The Journal of Open Source Software

The PHITS Tools Python package for parsing, organizing, and analyzing results from the PHITS radiation transport and DCHAIN activation codes

Hunter N. Ratliff [1]

1 Western Norway University of Applied Sciences, Inndalsveien 28, 5063 Bergen, Norway

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

License
Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License (CC BY 4.0).

Summary

Various areas within the nuclear sciences—such as nuclear facility design, medical physics, experimental nuclear physics, and radiation protection—rely on complex codes employing nuclear data and a large variety of physics models to simulate the transport (and interactions) of radiation through matter to answer important questions, in both research and applied contexts. For example: How should a shielding wall be designed to comply with radiation safety regulations? What dose will an individual receive in a particular exposure scenario? After how long will an irradiated radioactive sample decay enough to become safe to handle? How should an experiment be designed to make the most of limited time at an accelerator facility?

While simplified "rule of thumb" calculations can provide crude answers in some basic scenarios, fully modeling a radiation scenario is often necessary to obtain a much more precise answer. PHITS (Sato et al., 2024) (Particle and Heavy Ion Transport code System) is one such general purpose Monte Carlo particle transport simulation code, presently with over 7400 users[1]. Though PHITS can simulate a large variety of complex physics, it can only do so on the extremely short time scales of nuclear reactions. To calculate the creation and destruction of nuclides with time (activation, buildup, burnup, and decay) on any scale (seconds to centuries), distributed with and coupled to PHITS is the DCHAIN (Ratliff et al., 2020) code[2].

Radiation transport simulations minimally require specifying the involved geometry (defined shapes, regions, and materials), radiation source terms, and "tallies" that filter and score the various physical quantities of interest to be outputted. PHITS's tallies can score the number of particles passing through a region in space or crossing a surface, the frequency and products of nuclear interactions of various types, deposition of energy/dose, timing of interactions, radiation damage (in displacements per atom, DPA), and more. Users provide the desired binning for the tally histograms to be created (e.g., specifying a range of energies of interest and how many bins the tally will have in that energy range), and the code will simulate the histories, or "lives", of many particles, outputting the aggregate distributions for the quantities being tallied, which should be converged to the "true/real" distributions provided a statistically sufficient number of histories were simulated (often on the order of millions or more). For a few tallies, PHITS provides the option to output "dump" files where, in addition to the histograms, detailed raw history-by-history event data are recorded to a text or binary file for every history satisfying the tally's criteria and being scored by it, allowing users to, in post, create even more complex tallies and analyses than possible with the stock tallies in PHITS.

The DCHAIN code coupled to PHITS specializes in calculating nuclide inventories and derived quantities (such as activity, decay heat, decay gamma-ray emission spectra, and more) as a

[1] For current PHITS userbase statistics, see: https://phits.jaea.go.jp/usermap/PHITS_map_userbase.html

[2] PHITS and DCHAIN are distributed by the Japan Atomic Energy Agency and the OECD/NEA Data Bank. For more information, see: https://phits.jaea.go.jp/howtoget.html.

DRAFT

function of time for any arbitrary irradiation schedule from any radiation source.

The package presented here automates the time-consuming task of extracting the numerical results and metadata from PHITS/DCHAIN simulations and organizes them into a standardized format, easing and expediting further practical real-world analyses. It also provides functions for some of the most common analyses one may wish to perform on simulation outputs.

Statement of need

PHITS Tools and its DCHAIN Tools submodule serve as an interface between the plaintext (and binary) outputs of the PHITS and DCHAIN codes and Python—greatly expediting further programmatic analyses, comparisons, and visualization—and provide some extra analysis tools. The outputs of the PHITS code are, aside from the special binary "dump" files, plaintext files formatted for processing by a custom visualization code (generating Encapsulated PostScript files) shipped with and automatically ran by PHITS, and those of the DCHAIN code are formatted in a variety of tabular, human-readable structures.

Historically, programmatic extraction and organization of numerical results and metadata from both codes often required writing a bespoke processing script for most individual simulations, possibly preceded by manual data extraction/isolation too. PHITS Tools provides universal output parsers for the PHITS and DCHAIN codes, capable of processing all of the relevant output files produced by each code and outputting the numerical results and metadata in a consistent, standardized output format, also able to automatically make and save plots of tally results. No similar comprehensive PHITS/DCHAIN output parsing utilities presently exist. The MCPL: Monte Carlo Particle Lists (Kittelmann et al., 2017) package can parse PHITS binary dump files if using one of two specific combinations of tally dump parameter settings, and recent developments to FLUKA's (The FLUKA Collaboration et al., 2024) FLAIR utility (Donadon, André et al., 2024) involve ongoing integration efforts with PHITS.

The substantial number of combinations within PHITS of geometry specification, scoring axes (spatial, energy, time, angle, LET, etc.), tally types (scoring volumetric and surface crossing particle fluxes, energy deposition, nuclide production, interactions, radiation damage in DPA, and more), potential particle species, and fair amount of exceptions/edge cases related to specific tallies and/or their settings highlight the utility of such a universal processing code for PHITS. When parsing standard PHITS tally output, PHITS Tools returns a metadata dictionary, a 10-dimensional NumPy (Harris et al., 2020) array universally accommodating of all possible PHITS tally output containing all numerical results (structure illustrated in Table 1), and a Pandas (The pandas development team, 2020) DataFrame containing the same numerical information for users preferring working with Pandas.

Table 1: Structure of returned parsed tally output (NumPy array axes/Pandas DataFrame columns)

axis	description (using PHITS nomenclature, input syntax in `monospace` font)
0 / `ir`	Geometry mesh: `reg` / `x` / `r` / `tet` *
1 / `iy`	Geometry mesh: `1` / `y` / `1`
2 / `iz`	Geometry mesh: `1` / `z` / `z` *
3 / `ie`	Energy mesh: `eng` ([T-Deposit2] `eng1`)
4 / `it`	Time mesh
5 / `ia`	Angle mesh
6 / `il`	LET mesh
7 / `ip`	Particle type / group (`part =`)
8 / `ic`	Special: [T-Deposit2] `eng2`, [T-Yield] `mass/charge/chart`, [T-Interact] `act`
9 / `ierr`	`= 0/1/2`, Value / relative uncertainty / absolute uncertainty *

*exceptional behavior with [T-Cross] tally when `enclos = 0` is set; see full documentation

DRAFT

PHITS Tools is also capable of parsing the "dump" output files (both binary and plaintext formats) that are available for some tallies, and it can also automatically detect, parse, and process all PHITS output files listed in a provided directory or PHITS input file—very convenient for simulations employing multiple tallies, each with its own output file, whose output are to be further studied, e.g., compared to experimental data or other simulations. PHITS Tools can be used by:

1. importing it as a Python package in a script and calling its functions,
2. running it in the command line via its CLI with a provided PHITS output file (or directory/input file) path and settings flags/options, or
3. running it without any arguments or via the `PHITS-Tools-GUI` executable to launch a GUI and be stepped through the available output processing options and settings.

When used as an imported package, PHITS Tools provides a number of supplemental functions aiding with further analyses, such as tools for constructing one's own tally over the history-by-history output of the "dump" files, rebinning histogrammed results to a different desired binning structure, applying effective dose conversion coefficients from ICRP 116 (Petoussi-Henss et al., 2010) to tallied particle fluences, or retrieving a PHITS-input-formatted [Material] section entry (including its corresponding density) from a large database of over 350 materials (primarily consisting of the selection of materials within the PNNL Compendium of Material Composition Data for Radiation Transport Modeling (McConn et al., 2011)), among other useful functions.

The DCHAIN Tools submodule handles the outputs of the DCHAIN code. Its primary function parses all of the various output files of the DCHAIN code and compiles the metadata and numeric results—the confluence of the specified regions, output time steps, all nuclides and their inventories (and derived quantities), and complex decay chain schemes illustrating the production/destruction mechanisms for all nuclides—into a single unified dictionary object. The DCHAIN Tools submodule includes some additional useful functions such as retrieving neutron activation cross sections from DCHAIN's built-in nuclear data libraries, calculating flux-weighted single-group activation cross sections, and visualizing and summarizing the most significant nuclides (in terms of activity, decay heat, or gamma-ray dose) as a function of time. If PHITS Tools is provided DCHAIN-related files, DCHAIN Tools will be automatically imported and its primary function executed on the DCHAIN output.

In all, the PHITS Tools package makes the results produced by the PHITS and DCHAIN codes far more accessible for further use, analyses, comparisons, and visualizations in Python, removing the initial hurdle of parsing and organizing the raw output from these codes, and provides some additional tools for easing further analyses and drawing conclusions from the PHITS and DCHAIN results.

Acknowledgements

A portion of this work has been completed by the author while under the support of European Innovation Council (EIC) grant agreement number 101130979. The EIC receives support from the European Union's Horizon Europe research and innovation programme.

References

Donadon, André, Hugo, Gabrielle, Theis, Christian, & Vlachoudis, Vasilis. (2024). FLAIR3 – recasting simulation experiences with the advanced interface for FLUKA and other monte carlo codes. *EPJ Web Conf.*, *302*, 11005. https://doi.org/10.1051/epjconf/202430211005

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., … Oliphant,

DRAFT

T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Kittelmann, T., Klinkby, E., Knudsen, E. B., Willendrup, P., Cai, X. X., & Kanaki, K. (2017). Monte carlo particle lists: MCPL. *Computer Physics Communications*, *218*, 17–42. https://doi.org/https://doi.org/10.1016/j.cpc.2017.04.012

McConn, R. J., Gesh, C. J., Pagh, R. T., Rucker, R. A., & Williams, R., III. (2011). *Compendium of material composition data for radiation transport modeling*. Pacific Northwest National Lab. (PNNL), Richland, WA (United States). https://doi.org/10.2172/1023125

Petoussi-Henss, N., Bolch, W. E., Eckerman, K. F., Endo, A., Hertel, N., Hunt, J., Pelliccioni, M., Schlattl, H., & Zankl, M. (2010). Conversion coefficients for radiological protection quantities for external radiation exposures. *Annals of the ICRP*, *40*(2-5), 1–257. https://doi.org/10.1016/j.icrp.2011.10.001

Ratliff, H. N., Matsuda, N., Abe, S., Miura, T., Furuta, T., Iwamoto, Y., & Sato, T. (2020). Modernization of the DCHAIN-PHITS activation code with new features and updated data libraries. *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, *484*, 29–41. https://doi.org/10.1016/j.nimb.2020.10.005

Sato, T., Iwamoto, Y., Hashimoto, S., Ogawa, T., Furuta, T., Abe, S.-I., Kai, T., Matsuya, Y., Matsuda, N., Hirata, Y., Sekikawa, T., Yao, L., Tsai, P.-E., Ratliff, H. N., Iwase, H., Sakaki, Y., Sugihara, K., Shigyo, N., Sihver, L., & Niita, K. (2024). Recent improvements of the particle and heavy ion transport code system – PHITS version 3.33. *Journal of Nuclear Science and Technology*, *61*(1), 127–135. https://doi.org/10.1080/00223131.2023.2275736

The FLUKA Collaboration, Ballarini, Francesca, Batkov, Konstantin, Battistoni, Giuseppe, Bisogni, Maria Giuseppina, Böhlen, Till T., Campanella, Mauro, Carante, Mario P., Chen, Daiyuan, De Gregorio, Angelica, Degtiarenko, Pavel V., De la Torre Luque, Pedro, dos Santos Augusto, Ricardo, Engel, Ralph, Fassò, Alberto, Fedynitch, Anatoli, Ferrari, Alfredo, Ferrari, Anna, Franciosini, Gaia, … Zana, Lorenzo. (2024). The FLUKA code: Overview and new developments. *EPJ Nuclear Sci. Technol.*, *10*, 16. https://doi.org/10.1051/epjn/2024015

The pandas development team. (2020). *pandas-dev/pandas: Pandas* (latest). Zenodo. https://doi.org/10.5281/zenodo.3509134

DRAFT