

dchain_tools.py

Hunter Ratliff

Last updated: 29 July 2020

This document details the `dchain_tools.py` Python 3 library of functions. Note that this module is simply a collection of functions I have developed over time to speed up my usage of PHITS and DCHAIN-PHITS; these functions are not officially supported by the PHITS development team. They were developed to serve my own needs, and I am just publicly sharing them because others may also find utility in them. I may more professionally repackage and redistribute these functions in the future in a more standard way. If I write any other notably useful functions related to DCHAIN in the future, I will add them to this module.

While `dchain_tools.py` contains a variety of functions, its primary intended usage is with the `process_dchain_simulation_output()` function which parses all of the relevant DCHAIN output files for a simulation and returns a single Python dictionary object (which can also be accessed in an attribute/class style thanks to the `munch` library) containing all of the relevant output information in a much more accessible form.

1 Installation

The DCHAIN tools module is a single Python file called `dchain_tools.py` and is just a collection of functions. Place the `dchain_tools.py` file in a desired location and add the path to that folder to your `PYTHONPATH` system environmental variable if not already included in it. Then, the functions can be accessed just like with any other Python library by placing the following code at the top of any Python script:

```
1 from dchain_tools import *
```

And that's it! There are a number of libraries which importing `dchain_tools` will also automatically import:

```
1 import numpy as np
2 import os
3 import sys
4 import matplotlib.pyplot as plt
5 import time
6 import re
7 import bisect
8 import unicodedata as ud
9 from munch import *
```

These libraries will likely be installed already in any Python environment, especially if coming from a distribution including libraries such as Anaconda, unless it is brand new.

2 Available functions

All of these functions are reasonably well-documented within the `dchain_tools.py` script itself, so this document will not seek to rehash everything already written in the comment block at the start of each function. Instead, it serves to just highlight what functions are available. How the main `process_dchain_simulation_output()` function is used is covered in the following section.

2.1 Nuclide/element formatting functions

- `nuclide_plain_str_to_Dname` - convert a plaintext string for a nuclide (written in any of a large variety of sensible formats) to the format specifically used by DCHAIN in its output. **useful for searching for nuclides in output lists**
- `nuclide_plain_str_ZZZAAM` - convert a plaintext string for a nuclide to an integer ZZZAAM nuclide identifying number ($ZZZAAM = Z \cdot 10^4 + A \cdot 10 + M$)
- `nuclide_plain_str_to_latex_str` - convert a plaintext string for a nuclide to a LaTeX-formatted raw string useful for making plots and tables
- `Dname_to_ZAM` - converts a DCHAIN-formatted nuclide name to a ZZZAAM
- `ZAM_to_Dname` - converts a ZZZAAM to a DCHAIN-formatted nuclide name
- `Dname_to_Latex` - converts a DCHAIN-formatted nuclide name to LaTeX format
- `Element_Z_to_Sym` - returns elemental symbol provided the atomic number Z
- `Element_Sym_to_Z` - returns an atomic number Z provided the elemental symbol
- `Element_ZorSym_to_name` - returns a string of the full name of an element provided its atomic number Z or symbol
- `Element_ZorSym_to_mass` - returns the average atomic mass of an element provided its atomic number Z or symbol

2.2 Relating to DCHAIN data libraries

- `rxn_to_dchain_str` - converts a reaction to the format used by the neutron reaction cross section libraries in DCHAIN
- `ZZZAAM_to_dchain_xs_lib_str` - converts a ZZZAAM number to the 7-character nuclide string used by the neutron reaction cross section libraries
- `ECC01968_Ebins` - returns the n highest energy bins of ECCO 1968-group structure
- `retrieve_rxn_xs_from_lib` - returns the cross section for a specified reaction from a provided DCHAIN neutron reaction cross section library file
- `calc_one_group_nrxn_xs_dchain` - provided a neutron flux, reaction, and library file, determine the single-group neutron reaction cross section

2.3 DCHAIN output file parsing

- `process_dchain_simulation_output` - the main master function for parsing all output from DCHAIN, restructuring it, and placing it into a single dictionary object
- `parse_DCHAIN_act_file` - parser for the *.act file from DCHAIN
- `generate_nuclide_time_profiles` - processes output from `parse_DCHAIN_act_file` to turn raw nuclide output into a more usable / more easily referenced format
- `parse_DCHAIN_act_file_legacy` - legacy version of `parse_DCHAIN_act_file` from before statistical uncertainty propagation implementation
- `generate_nuclide_time_profiles_legacy` - legacy version from before statistical uncertainty propagation implementation of `generate_nuclide_time_profiles`
- `parse_DCS_file_from_DCHAIN` - parser for the *.dcs file from DCHAIN
- `parse_dtrk_file` - parser for the *.dtrk file from PHITS meant for DCHAIN
- `parse_dyld_files` - parser for the *.dyld files from PHITS meant for DCHAIN
- `plot_top10_nuclides` - generates a nice visualization on the ranking of nuclides across a desired axis (activity, decay heat, etc.) over time or over regions

3 Primary function usage

As mentioned earlier, `process_dchain_simulation_output()` is the primary function in this library; most of the other functions serve to just be called by this one (but still work fine standalone). It returns one output, referred to just as `dchain_output`, which will be discussed in detail later. The five inputs (2 mandatory, 3 optional) are detailed below.

Inputs for `process_dchain_simulation_output()`

<code>simulation_folder_path</code>	text string of path to folder containing simulation output (if not an empty string, it should end with / or \)
<code>simulation_basename</code>	common string of the DCHAIN simulations; output files are named <code>simulation_basename.*</code> . This string is also what is entered in the <code>file</code> variable in the [T-Dchain] tally, without any extension included.
<code>dtrk_filepath</code>	(<i>optional</i>) file path to *.dtrk file, only necessary if it has a different basename and there are multiple *.dtrk files in the simulation folder
<code>dyld_filepath</code>	(<i>optional</i>) file path to *.dyld file(s), only necessary if it has a different basename and there are multiple *.dyld files in the simulation folder
<code>process_DCS_file</code>	(<i>optional</i>) Boolean variable specifying whether the DCS file should be processed too. As this file can sometimes be quite large, for performance reasons its processing is disabled by default. (default=False)

The `process_dchain_simulation_output()` function primarily serves to parse the `*.act` main output file of DCHAIN but can also read the `*.dcs` file and will automatically attempt to read the PHITS-generated `*.dtrk` and `*.dyld` files. (The PHITS-generated files are mostly useful for diagnostic and secondary purposes.)

To make this usage more clear, a brief example will now be covered. In this example, the initial PHITS simulation input file was located in a folder whose path is `C://path/to/simulation/folder/`, and the [T-Dchain] tally had `file = example.in` set as the name of the DCHAIN input file to be automatically generated. After completing the PHITS simulation (resulting in `example.in`, `example.dyld`, `example.dtrk`, and `dch_link.dat` being written), the DCHAIN input file was ran through DCHAIN without moving or renaming any files (resulting in a number of files named `example.*` being produced). After this, the below Python code is used to extract the results from the output files.

```

1 from dchain_tools import *
2
3 folder_path = r'C:\path\to\simulation\folder\'
4 simulation_name = 'example' # note: no extension included
5
6 dchain_output = process_dchain_simulation_output(folder_path,
7         simulation_name, process_DCS_file=True)
8
9 # print list of all nuclides (formatted as plain text strings) found in
10 # the first region across all times
11 print(dchain_output['nuclides']['names'][0]) # dictionary-style access
12
13 # print the total activity and its absolute error in the first region
14 # and first time step
15 print(dchain_output.nuclides.total.activity.value[0][0], dchain_output.
16         nuclides.total.activity.error[0][0]) # attribute-style access
17
18 # find the activity of Na22 in the 2nd region at the 5th time step
19 ri = 1 # region index
20 ti = 4 # time index
21 # get desired nuclide name formatted in DCHAIN syntax
22 Na22_Dname = nuclide_plain_str_to_Dname('Na-22')
23 # determine index of desired nuclide among all nuclides in this region
24 Na22_index = dchain_output['nuclides']['names'][ri].index(Na22_Dname)
25 # now extract the activity value
26 A_Na22 = dchain_output.nuclides.activity.value[ri][ti, Na22_index]
27 print('A(Na-22) in [Bq/cc] in region 2 at 5th output time: ', A_Na22)

```

The `dchain_output` variable is a Python dictionary containing all of the relevant values outputted by DCHAIN. It has been “munchified” by the munch library too, meaning it can be accessed both as a normal Python dictionary or in the attribute style associated with classes. Generally, all entries are either single values, lists of values, or lists of NumPy arrays. For most outputs, this is of the form of a list containing elements for each region where each element is either a single string/value or a NumPy array/list further dividing results by time, nuclide species, etc. This dictionary is structured as outlined below. While the parts relevant to the `*.act` file are always written, the sections relevant to the other files are only written if the files are found / their parsing is requested (in the case of the `*.dcs` file). A more convenient version of this information for general reference or printing can be found in the accompanying `dchain_tools_output_structure.pdf` file.

```

1
2 # Notation for output array dimensions
3 #   R   regions
4 #   T   time steps
5 #   N   max number of nuclides found in a single region
6 #   E   number of gamma energy bins
7 # 1e10 for top 10 lists, a number <= 10
8
9
10 dchain_output = {
11   'time':{                                # ~ Time information
12     'from_start_sec'                      # [T] list of times from start time [sec]
13     'from_EOB_sec'                       # [T] list of times from end of final bombardment [sec]
14     'of_EOB_sec'                         # scalar time marking end of final bombardment [sec]
15   }
16
17   'region':{                              # ~ Information which only varies with region
18     'numbers'                            # [R] region numbers
19     'number'                            # [R] region numbers
20     'irradiation_time_sec'              # [R] irradiation time per region
21     'volume'                            # [R] volume in [cc] per region
22     'neutron_flux'                      # [R] neutron flux in [n/cm^2/s] per region
23     'beam_power_MW'                     # [R] beam power in [MW] per region
24     'beam_energy_GeV'                   # [R] beam energy in [GeV] per region
25     'beam_current_mA'                   # [R] beam current in [mA] per region
26   }
27
28   'nuclides':{                           # ~ Main nuclide results from *.act file
29     'names'                             # [R][N] names of nuclides produced in each region
30     'TeX_names'                         # [R][N] LaTeX-formatted names of nuclides produced
31     'ZZZAAAM'                          # [R][N] ZZZAAAM values (=10000*Z+10*A+M) of nuclides
32                                         # (ground state m=0, metastable m=1,2,etc.)
33     'half_life'                         # [R][N] half lives of nuclides produced [sec]
34     'inventory':{'value'                # [R][T,N] atoms [#/cc]
35                   'error'}              # [R][T,N] atoms [#/cc]
36     'activity':{'value'                 # [R][T,N] activity [Bq/cc]
37                  'error'}              # [R][T,N] activity [Bq/cc]
38     'dose_rate':{'value'                # [R][T,N] dose-rate [uSv/h*m^2]
39                   'error'}             # [R][T,N] dose-rate [uSv/h*m^2]
40     'decay_heat':{
41       'total':{'value'                  # [R][T,N] total decay heat [W/cc]
42                 'error'}               # [R][T,N] total decay heat [W/cc]
43       'beta':{'value'                   # [R][T,N] beta decay heat [W/cc]
44                'error'}                # [R][T,N] beta decay heat [W/cc]
45       'gamma':{'value'                  # [R][T,N] gamma decay heat [W/cc]
46                 'error'}              # [R][T,N] gamma decay heat [W/cc]
47       'alpha':{'value'                  # [R][T,N] alpha decay heat [W/cc]
48                 'error'}              # [R][T,N] alpha decay heat [W/cc]
49     }
50     'column_headers'                   # Length 7 list of the *.act columns' descriptions
51     'total':{                          # ~ Total values summed over all nuclides
52       'activity':{'value'               # [R][T] total activity [Bq/cc]
53                    'error'}            # [R][T] total activity [Bq/cc]
54       'decay_heat':{'value'             # [R][T] total decay heat [W/cc]
55                      'error'}          # [R][T] total decay heat [W/cc]
56       'beta_heat':{'value'              # [R][T] total beta decay heat [W/cc]
57                     'error'}           # [R][T] total beta decay heat [W/cc]
58       'gamma_heat':{'value'             # [R][T] total gamma decay heat [W/cc]
59                      'error'}          # [R][T] total gamma decay heat [W/cc]
60       'alpha_heat':{'value'             # [R][T] total alpha decay heat [W/cc]
61                      'error'}          # [R][T] total alpha decay heat [W/cc]
62       'activated_atoms':{'value'        # [R][T] total activated atoms [#/cc]
63                           'error'}     # [R][T] total activated atoms [#/cc]
64       'gamma_dose_rate':{'value'        # [R][T] total gamma dose rate [uSV/h*m^2]
65                           'error'}     # [R][T] total gamma dose rate [uSV/h*m^2]
66     }
67   }
68
69
70   'gamma':{                             # ~ Gamma spectra and totals
71     'spectra':{
72       'group_number'                   # [R][T,E] group number

```

```

73         'E_lower'                # [R][T,E] bin energy lower-bound [MeV]
74         'E_upper'                # [R][T,E] bin energy upper-bound [MeV]
75         'flux': {'value'          # [R][T,E] flux [#s/cc]
76                 'error'}         # [R][T,E] flux [#s/cc]
77         'energy_flux': {'value'   # [R][T,E] energy flux [MeV/s/cc]
78                          'error'} # [R][T,E] energy flux [MeV/s/cc]
79     }
80     'total_flux': {'value'         # [R][T] total gamma flux [#s/cc]
81                   'error'}        # [R][T] total gamma flux [#s/cc]
82     'total_energy_flux': {'value'  # [R][T] total gamma energy flux [MeV/s/cc]
83                           'error'} # [R][T] total gamma energy flux [MeV/s/cc]
84     'annihilation_flux': {'value'  # [R][T] annihilation gamma flux [#s/cc]
85                           'error'} # [R][T] annihilation gamma flux [#s/cc]
86     'current_underflow': {'value'  # [R][T] gamma current underflow [#s]
87                           'error'} # no error reported
88     'current_overflow': {'value'   # [R][T] gamma current overflow [#s]
89                          'error'}  # no error reported
90 }
91
92 'top10': {                          # ~ Top 10 lists from *.act file
93     'activity': {
94         'rank'                    # [R][T,le10] rank
95         'nuclide'                # [R][T,le10] nuclide name
96         'value'                  # [R][T,le10] activity [Bq/cc]
97         'error'                  # [R][T,le10] activity [Bq/cc]
98         'percent'                # [R][T,le10] percent of total activity
99     }
100    'decay_heat': {
101        'rank'                    # [R][T,le10] rank
102        'nuclide'                # [R][T,le10] nuclide name
103        'value'                  # [R][T,le10] decay heat [W/cc]
104        'error'                  # [R][T,le10] decay heat [W/cc]
105        'percent'                # [R][T,le10] percent of total decay heat
106    }
107    'gamma_dose': {
108        'rank':                  # [R][T,le10] rank
109        'nuclide'                # [R][T,le10] nuclide name
110        'value'                  # [R][T,le10] dose-rate [uSv/h*m^2]
111        'error'                  # [R][T,le10] dose-rate [uSv/h*m^2]
112        'percent'                # [R][T,le10] percent of total gamma dose rate
113    }
114 }
115 'number_of': {                    # ~ Maximum values of R, T, N, and E
116     'regions'                   # R = total number of regions
117     'time_steps'                # T = total number of time steps
118     'max_nuclides_in_any_region' # N = maximum unique nuclides found in any region
119     'gamma_energy_bins'         # E = number of gamma energy bins (default=42)
120 }
121 }
122
123 if process_dtrk_file: dchain_output.update({
124     'neutron': {                  # ~ Neutron spectra and totals
125         'spectra': {              # - Actual values used in DCHAIN
126             'E_lower'            # [R][E] bin energy lower-bound [MeV]
127             'E_upper'            # [R][E] bin energy upper-bound [MeV]
128             'flux': {'value'     # [R][E] neutron flux [#s/cm^2]
129                       'error'}   # [R][E] neutron flux [#s/cm^2]
130         }
131         'total_flux': {'value'   # [R] total neutron flux [#s/cm^2]
132                        'error'}  # [R] total neutron flux [#s/cm^2]
133         'unit_spectra': {        # - Flux per unit source particle (raw *.dtrk output)
134             'E_lower'            # [R][E] bin energy lower-bound [MeV]
135             'E_upper'            # [R][E] bin energy upper-bound [MeV]
136             'flux': {'value'     # [R][E] neutron flux [#s/cm^2/s.p.]
137                       'error'}   # [R][E] neutron flux [#s/cm^2/s.p.]
138         }
139     })
140
141 if process_dyld_files:
142     dchain_output.update({
143         'yields': {              # ~ Yield spectra
144             'all_names'          # [N] names of all nuclides produced
145             'names'              # [R][N] names of nuclides produced in each region
146             'TeX_names'          # [R][N] LaTeX-formatted names of nuclides produced

```

```

146         'ZZZAAAM'                # [R][N] ZZZAAAM values (=10000Z+10A+M) of nuclides
147                                   # (ground state m=0, metastable m=1,2,etc.)
148         'rate':{                  # - Actual values used in DCHAIN (at 100% beam power)
149             'value'               # [R][E] nuclide yield rate [#s/cm^3]
150             'error'               # [R][E] nuclide yield rate [#s/cm^3]
151         }
152         'unit_rate':{             # - Yields per unit source particle
153             'value'               # [R][E] nuclide yield rate [#s.p.]
154             'error'               # [R][E] nuclide yield rate [#s.p.]
155     }}})
156
157     if process_DCS_file: # add extra information
158         # Notation for output array dimensions
159         #   R (n_reg) regions
160         #   Td (ntsteps) time steps in DCS file (usually different from that of *.act file!)
161         #   Nd (nnuc_max) max number of nuclides (this index differs from the *.act N index)
162         #   C (chni_max) maximum index of relevant chains
163         #   L (chln_max) maximum number of links per chain
164
165     dchain_output.update({
166     'DCS':{
167         'time':{
168             'from_start_sec'       # [Td] list of times from start time [sec]
169             'from_EOB_sec'         # [Td] list of times from end of final bombardment [sec]
170             'of_EOB_sec'           # scalar time marking end of final bombardment [sec]
171         }
172
173         'number_of':{
174             'regions'              # ~ Maximum values of R, Td, Nd, C, and L
175             'time_steps'           # R = total number of regions
176             'max_nuclides'         # Td = total number of time steps
177             'max_number_of_chains' # Nd = max number of end nuclides in any time step
178             'max_chain_length'     # C = highest index of a relevant chain found
179             'max_chain_length'     # L = max number of links (nuclides) in any chain
180         }
181
182         'end_nuclide':{
183             'names'                # ~ Informtaion on nuclides at the end of each chain
184             'inventory':{          # [R][Td,Nd] nuclide names
185                 'N_previous'       # [R][Td,Nd,C] inventory in previous time step [atoms/cc]
186                 'N_now'            # [R][Td,Nd,C] inventory in current time step [atoms/cc]
187                 'dN'               # [R][Td,Nd,C] change in inventory of end nuclide from
188                                     # previous to current time step [atoms/cc]
189             }
190             'activity':{
191                 'A_previous'        # [R][Td,Nd,C] activity in previous time step [Bq/cc]
192                 'A_now'             # [R][Td,Nd,C] activity in the current time step [Bq/cc]
193                 'dA'                # [R][Td,Nd,C] change in activity of end nuclide from
194                                     # previous to current time step [Bq/cc]
195             }
196         }
197
198         'chains':{
199             'indices_of_printed_chains' # ~ Chains, individual links, and their contributions
200             'length'                    # [R][Td,Nd] the chain indices which were printed
201             'link_nuclides'             # [R][Td,Nd,C] length of listed chain
202             'link_decay_modes'          # [R][Td,Nd,C,L] strings of the nuclides in each chain
203             'link_dN':{                 # [R][Td,Nd,C,L] strings of the decay modes each link
204                 'beam'                  # undergoes to produce the next link
205                 'decay_nrxn'            # (only generated if values in file, 'None' otherwise)
206                 'total'                 # [R][Td,Nd,C,L] beam contribution to dN from each link
207             }
208         }
209
210         'relevant_nuclides':{
211             'names'                    # ~ A vs t profiles of nuclides over relevancy threshold
212             'times'                    # [R] list of relevant nuclides per region
213             'inventory'                 # [R][Td,Nd] time [s]
214             'activity'                  # [R][Td,Nd] inventory [atm/cc]
215             'activity'                  # [R][Td,Nd] activity [Bq/cc]
216         }
217     }}})

```

dchain_tools_output_structure