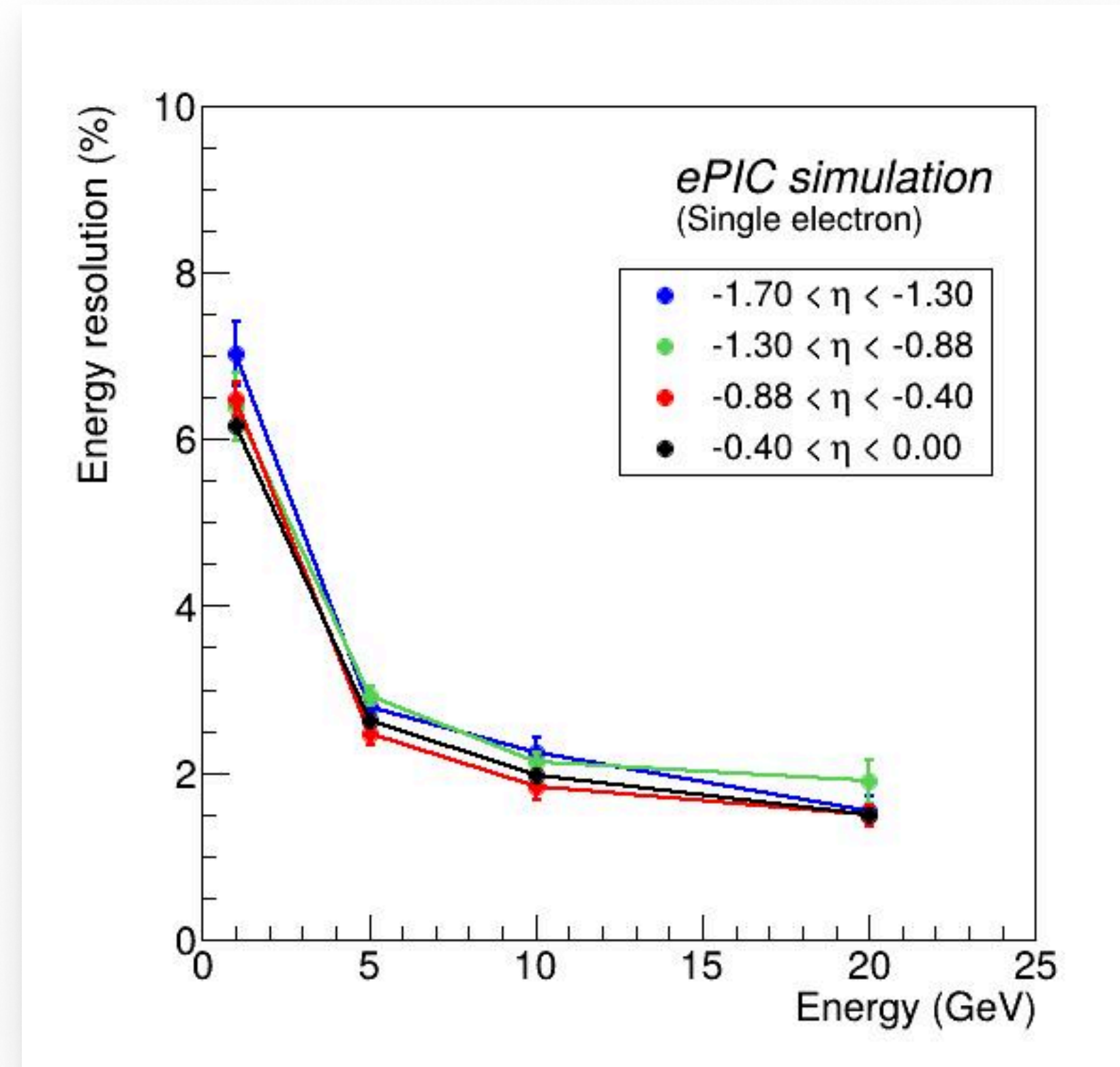


```
1  for CAMPAIGN in $CAMPAIGNS; do
2      # Validate explicitly provided campaigns.
3      if [ $# -gt 0 ]; then
4          echo "Verifying campaign $CAMPAIGN..."
5          if ! check_campaign_data "$CAMPAIGN"; then
6              echo "Skipping invalid campaign:
7              $CAMPAIGN"
8              echo ""
9              continue
10             fi
11         fi
12         echo "Submitting EnergyResolution for campaign
13         $CAMPAIGN..."
14         sbatch \
15             --job-name=er_${CAMPAIGN} \
16             --output=$LOGDIR/er_${CAMPAIGN}_%j.out \
17             --error=$LOGDIR/er_${CAMPAIGN}_%j.err \
18             --time=02:00:00 \
19             --partition=production \
20             --mem=8000 \
21             --gres=disk:1G \
22             --account=eic \
23             --
24         export=ALL,CAMPAIGN=$CAMPAIGN,DETECTOR=$DETECTOR \
25         --wrap="/volatile/eic/roark/epic-tdr-bic/
26         run_analysis.sh EnergyResolution $CAMPAIGN
27         $DETECTOR"
28         echo ""
29     done
```

Do they actually match?



From BIC TDR repo



What I generated

I haven't had time to figure out where I messed up but that is coming!

Caution!

This is absolutely a work in progress that Diana and I just started last week. Seriously. *Work in progress.*

Genesis mission datacards have been under development for much longer.

Datacards

“Let’s just do it” -- Torri’s famous last words



Data Cards

What's a datacard?

A structured metadata document that describes a dataset -- what it is, where it came from, who created it, how it can be accessed, and how it can be used.

Data cards serve both humans (who need to understand a dataset before using it) and machines (automated pipelines that ingest, catalog, and validate datasets).

There is no expectation that the ePIC community should already be familiar with datacards. This requires us to think about the data we produce in a way we most likely haven't before.

Data Cards

Datacard Profiles

Profile	When to use	Time to complete
core	In-workflow, draft, or simple datasets not yet being shared	10-15 minutes
extended	Datasets being shared with partners or submitted to OSTI, Zenodo, or similar	30-45 minutes
ai_ready	Datasets intended for AI/ML training, inference, or evaluation	45-60 minutes
sensitive	CUI, export-controlled, PII, or classified datasets	45-60 minutes

Data Cards

Field Annotations

Annotation	Meaning
core	Required for all profiles
extended	Required for extended, ai_ready, and sensitive profiles
ai_ready	Required for ai_ready profile only
sensitive	Required for the sensitive profile only
pub	Required for when release_status=approved or published
if_applicable	Fill in if it applies to your dataset; else skip
system	Do not fill in -- the repository system populates this at ingest

Datacards

Datacard Metadata

Describes the datacard itself, *NOT* the dataset.

Version controlled with patch, minor, and major. For each update, add corresponding entry to change_log

Profile determines the completeness level you are committing to for this datacard.

For right now, we are completing [core].

```
1 datacard:
2   template_version: "1.0"
3   datacard_version: "1.0"
4
5   profile: core
6   filename: "genesis_datacard_epic_simulation_campaign_march_2026.md"
7   language: en
8
9   id:
10    type: __TYPE__
11    value: __VALUE__
12
13   sensitivity_tier: tier0_open
14   access_level: open
15
16   creation_method: manual
17   created_date: "${2026-04-24}"
18   updated_date: "${YYYY-MM-DD}"
19
20   change_log:
21    - date: "${YYYY-MM-DD}"
22      datacard_version: "1.0"
23      summary: "Initial creation"
24
25   created_by:
26    - role: editor
27      date: "2026-04-24"
28      description: "Generated first version of datacard"
29      creator:
30        type: person
31        person:
32          given_name: Torri
33          family_name: Jeske
34          orcid: 0000-0002-5191-0501
35          email: roark@jlab.org
36          affiliation:
37            name: Thomas Jefferson National Accelerator Facility
38            ror_id: https://ror.org/02vwzrd76
39          organization:
```

Datacards

Level 1: Basic and Discoverable

Required for all datasets regardless of publication state or profile.

Define a single human-readable name for this dataset.

Define a project the data set belongs to, a version of the dataset

Identify a primary persistent identifier for this dataset → TO DO!

```
1  identification:
2    name: "ePIC simulation campaign root trees from March 2026"
3    project: "epic"
4    version: "1.0"
5
6    primary_id:
7      type: other
8      value: 26.04
9
10
11   additional_ids:
12     - type: __TYPE__
13       value: __VALUE__
14
15   supersedes:
16
17     type: __TYPE__
18     value: __VALUE__
19
20   superseded_by:
21
22     type: __TYPE__
23     value: __VALUE__
24
25   parent_collection:
26     name: __NAME__
27     identifier:
28       type: __TYPE__
29       value: __VALUE__
```

Datacards

Description

Defines human-readable, high-level description of the data set, collection methodology, intended use, current use, out of scope use, limitations, and keywords.

```
1 description:
2   summary: "This dataset contains reconstructed outputs in the format of ROOT
3   trees, produced in March of 2026. This dataset is intended for simulation studies
4   requested by the physics working groups (PWGs) and DSCs. Each data set contains
5   hepmp2 root files for input datasets, full geant4 simulation root files, and
6   reconstructed root files."
7   purpose: __PURPOSE__
8   collection_methodology: computational simulation
9
10  data_characteristics: __CHARACTERISTICS__
11
12  intended_use: simulation studies
13
14  current_use: __CURRENT_USE__
15
16  out_of_scope_use: __OUT_OF_SCOPE__
17
18  limitations: __LIMITATIONS__
19
20  keywords:
21    - simulation, ePIC, campaign
22
23  # --- Object & Dataset Type -----
24  object_type: dataset
25  dataset_type: ND
26
27  # --- Release Status -----
28  release_status: approved
29
30  # --- Workflow & Lifecycle -----
31
32  state: raw
33
34  is_intermediate: True
35
36  pipeline_stage: "post-reconstruction"
```

Datacards

Dataset Readiness

An assessment of the dataset's usability and interoperability level against the Genesis Dataset Readiness Model. Readiness describes usability, not scientific quality or value.

As of right now, ePIC simulation data is categorized as “discoverable”

```
1 dataset_readiness:
2   level: 1
3   evaluated_against: __MODEL__
4   evaluated_at: __YYYY-MM-DD__
5   evaluated_by:
6
7   type: __TYPE__
8   person:
9     given_name: __GIVEN_NAME__
10    family_name: __FAMILY_NAME__
11    orcid: __ORCID__
12    email: __EMAIL__
13    affiliation:
14      name: __ORG_NAME__
15      ror_id: __ROR_ID__
16    organization:
17      name: __ORG_NAME__
18      ror_id: __ROR_ID__
19    confidence: __CONF__
```

Datacards

Access Policy

Describes who can access the dataset and under what conditions.

Indicate the sensitivity tier for access control purposes, the access level required to obtain the dataset, and the access policy for the specific type of authorization required to access this dataset.

```
1 access_policy:  
2   sensitivity_tier: open  
3   access_level: open  
4   access_restrictions: None - publicly accessible  
5  
6   authorization_required: none  
7   policy_url: __URL__  
8   policy_text: __TEXT__
```

Datacards

Contacts

Used to establish the primary point of contact for questions about this dataset. Required for all profiles -- every dataset must have a reachable contact person or organization.

Defaulted to Production Working Group as an organizational contact

```
1 # --- Contacts -----
2 contact:
3   type: organization
4   person:
5     given_name: ${GIVEN_NAME}
6     family_name: ${FAMILY_NAME}
7     orcid: __ORCID__
8     email: ${EMAIL}
9     affiliation:
10      name: ${ORG_NAME}
11      ror_id: __ROR_ID__
12   organization:
13     name: Production Working Group
14     ror_id: __ROR_ID__
15     valid_until: __YYYY-MM-DD__
16     succession_note: __NOTE__
17 additional_contacts: []
```

Datacards

Authorship and Credits

Establish authorship via person or organization,
can provide list of contributors.

```
1 # --- Authorship & Credit -----
2
3 authors:
4   - type: ${TYPE}
5     person:
6       given_name: ${GIVEN_NAME}
7       family_name: ${FAMILY_NAME}
8       orcid: __ORCID__
9       email: __EMAIL__
10      affiliation:
11        name: ${ORG_NAME}
12        ror_id: __ROR_ID__
13      organization:
14        name: __ORG_NAME__
15        ror_id: __ROR_ID__
16      role: ${ROLE}
17 contributors: []
```

Datacards

Categorization

Indicate the high-level scientific domain or discipline associated with the dataset, structured tags for catalog filtering, the object type, and risk level.

In the future, for ai_ready datacards, indicate task category (e.g., classification) and relevant subcategories (e.g., binary classification, multi_label_classification)

```
1 # --- Categorization -----
2 categorization:
3   science_domain: "physics"
4
5   tags:
6     project: epic
7     science: physics
8     type: dataset
9     risk: general
10  task_category: []
11  task_subcategory: []
```

Datacards

Dataset Characteristics

Define a number of characteristics about your dataset.

```
1 # --- Dates
2 -----
3 ---
4
5 dates:
6   data_collection_start: __YYYY-MM-DD__
7   data_collection_end: __YYYY-MM-DD__
8   issued: __YYYY-MM-DD__
9   modified: __YYYY-MM-DD__
```

```
1 # --- Dataset Characteristics -----
2 dataset_info:
3   formats: ["ROOT"]
4   encoding: __ENCODING__
5   schema_version: __VERSION__
6
7   modalities: ["tree"]
8
9   features: []
10  splits: []
11  language: __LANG__
12  spatial_coverage:
13    description: __DESC__
14    bounding_box:
15      west: __DECIMAL_DEG__
16      east: __DECIMAL_DEG__
17      south: __DECIMAL_DEG__
18      north: __DECIMAL_DEG__
19  temporal_coverage:
20
21    start: __YYYY-MM-DD__
22    end: __YYYY-MM-DD__
23    description: __DESC__
24
25  dataset_scale:
26    record_count: __COUNT__
27    record_unit: __UNIT__
28    compressed_bytes: __BYTES__
29    uncompressed_bytes: __BYTES__
30
31  # --- Access Endpoints -----
32  ---
33  access:
34    current_location: "dtn-eic.jlab.org//
35    volatile/eic/EPIC/"
36    intended_repositories:
37      - name: __NAME__
38        access_level: __LEVEL__
39        is_primary: __BOOL__
40        date_deposited: __YYYY-MM-DD__
41    api:
42      endpoint: __URL__
43      documentation_url: __URL__
44      authentication: __AUTH__
45      version: __VERSION__
46      rate_limit: __LIMIT__
```

Datacards

Provenance

Describes how this dataset was created, what it was derived from, and what processing was applied.

Clear provenance sections in the datacards can significantly aid reproducibility

This section could capture a lot of the relevant information for Snakemake

```
1 # --- Provenance -----
2
3 provenance:
4   was_generated_by: "ePIC production and simulation campaign"
5   source_data:
6     - name: __NAME__
7       identifier:
8         type: __TYPE__
9         value: __VALUE__
10      relationship: __REL__
11   processing_steps: __DESCRIPTION__
12   instrumentation: __DESCRIPTION__
13   simulation_details: __DESC__
14   software_environment:
15     os: __OS__
16     compiler: __COMPILER__
17     container: __CONTAINER__
18     hpc_environment: __ENV__
19     notes: __NOTES__
```

Campaign-level datacard summary

Pick a profile

As a first attempt, I decided to go with [core]

Fill out datacards with [core] annotations

Remember the distinction between placeholder conventions (blank → information not yet known!)

Temporarily store datacards in eic/ datacards

This is a *private* repository for now while these cards are under active development

Datacards → Snakemake workflows

How can we use datacards and tools to generate Snakemake workflows that reproduce QA plots?

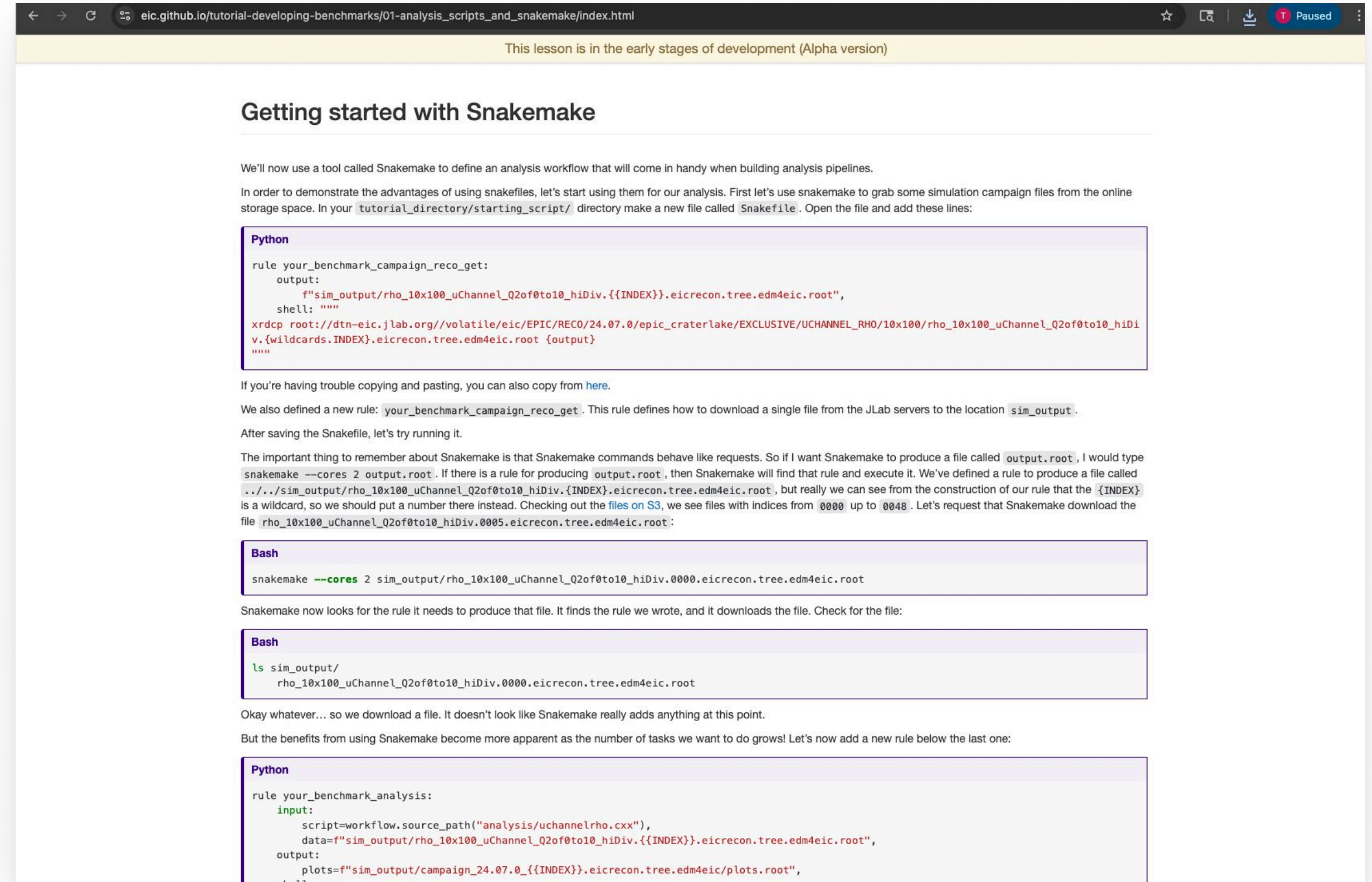
Important context: Most ePIC users are probably not familiar with Snakemake

Datacards → Snakemake

What is Snakemake?

Snakemake is an open-source workflow management system used to create reproducible and scalable data analysis pipelines. Users define “rules” that specify input files, output files, and the command(s) to generate them.

Dmitry Kalinkin led the development of Snakemake workflows for physics and detector benchmark plots.

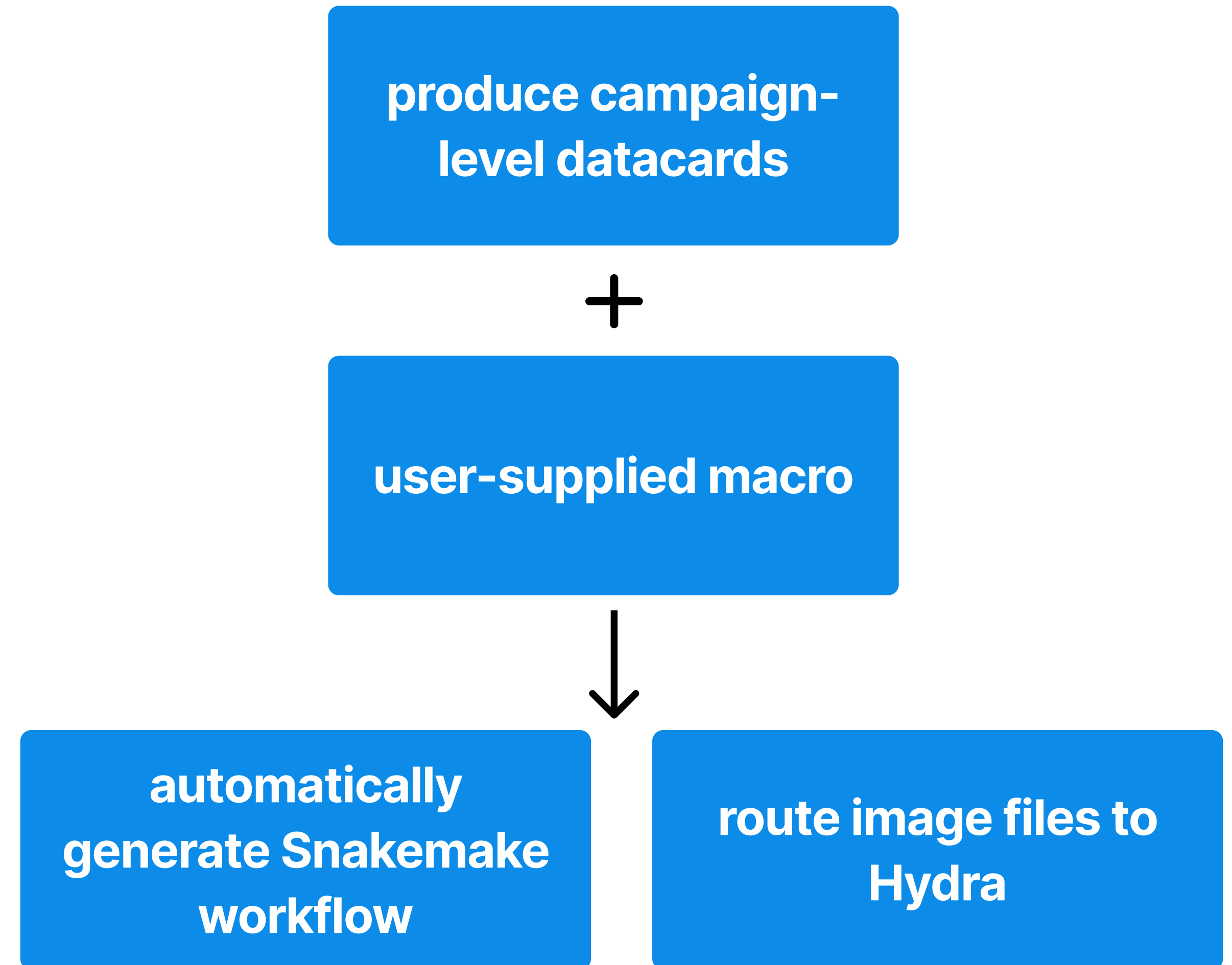


Datacards → Snakemake

Snakefile generation

With modifications, datacards could be used to automatically generate Snakefiles to aid in production of QA plots.

There is ongoing work in Genesis to develop agents with tools and skills that can interpret and generate workflows from datacards



Conclusions + Future Outlook + Action Items

Datacards

Structured metadata document that describes a dataset

Can be generated for campaign data as well as corresponding benchmark/QA plots

Datacard → Snakemake workflows

The ePIC community should have access to tools that can generate Snakemake workflows on their behalf using datacards

Input and collaboration

Datacards can help point out where ePIC production and analysis workflows are insufficient in capturing important aspects of the data, ePIC datasets and workflows can inform structure of datacards