ORNL/TM-2015/409

ORIGAMI Automator Primer: Automated ORIGEN Source Terms and Spent Fuel Storage Pool Analysis

William A. Wieselquist
Adam B. Thompson
Joshua L. Peterson
Stephen M. Bowman

April 2016

**Approved for public release.
Distribution is unlimited.**

OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE US DEPARTMENT OF ENERGY

ORNL/TM-2015/409

Reactor and Nuclear Systems Division

ORIGAMI AUTOMATOR PRIMER: AUTOMATED ORIGEN SOURCE TERMS AND SPENT FUEL STORAGE POOL ANALYSIS

William A. Wieselquist
Adam B. Thompson
Joshua L. Peterson
Stephen M. Bowman

Date Published: April 2016

Prepared by
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, TN 37831-6283
managed by
UT-BATTELLE, LLC
for the
US DEPARTMENT OF ENERGY
under contract DE-AC05-00OR22725

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

BWR	boiling water reactor
CSV	comma-separated value
DOE	US Department of Energy
EFPD	effective full-power day
GUI	graphical user interface
INMM	Institute of Nuclear Materials Management
JSON	JavaScript Object Notation
MelMACCS	MELCOR to MACCS2 interface
MOX	mixed oxide
NRC	Nuclear Regulatory Commission
OPUS	ORIGEN Post-processing Utility for SCALE
ORIGAMI	ORIGEN Assembly Isotopics
ORIGEN	Oak Ridge Isotope Generation and Depletion Code
ORNL	Oak Ridge National Laboratory
PWR	pressurized water reactor
SCALE	Standardized Computer Analyses for Licensing Evaluation
SFP	spent fuel pool
SNF	spent nuclear fuel

1. INTRODUCTION

Source terms and spent nuclear fuel (SNF) storage pool decay heat load analyses for operating nuclear power plants require a large number of Oak Ridge Isotope Generation and Depletion (ORIGEN) calculations. SNF source term calculations also require a significant amount of bookkeeping to track quantities such as core and assembly operating histories, spent fuel pool (SFP) residence times, heavy metal masses, and enrichments. The ORIGEN Assembly Isotopics (ORIGAMI) module in the SCALE code system [1] provides a simple scheme for entering these data. However, given the large scope of the analysis, extensive scripting is necessary to convert formats and process data to create thousands of ORIGAMI input files (one per assembly) and to process the results into formats readily useable by follow-on analysis tools.

This primer describes a project within the SCALE Fulcrum graphical user interface (GUI) called ORIGAMI Automator that was developed to automate the scripting and bookkeeping in large-scale source term analyses. The ORIGAMI Automator enables the analyst to (1) easily create, view, and edit the reactor site and assembly information, (2) automatically create and run ORIGAMI inputs, and (3) analyze the results from ORIGAMI. ORIGAMI Automator uses the standard ORIGEN binary concentrations files produced by ORIGAMI, with concentrations available at all time points in each assembly's life. The GUI plots results such as mass, concentration, activity, and decay heat using a powerful new ORIGEN Post-Processing Utility for SCALE (OPUS) GUI component. This document includes a description and user guide for the GUI, a step-by-step tutorial for a simplified scenario, and appendices that document the file structures used.

2. USER GUIDE

2.1 ORIGAMI AUTOMATOR OVERVIEW

The ORIGAMI Automator project within Fulcrum is a tool designed to manage large analyses consisting of hundreds to thousands of SCALE/ORIGAMI depletion and decay calculations. The main purpose of this guide is to describe the various components of the ORIGAMI Automator GUI, which necessarily begins with some description of how the GUI is organized. At the top level are a set of *panels*. For example, the side panel contains navigation items that are used to select pages where the required data are entered. In this guide, words in *italics* denote keywords displayed on the GUI screens, or *panels*. Words in quotation marks (“ ”) denote words to be entered by the user or selected from dropdown menus. Roman numerals (I, II, III) are used to label the navigation items. Letters (A, B, C) are used to label various panels. Arabic numerals (1, 2, 3) are used to label specific parts of a GUI screenshot. This scheme is illustrated in Fig. 1.

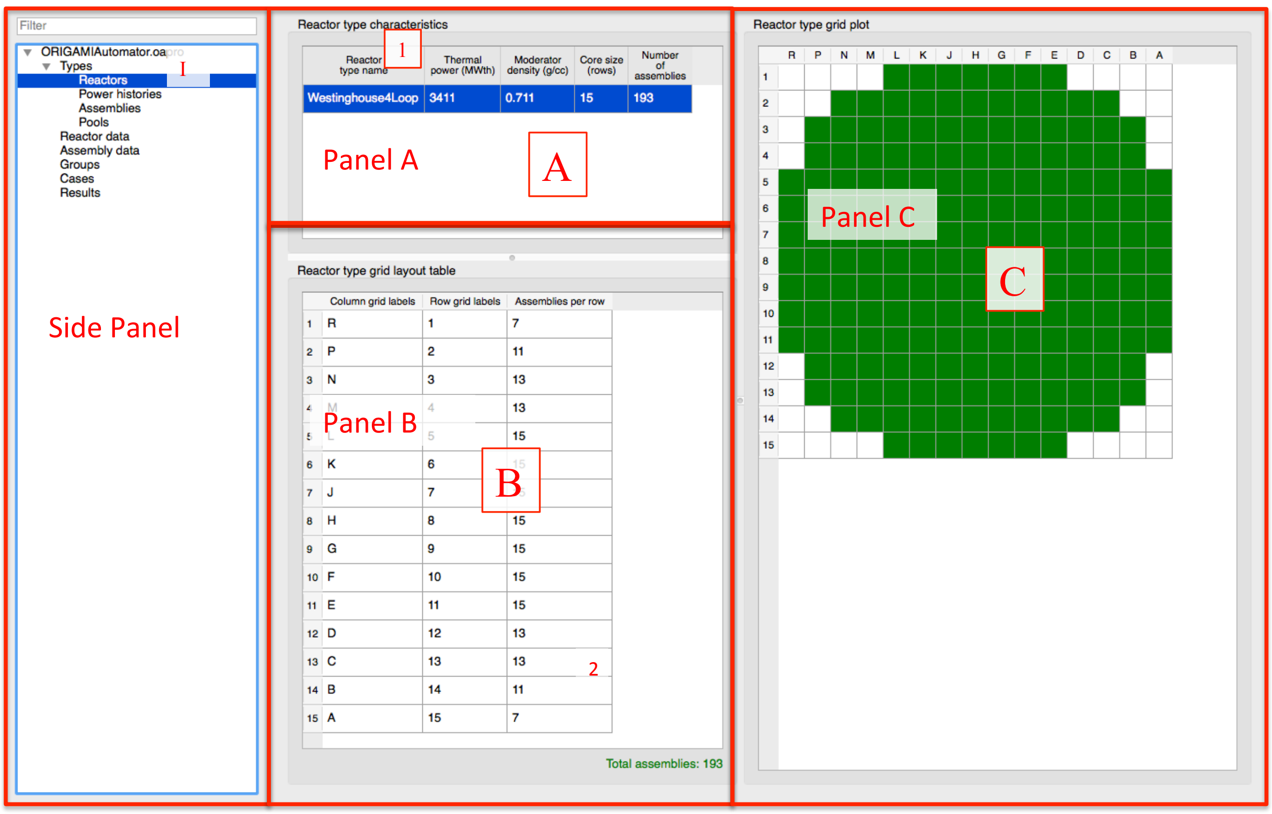


Fig. 1. Layout of an ORIGAMI Automator navigation item showing labeling conventions.

Data are stored in a set of JavaScript Object Notation (JSON) files, which are described in Appendix A. JSON is a standard simple text data format that can be easily manipulated with a text editor. Each GUI panel represents either

- a means to view and/or manipulate some or all data in a particular JSON file, or
- a special view of the data in a JSON file or other file (e.g., output results).

The information flow of the ORIGAMI Automator tool is shown in Fig. 2.

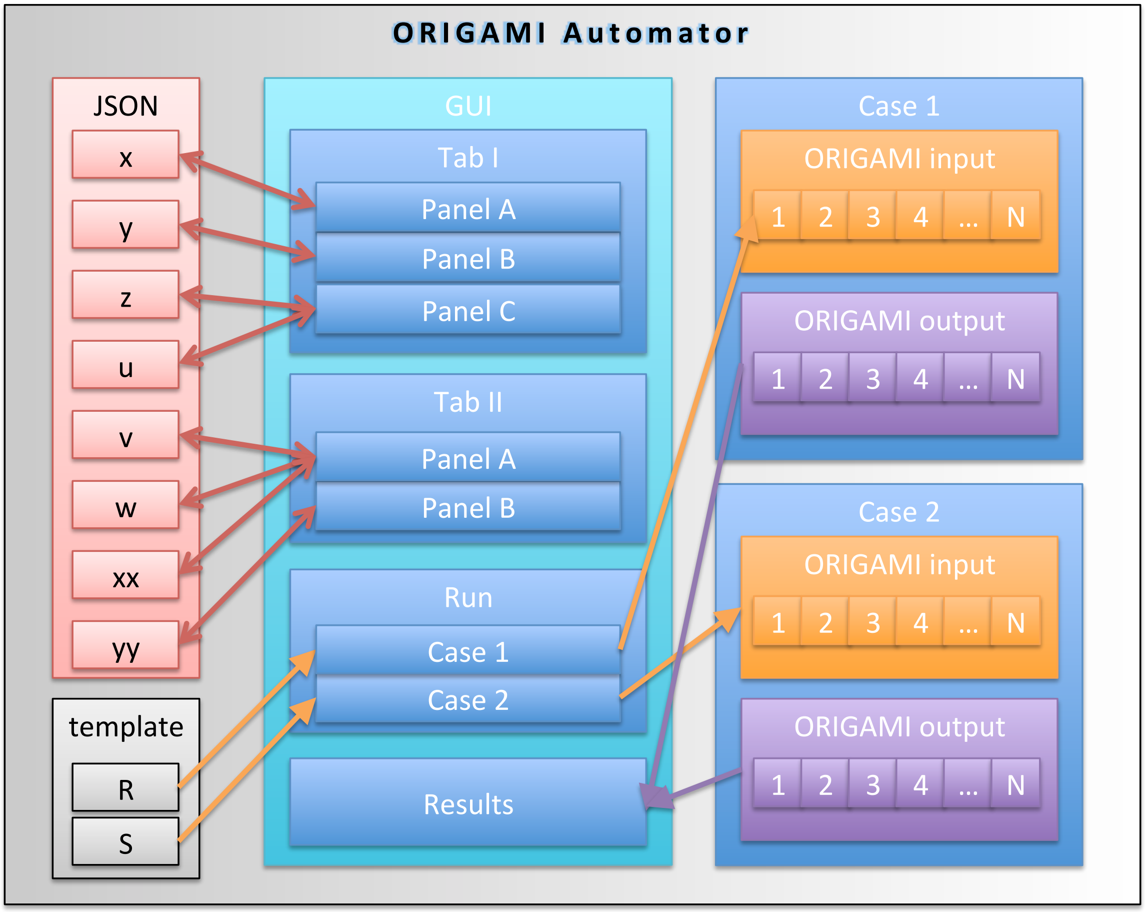


Fig. 2. Information flow of the ORIGAMI Automator tool.

Moving from left to right through Fig. 2, the JSON data files represent the type of database with which the majority of the GUI navigation items and panels interact. The connection to the JSON data files is two-way, meaning the GUI can read and write the files. As long as the GUI is open, the files are synchronized with all data present in the GUI.

The template approach implemented in Used Nuclear Fuel Storage, Transportation, and Disposal Analysis Resource and Data System (UNF-ST&DARDS) [2] has been used in this work, which results in an easy-to-use system for routine analysis that can be modified for special analyses by altering the underlying templates. As in UNF-ST&DARDS, the standard JSON format has been used for data files, and the UNF-ST&DARDS TemplateEngine has been used to process template files. However, a deliberate departure from UNF-ST&DARDS has been made in the use of JSON files for ORIGAMI Automator's database. A JSON database is used rather than a conventional database because JSON is much easier to inspect and modify by hand.

The *Cases* navigation item allows a user to select a particular time of interest for the analysis and then create all of the relevant ORIGAMI inputs (usually thousands of input files). This operation is a one-way operation drawing on user-defined templates (shown above as R and S) which contain all the necessary information to expand the specifications provided in the JSON files into fully qualified SCALE input files after expansion through the TemplateEngine. The templates are described in more detail in Appendix B. The operation combines templates with information generated from within the GUI, writes the relevant ORIGAMI input files, and then starts SCALE jobs in the background to run the cases.

The final *Results* navigation item allows the user to read ORIGAMI output files (i.e., ORIGEN binary concentration files) and make various plots using the OPUS utility in SCALE, thus representing another one-way operation.

2.1.1 Import Options

ORIGAMI Automator is designed to analyze all of the fuel assemblies at a single nuclear plant site. If this site includes multiple reactors and SFPs, several thousand data files are required to describe the site, reactor types, and assembly designs, as well as the operating history of each assembly. To ease the initial import of a large data set, a Perl utility has been written to transform a set of comma-separated value (CSV) files into JSON files, as described in Appendix C. The advantage of the CSV format is that data is contained in a single file, making the data convenient to generate. The JSON files used by ORIGAMI Automator have been defined as small self-contained data packets, and they are hierarchically organized in the file system. This makes it more convenient for the user to view/modify small subsets of data. Thus, as shown in Fig. 3, there are two basic options to initially populate the JSON files:

1. convert the raw data to CSV first and then use the Perl import script described in Appendix C, or
2. convert the raw data directly to JSON files using the hierarchy and definitions in Appendix A.

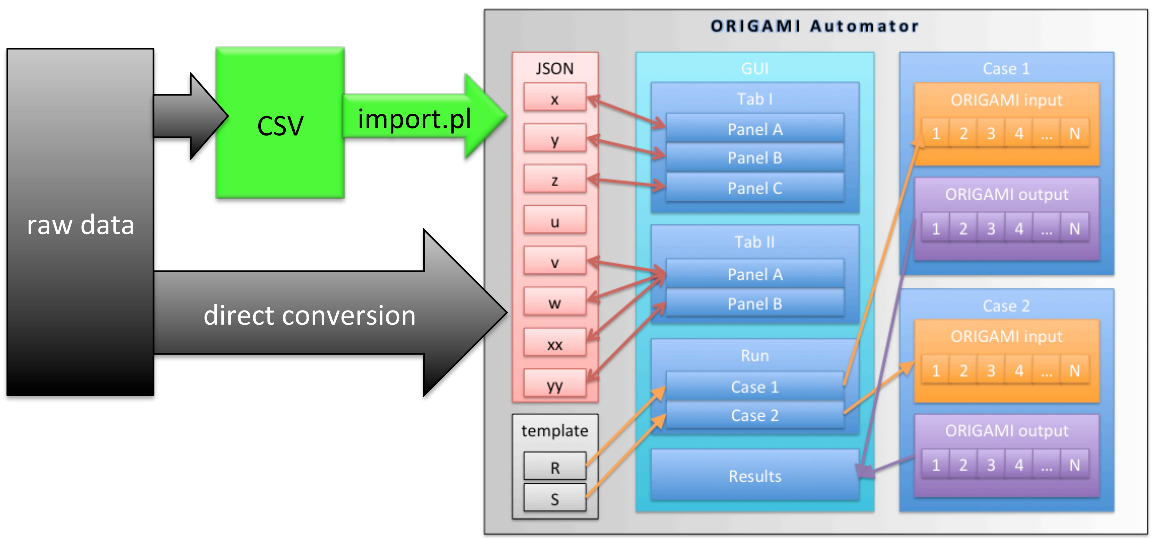


Fig. 3. Import options to initially populate JSON data files.

2.1.2 Export Options

The results visualization capability in the GUI is based on processing the ORIGEN binary concentrations file (output by ORIGAMI) and applying in-memory OPUS operations on them. The export of data in the MELCOR format is provided for each group created in the ORIGAMI Automator.

2.2 GUI NAVIGATION ITEM DESCRIPTIONS

The high-level navigation items in ORIGAMI Automator are shown in Fig. 4.

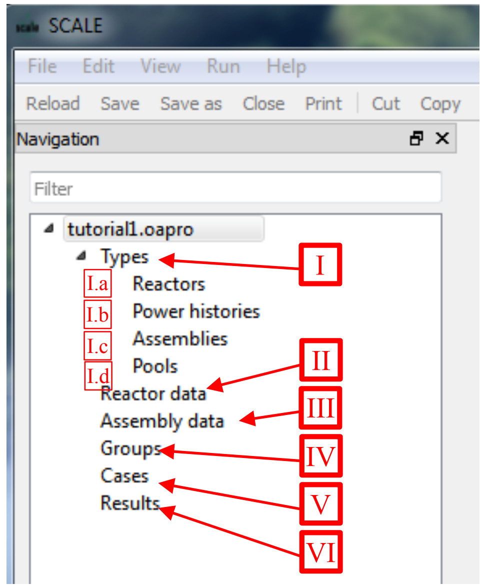


Fig. 4. Navigation items for the ORIGAMI Automator project within Fulcrum.

The following list briefly describes each navigation item and gives an example of its usage.

I. The **Types** navigation item contains all navigation items that define reusable types, including reactor core types, power histories, assemblies, and pools, as follows:

   a. The **Types/Reactors** navigation item includes defined reactor core types. Information about each reactor type includes the core size, design thermal power, and average moderator density.
   Example: The "Westinghouse4Loop" reactor type uses the core layout and base operational characteristics of a standard Westinghouse 4-loop reactor.
   b. The **Types/Power histories** navigation item is used to define the power produced during a given time period in relative terms such that it can apply to cycles of different durations.
   Example: A general "Coastdown" power history may include constant power for the first 99.0% of the cycle and then a linear ramp down to 50% power for the last 1.0% of the cycle.
   c. The **Types/Assemblies** navigation item is used to define each class of assemblies, including the total active fuel height and cross section libraries.
   Example: The "W17LOPAR" and "W17V5" assembly types could initially be defined with the same axial height of 365.76 cm, and generic W17 × 17 libraries. If different libraries were generated for each of these assembly types at a later time, the update would only require a change in a single place: this type file.
   d. The **Types/Pools** navigation item is used to create a pool by name. Pools can be used to assign SFP locations to an assembly or any other location (e.g. cask) where burnup is not accumulated.

II. The **Reactor data** navigation item is used to define an actual physical reactor by assigning it a name, a reactor type, and an operational history.
Example: The PWR could be defined as a "Westinghouse4Loop" type with operating history and power uprate history.

III. The **Assembly data** navigation item is used to characterize an assembly by assigning it a name (e.g., serial number), an assembly type, and an operational history.
Example: A specific assembly such as "5AZH" can be defined as a "W17LOPAR" that was initially inserted in Westinghouse 4 loop plant and depleted to a burnup of 15 gigawatt-days per metric ton of uranium (GWd/MTU).

IV. The **Groups** navigation item is used to declare groups of assemblies. When all single assembly calculations are complete, all assemblies in a group are combined into a single ORIGEN binary concentration file for the group and a set of MelMACCS inventory files for that group.

V. The **Cases** navigation item is used to create "Cases," each of which defines a small set of parameters such as the target time of interest and the assembly groupings relevant for this case.
Example: One case could be defined to investigate the in-core isotopics at middle-of-cycle, whereas another could define the decay heat in SFP after fuel discharge. Both cases are for the same site, but they differ in their target times.

VI. The **Results** navigation item is used to visualize results for individual assemblies and groups.
Example: After a calculation completes, the comparison of uranium (U) and plutonium (Pu) content in all assemblies can be viewed.

The following sections discuss each navigation item in more detail.

2.2.1 Types/Reactors Navigation Item

The *Types/Reactors* navigation item shown in Fig. 5 includes the following panels:

- *Reactor type characteristics* panel (A),
- *Reactor type grid layout table* panel (B), and
- *Reactor type grid plot* panel (C).

Reactor type characteristics (A) are defined by

- reactor type name (1),
- thermal power (MWth) (2),
- moderator density (g/cc) (3),
- core size (number of rows) (4), and
- number of assemblies (5).

The *Reactor type grid layout table* panel (B) provides additional parameters:

- column grid labels (6),
- row grid labels (7), and
- assemblies per row (8).

The "Total assemblies" text at the bottom of the layout table (9) will be green when the sum of the assemblies per row matches the total number of assemblies in the reactor type given by the core size parameter (4); otherwise, the text will be red.

The *Reactor type grid plot* panel (C) shows the available core locations in green as defined by the assemblies per row (8) with column and row labels 6 and 7. The core labels defined here may be used to specify the location of an assembly in the core for a particular cycle using the optional location parameter in the *Assembly data* panel.

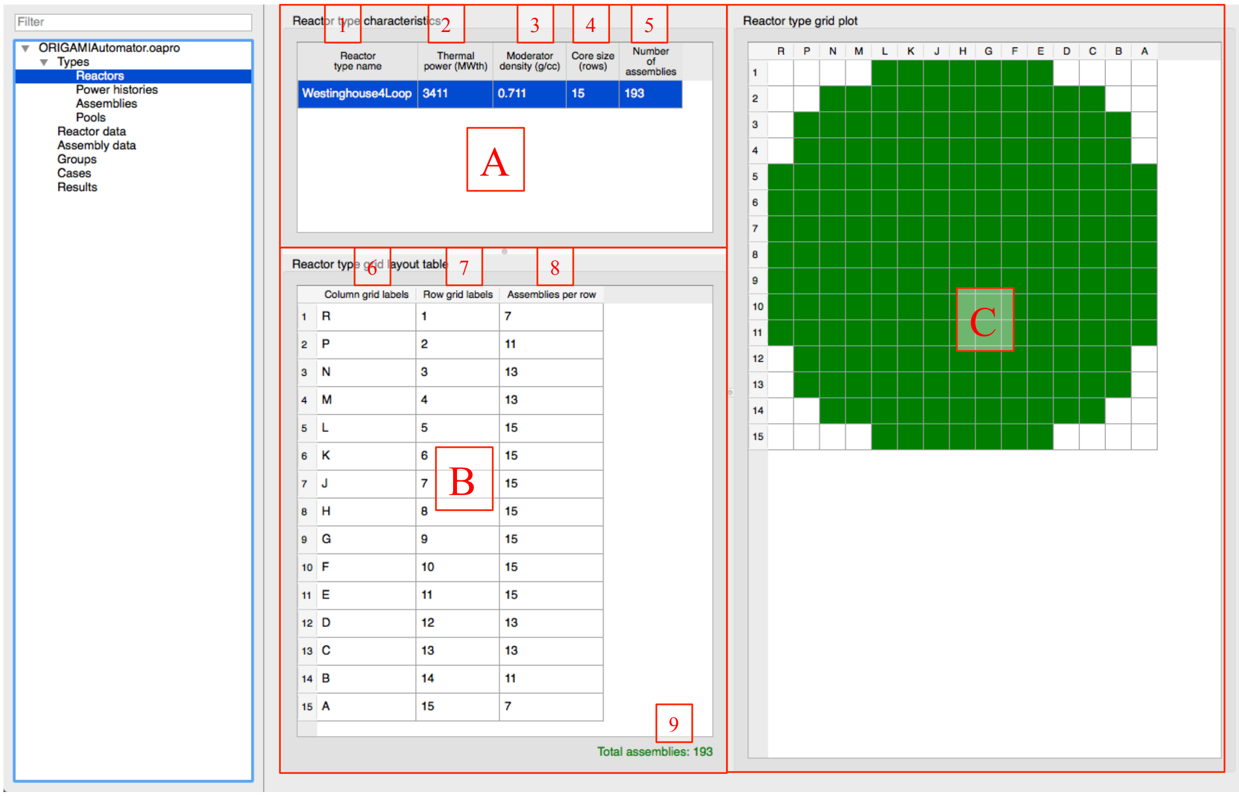


Fig. 5. *Reactors* navigation item of ORIGAMI Automator.

2.2.2 Types/Power Histories Navigation Item

The *Power histories* navigation items shown in Fig. 6 include the following panels:

- *Power history profiles* panel (A),
- *Power history table* panel (B), and
- *Power history plot* panel (C).

A new power history profile (A) is defined by

- name (1) and
- number of time steps (2).

The active power histories are shown in the *Power history table* (B) and *Power history plot* (C).

Power histories (B) are defined in terms of

- percent time (3) and
- percent full power (4)

so that they may be applicable to more than one cycle.

The power history simply provides a time-dependent, relative power scaling. Note that cycle power will always be scaled to satisfy the number of effective full-power days (EFPD) given in each cycle. For most cycles, a simple constant power history will be sufficient. A visualization of the power history is shown in the plot panel (C).

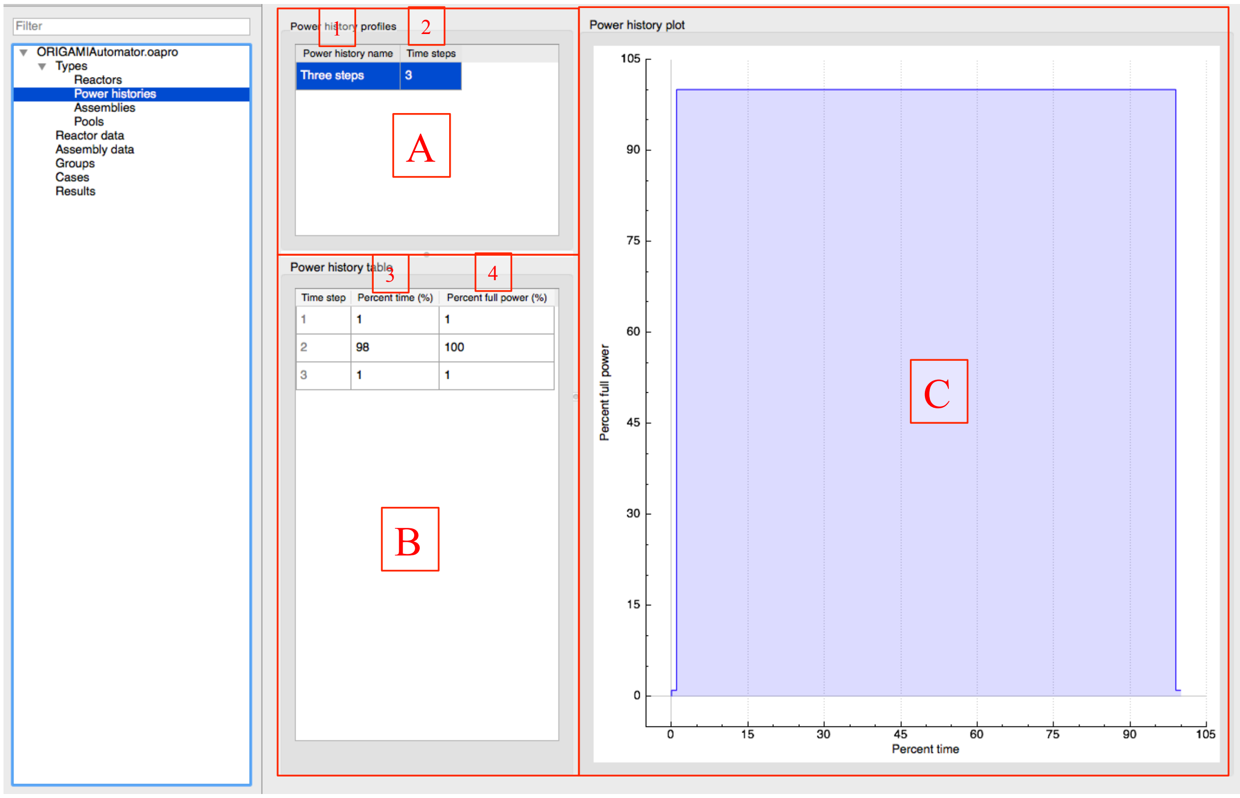


Fig. 6. ***Power histories*** **navigation item of ORIGAMI Automator.**

2.2.3 Types/Assemblies Navigation Item

The *Assemblies* navigation item shown in Fig. 7 includes the following panels:

- *Assembly type characteristics* panel (A),
- *Axial profile* panel (B), and
- *Axial profile plot* panel (C).

A new assembly type is created in the *Assembly type characteristics* panel A by defining

- the assembly type name (1),
- the number of axial nodes (2), and
- the total active fuel height (3).

The *Axial profile* panel (B) is then used to declare the following items for each axial node:

- axial height (4),
- cross section library name (5), and
- percent mass (6).

At the bottom of the table is the text “Total axial height” (7). This text will be green when the sum of the axial node heights for each node matches the total assembly height; otherwise, the text will be red. The axial zones allow for both axially varying ORIGEN libraries and axially varying fractions of fuel. This is especially important in boiling water reactor (BWR) bundles with vanished rods and axial enrichment and gadolinia variations. In pressurized water reactor (PWR) modeling, this capability is generally not needed.

The *Axial profile plot* panel (C) shows the current axial zone assignments.

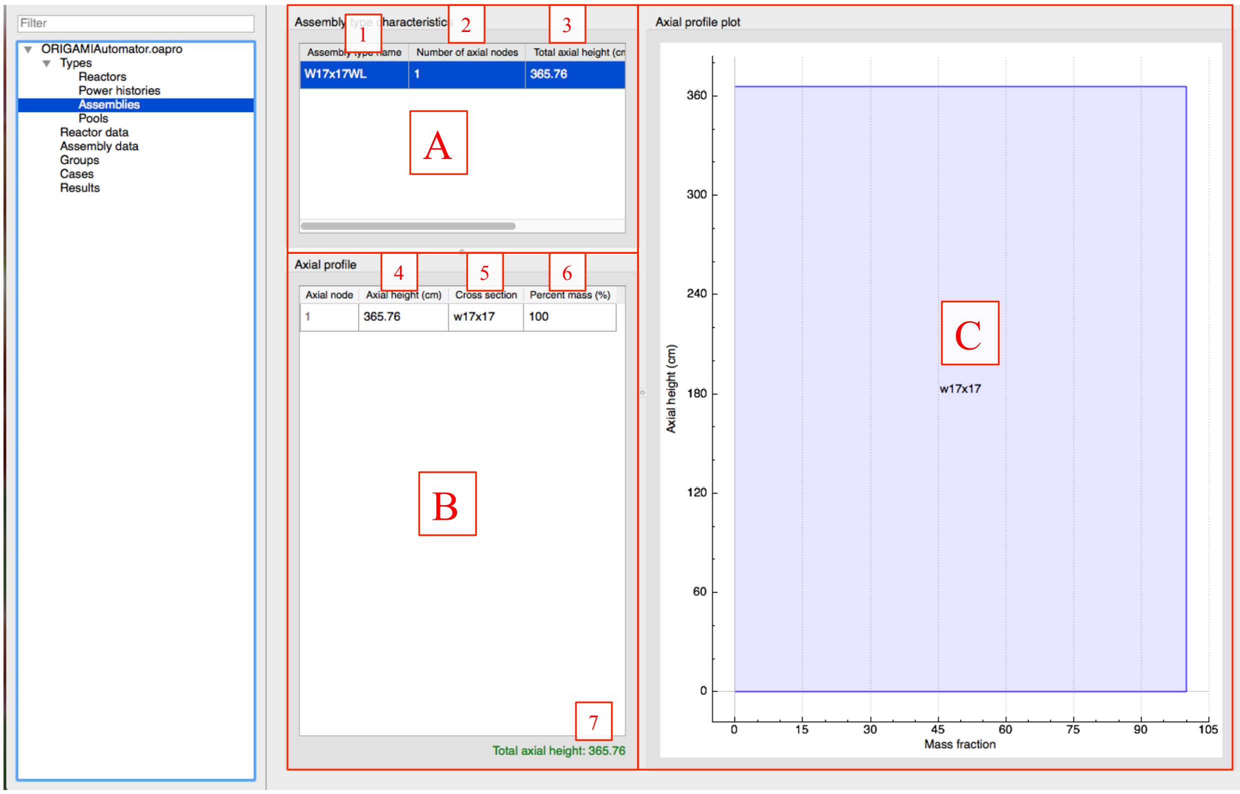


Fig. 7. *Assemblies* navigation item of ORIGAMI Automator.

2.2.4 Types/Pools Navigation Item

The *Pools* navigation item shown Fig. 8 includes a single panel for declaring a pool. Assemblies in pools do not accumulate burnup. A new pool type is created in the *Pool grid plot* panel (A) by defining

- the pool type name (1),
- the number of rows in the pool (2),
- the numbers of columns in the pool (3),
- and the number of assemblies in the pool (4).

Note: In this version, the Pool layout (B) and Pool grid plot (C) are not available for modification and the pool “type” creates a specific pool, unlike the reactor type which is a reusable definition specified when creating a new core.

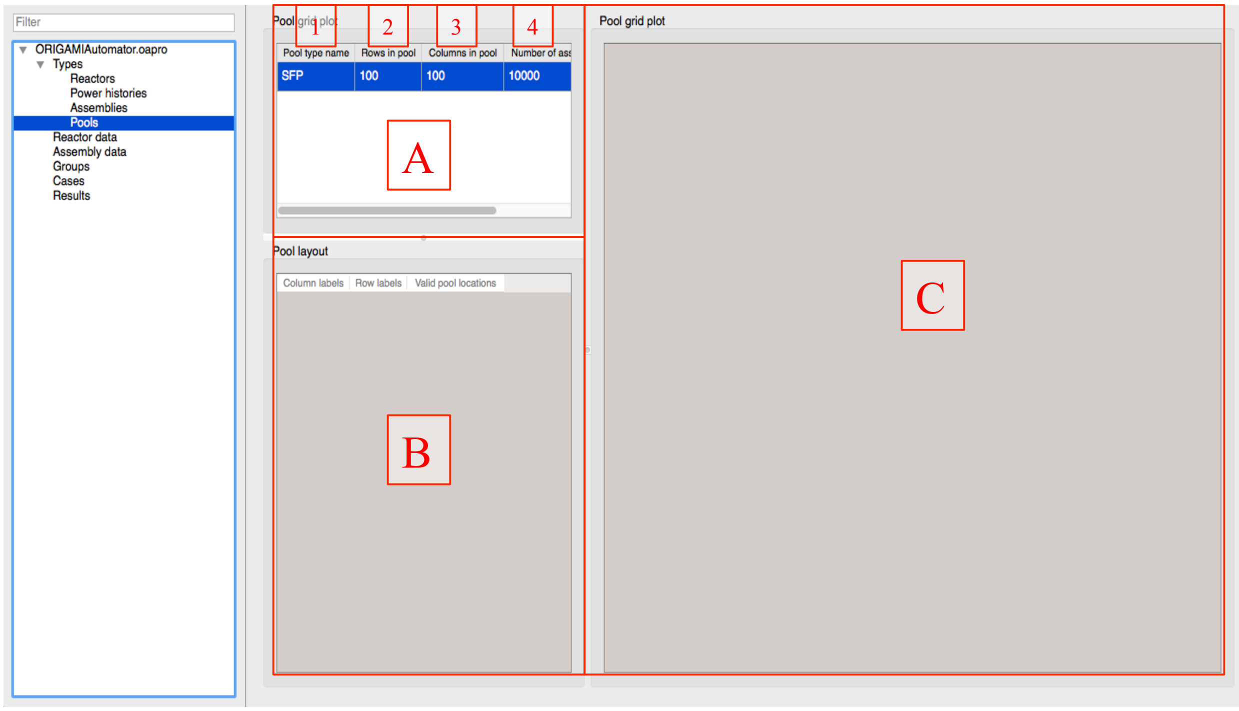


Fig. 8. ***Pools*** **navigation item of ORIGAMI Automator.**

2.2.5 Reactor Data Navigation Item

The *Reactor data* navigation item shown in Fig. 9 includes the following panels:

- *Reactors* panel (A),
- *Power uprate* panel (B), and
- *Cycle data* panel (C).

The *Reactors* panel (A) identifies each reactor by

- name (1) and
- reactor type (2).

The last column shows the design power (3) for the chosen reactor type.

The *Power uprate* panel (B) shows for the reactor selected in (A), a sequence of power uprates in terms of

- new rated thermal power (4) and
- effective date (5).

The power uprate modifies the absolute power (MWth) that corresponds to 100% relative power.

For the reactor selected in (A), the *Cycle data* panel (C) shows a sequence of cycles defined by

- cycle label (6),
- startup date (7),
- shutdown date (8),

- effective full power days (9), and
- power history (10).

The cycle label will typically be a simple number (e.g., 5), but it is treated as a string, so if there is a split cycle (5a and 5b), then it may be handled easily. The power history column (10) includes the names of those previously defined on the *Power histories* navigation item.

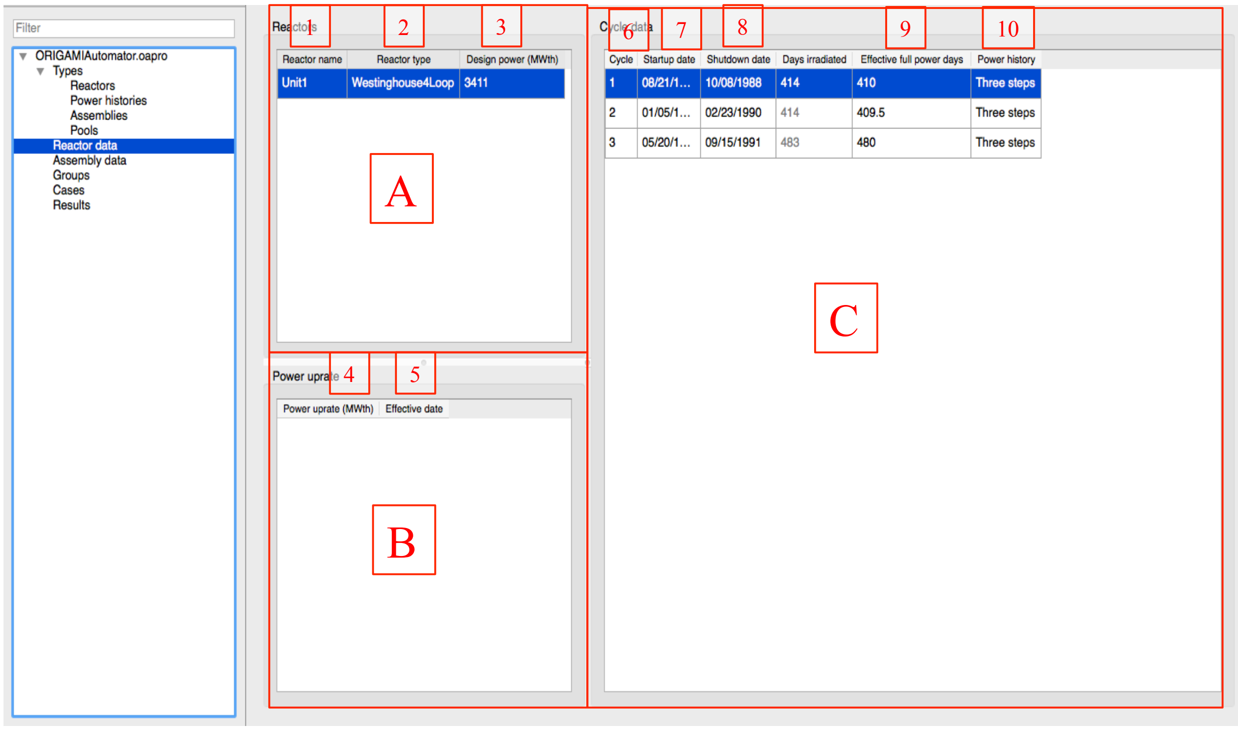


Fig. 9. *Reactor data* navigation item of ORIGAMI Automator.

2.2.6 Assembly Data Navigation Item

The *Assembly data* navigation item shown in Fig. 10 includes the following panels:

- *Assembly inventory* panel (A) and
- *Cycle data for assembly* panel (B) for the currently selected assembly in panel (A).

A new assembly may be created in the *Assembly inventory* panel (A) by defining the

- unique assembly ID (1),
- assembly type (2),
- fissile content (3),
- fuel composition (4), and
- the amount of heavy metal (5).

The only currently supported composition type is UO_2, so the fissile content of the fuel is the percent of ^{235}U for the UO_2 composition due to current limitations in ORIGAMI's composition specification. Other fuel compositions such as mixed oxide (MOX) fuel may be added in the future.

The *Cycle data for assembly* panel (B) is used to view/modify the operating history of the assembly. If it is moved to a named reactor or pool (6), then a cycle (7) and an incremental cycle burnup (10) must be supplied. In this case, the date (8) is implied by the cycle start date. However, if it is moved to a nonreactor (storage) location such as a pool, then the cycle (7) must be blank, the date (8) is optional, and the incremental burnup (10) must be zero. If the date (8) is not supplied, it is assumed to be equal to the cycle end date of the previous cycle. The position (9) is a label corresponding to the core row/column position or a storage pool location label. The total burnup of the fuel assembly summed over all of the cycles in which it has been irradiated is displayed in the bottom right of the screen (11).

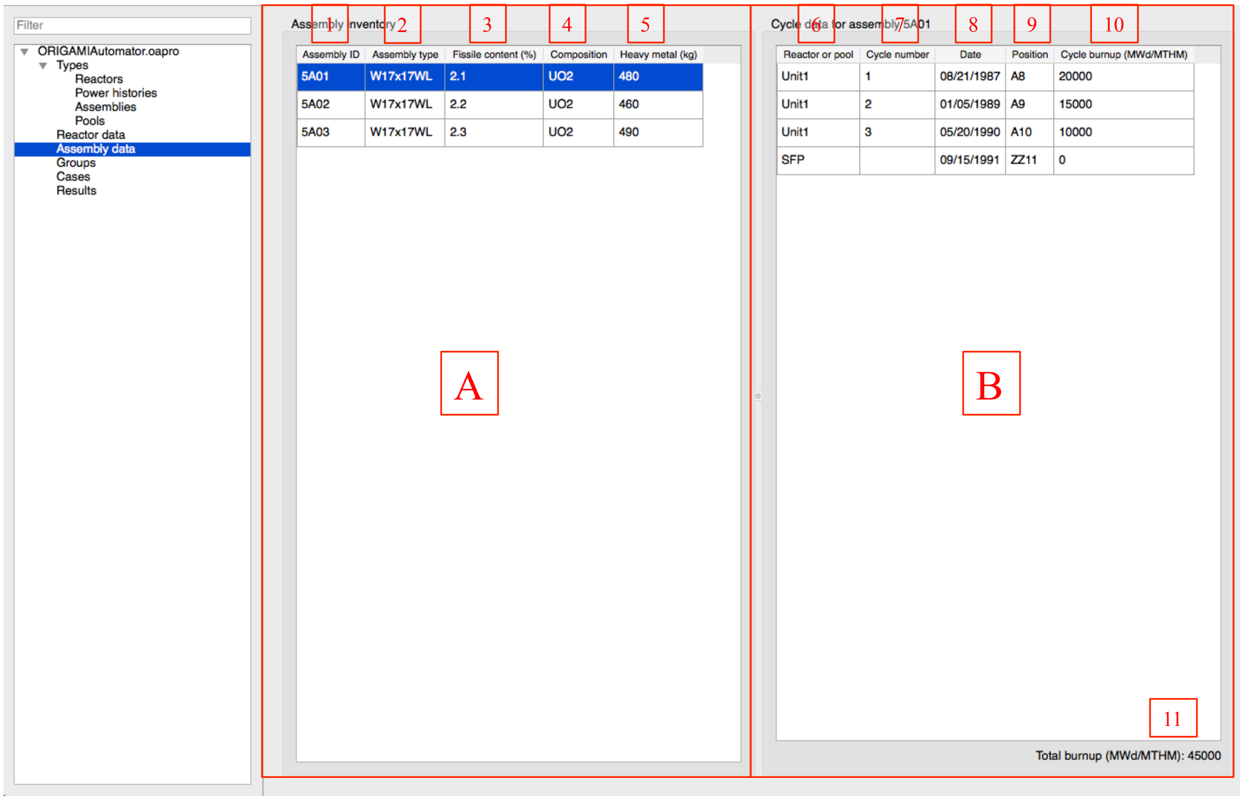


Fig. 10. *Assembly data* navigation item of ORIGAMI Automator.

2.2.7 Groups Navigation Item

The *Groups* navigation item shown in Fig. 11 includes a single panel (A) for creating a new group which combines all assemblies in a pool, reactor, or reactor in a specific cycle over time. A burnup range can also be supplied to include only assemblies in a specific range. Group calculations require assembly calculations, which are discussed in more detail for the next navigation item.

A new group may be created in the *Groups* panel (A) by defining the

- group name (1),
- reactor or pool name (2),
- reactor cycle (blank for an entire reactor group of assemblies or for a pool group of assemblies) (3),
- minimum burnup of the groups of assemblies (4), and
- maximum burnup of the group of assemblies (5).

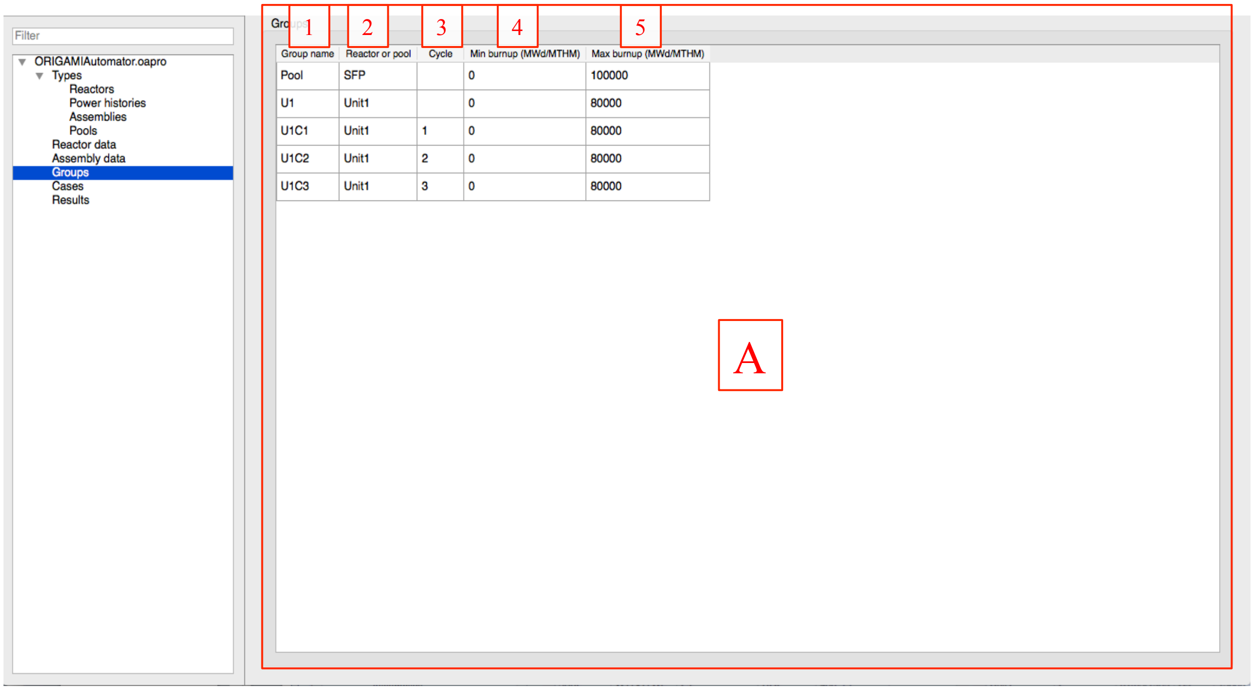


Fig. 11. *Groups* navigation item of ORIGAMI Automator.

2.2.8 Cases Navigation Item

The *Cases* navigation item shown in Fig. 12 includes the following panels:

- *Case to evaluate* panel (A) and
- *Group to evaluate* panel (B).

The *Cases to evaluate* panel (A) allows for setting

- the number of processors/threads (1) to use for the calculation,
- the target date for the results (2), and
- the location of the following files or directories:
  - irradiation template (3),
  - ORIGEN libraries directory (4),
  - SCALE executable (5), and
  - the output directory for calculation results (6).

When the *Run* button (7) button is pressed, the ORIGAMI inputs will automatically be generated using the selected template (3) and will be run with SCALE using the specified ORIGEN libraries (4) and executable (5). The results will be placed in the output directory (6). The *Groups to evaluate* panel (B) allows the user to regenerate groups in a specific case. The *Cancel* button (8), is used for canceling the individual assembly runs, the *groups to evaluate* (9) section is used to select which groups should be included in the results, the *Regenerate group results* (10) button is used to recalculate the results for each highlighted group, and the Cancel button (11) used to cancel the group calculations.

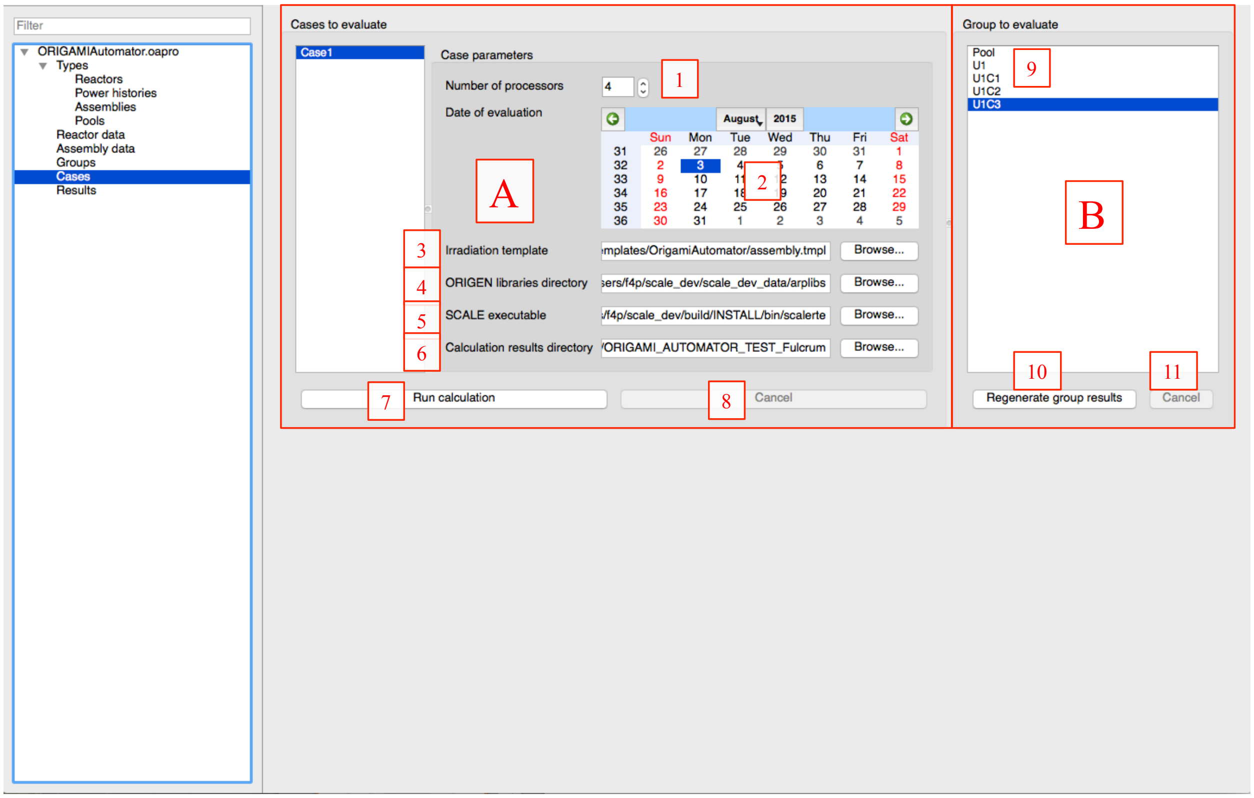


Fig. 12. *Cases* navigation item of ORIGAMI Automator.

A subdirectory in the output directory will be generated for each assembly and group. Each assembly subdirectory will contain the ORIGAMI input file, output file, and an ORIGEN concentrations (f71) file containing the result of the calculation. Each group subdirectory will contain the combined group concentrations (f71) file and a MelMACCS inventory file. An example of the subdirectories and files present in a calculation results directory is shown in Fig. 13. For details on how ORIGAMI Automator data are transformed into ORIGAMI inputs using the template system, refer to Appendix B.

```
├── 5A03
│   ├── 5A03.json
│   ├── 5A03.json.inp
│   ├── 5A03.json.inp.out.log
│   ├── 5A03.json.msg
│   ├── 5A03.json.out
│   ├── 5A03.json_5A03_AxialDecayHeat
│   ├── 5A03.json_5A03_assembly_dump.f71
│   ├── 5A03.json_5A03_compBlock
│   ├── 5A03.json_5A03_pinDumpMaster.f71
│   ├── 5A03.jsonerr.log
│   └── 5a03.ft33
├── Group_Pool
│   ├── Pool.f71
│   ├── Pool.ft33
│   ├── Pool.melmaccs.inv
│   ├── Pool_1990_03_16.melmaccs.inv
│   ├── Pool_1990_04_07.melmaccs.inv
│   ├── Pool_1990_04_28.melmaccs.inv
│   ├── Pool_1990_05_19.melmaccs.inv
│   ├── Pool_1991_09_15.melmaccs.inv
│   ├── Pool_1991_09_23.melmaccs.inv
│   ├── Pool_1991_10_01.melmaccs.inv
│   ├── Pool_1991_10_09.melmaccs.inv
│   ├── Pool_1991_10_17.melmaccs.inv
│   └── Pool_1991_10_18.melmaccs.inv
│
```

Fig. 13. Results subdirectories for assembly "5A03" and group "Pool."

Each group calculation creates MelMACCS inventory files. A table of contents file is created with the file name "*Group.melmaccs.inv,"* where *Group* is the group name, and individual inventory files are created as "*Group_YYYY_MM_DD.melmaccs.inv,"* where "*YYYY_MM_DD*" is the date corresponding to the time of evaluation. In a group calculation, every time that has nonzero concentrations results in an individual inventory file for that time state. An example of a MelMACCS table of contents inventory file is shown in Fig. 14, and a small portion of an isotopics file is presented in Fig. 15. For a particular MelMACCS scenario, a multistate MelMACCS case could be constructed by deleting some of the states from the "/CORE-LABEL" block and concatenating the relevant state inventory files to the reduced table of contents file. There are six sets of isotope data in each inventory file corresponding to mass and activity for activation nuclides, actinides, and fission products. There are 66 nuclides for actinide activity, 38 for actinide mass, 685 for fission product activity, 525 for fission product mass, 208 for activation product activity, and 194 for activation product mass. There are 1,090 unique nuclides.

```
/CORE-LABEL
U1C2_1989_01_05         "Group: U1C2 Date: Thu 05 Jan 1989 00:00:00 (t_days=503.0)"
U1C2_1989_01_09         "Group: U1C2 Date: Mon 09 Jan 1989 03:21:36 (t_days=507.2)"
U1C2_1989_04_20         "Group: U1C2 Date: Thu 20 Apr 1989 14:40:48 (t_days=608.6)"
U1C2_1989_07_31         "Group: U1C2 Date: Mon 31 Jul 1989 01:00:00 (t_days=710.0)"
U1C2_1989_11_09         "Group: U1C2 Date: Thu 09 Nov 1989 10:19:12 (t_days=811.5)"
U1C2_1990_02_18         "Group: U1C2 Date: Sun 18 Feb 1990 20:38:24 (t_days=912.9)"
U1C2_1990_02_23         "Group: U1C2 Date: Fri 23 Feb 1990 00:00:00 (t_days=917.0)"
U1C2_1990_02_23         "Group: U1C2 Date: Fri 23 Feb 1990 20:37:48 (t_days=917.9)"
U1C2_1990_03_16         "Group: U1C2 Date: Fri 16 Mar 1990 22:03:54 (t_days=939.0)"
U1C2_1990_04_07         "Group: U1C2 Date: Sat 07 Apr 1990 00:30:00 (t_days=960.0)"
U1C2_1990_04_28         "Group: U1C2 Date: Sat 28 Apr 1990 01:56:06 (t_days=981.1)"
U1C2_1990_05_19         "Group: U1C2 Date: Sat 19 May 1990 03:22:12 (t_days=1002.1)"
/END
```

Fig. 14. Example MelMACCS table of contents file.

```
****************************************************************
/CORE Pool_1990_03_16 MASS ACTIVATION
 H        1     2.160e-02
 H        3     1.943e-09
HE        3     1.614e-10
HE        4     1.581e+00
LI        6     7.542e-15
LI        7     5.646e-12
BE        9     2.064e-10
BE       10     3.246e-05
 B       10     3.122e-12
 B       11     1.160e-09
 C       12     4.197e-05
 C       13     5.048e+00
 C       14     3.424e-02
 N       14     5.143e-06
 N       15     3.109e-03
 O       16     6.572e+04
 O       17     2.659e+01
 O       18     1.512e+02
 F       19     8.362e-05
NE       20     1.848e-09
NE       22     2.254e-17
NA       23     5.013e-22
NA       24     5.144e-34
MG       24     1.152e-25
MG       25     2.262e-30
MG       26     1.369e-34
AL       27     1.499e-39
SI       28     2.315e-41
SI       29     1.416e-36
SI       30     2.788e-37
---lines 34-1741 not shown---
```

Fig. 15. Example MelMACCS isotopics inventory file.

2.2.9 Results Navigation Item

The *Results* navigation item shown Fig. 16 includes the following panels:

- *Navigation* panel (A) for assembly/group selection,
- *Data filtering (via OPUS)* output panel (B), and
- *Plot/Table* panels (C).

The *Navigation* panel displays a tree structure for each case (or group) created in the application, along with its associated assemblies. When the user selects an assembly, the displayed concentration results are updated in the *Plot* and *Table* navigation items. If no such results exist, the plot and table will be cleared to indicate that no data were found.

If the results exist, the parameters specified in the *Data filtering (via Opus)* panel will be used to filter the results to show only what the user requested. Any changes to the controls will cause the plot and table views to be updated automatically. The following items describe the data filtering capabilities provided by the given controls:

- Navigation (1) provides a navigation tree by Case, and one level deeper Groups and Assemblies.
- [elements] (2) plots values by element instead of nuclide.
- [symnuc] (3) is a list of nuclides or elements the user may explicitly request to be plotted.
- [symnuc] filter set (4) contains predefined list of to be ploted.
- [libtype] (5) allows the user to specify whether to include light elements, actinides, fission products, or a combination of the three.
- [nrank] (6) is used to specify that the top *nrank* number of nuclides is to be included when viewing the results. This may be used in conjunction with [symnuc] to set the final list of nuclides to plot. A value of zero (0) indicates that [nrank] will not be applied to the filter.
- [plot units] (7) indicates what concentration units are used to display the results in the plot or table.
- [sort units] (8) is used to specify what units are used to sort the results before they are displayed.
- [time units] (9) indicates what time units are used to display the results in the plot or table.

The output can be displayed either in a plot (10) or a table (11).

Examples of the various capabilities are shown in the following figures.

- Fig. 17 shows an example with “elements” and “symnuc” used to examine total activity from I and Cs.
- Fig. 18 shows the dropdown box with the available plotting units.
- Fig. 19 shows the dropdown box with the available time units.
- Fig. 20 shows the table view. The entire table may be copied and pasted into a spreadsheet or text file.

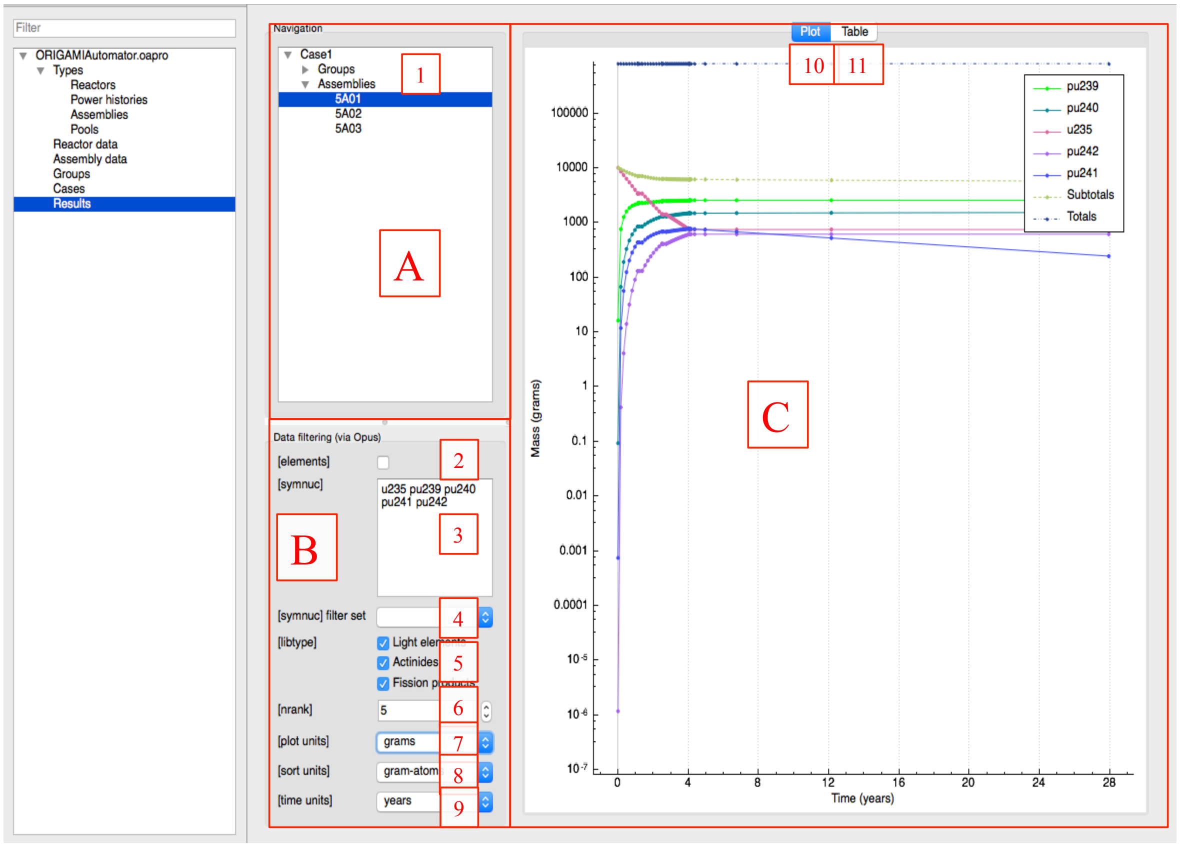


Fig. 16. *Results* navigation item of ORIGAMI Automator.

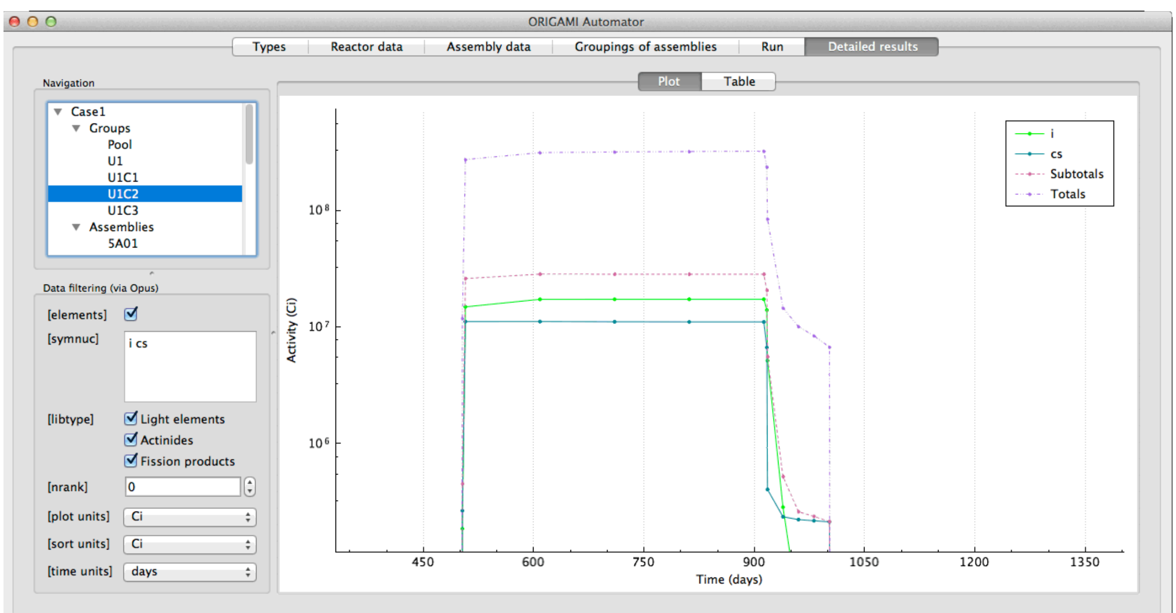


Fig. 17. Example using elements.

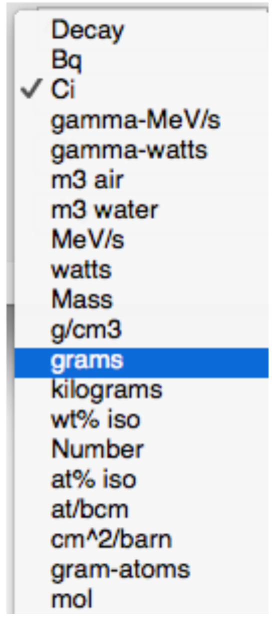


Fig. 18. Available plotting units.

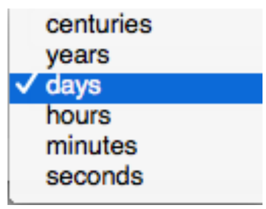


Fig. 19. Available time units.

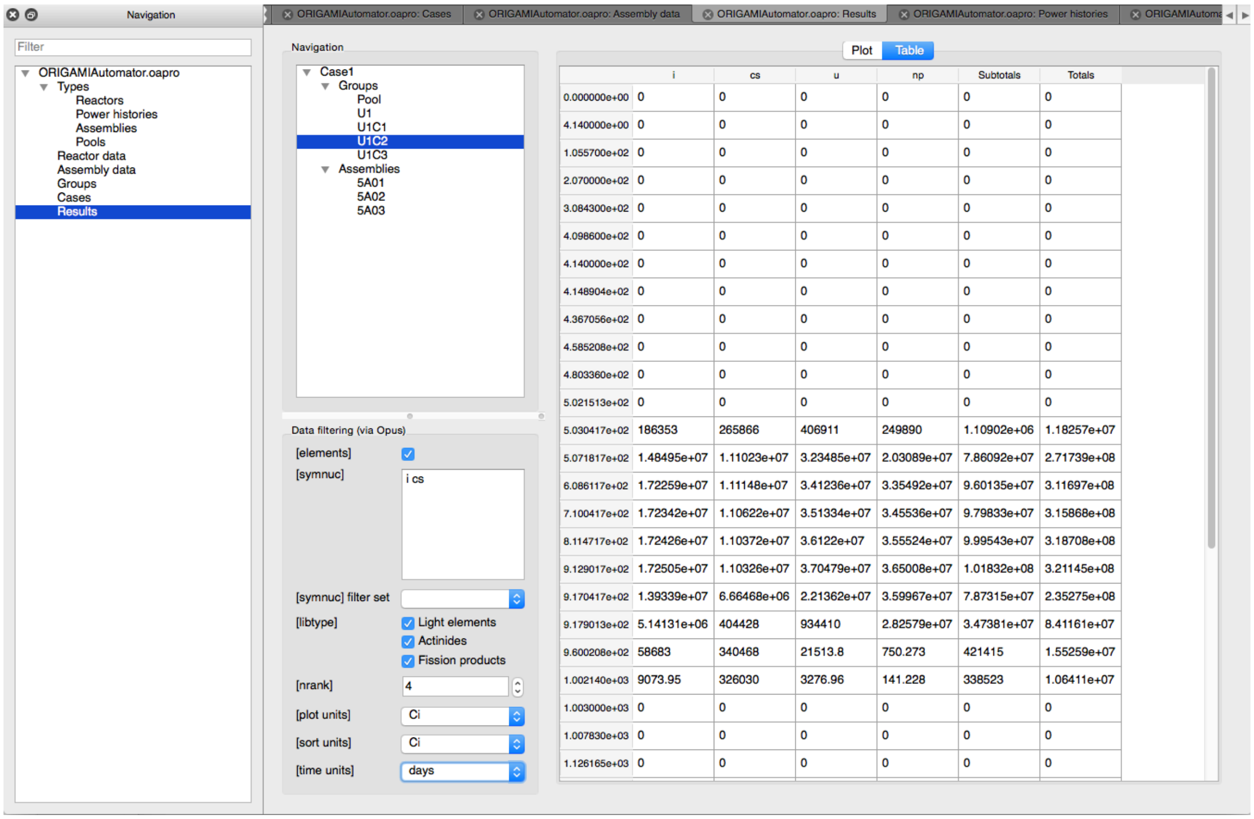


Fig. 20. Table view.

3. TUTORIAL

This section provides a step-by-step tutorial for creating a simple reactor model.

Launch SCALE's Fulcrum GUI, which resides in the bin directory of a SCALE installation as *Fulcrum*, and is typically installed as a shortcut on the Windows desktop or in the Mac Applications directory. Once the GUI appears, click on *File -> New ORIGAMI Automator project…* as shown in Fig. 21. In the open dialog, create a folder called "Tutorial1" and save the ORIGAMI Automator project file as "tutorial1.oapro", as shown in Fig. 22. The "tutorial1.oapro" project file and the folders that will be created inside "Tutorial1" completely described the ORIGAMI Automator environment and may be moved together, or the parent folder "Tutorial1" renamed, without any loss of information. saves project details. double click.

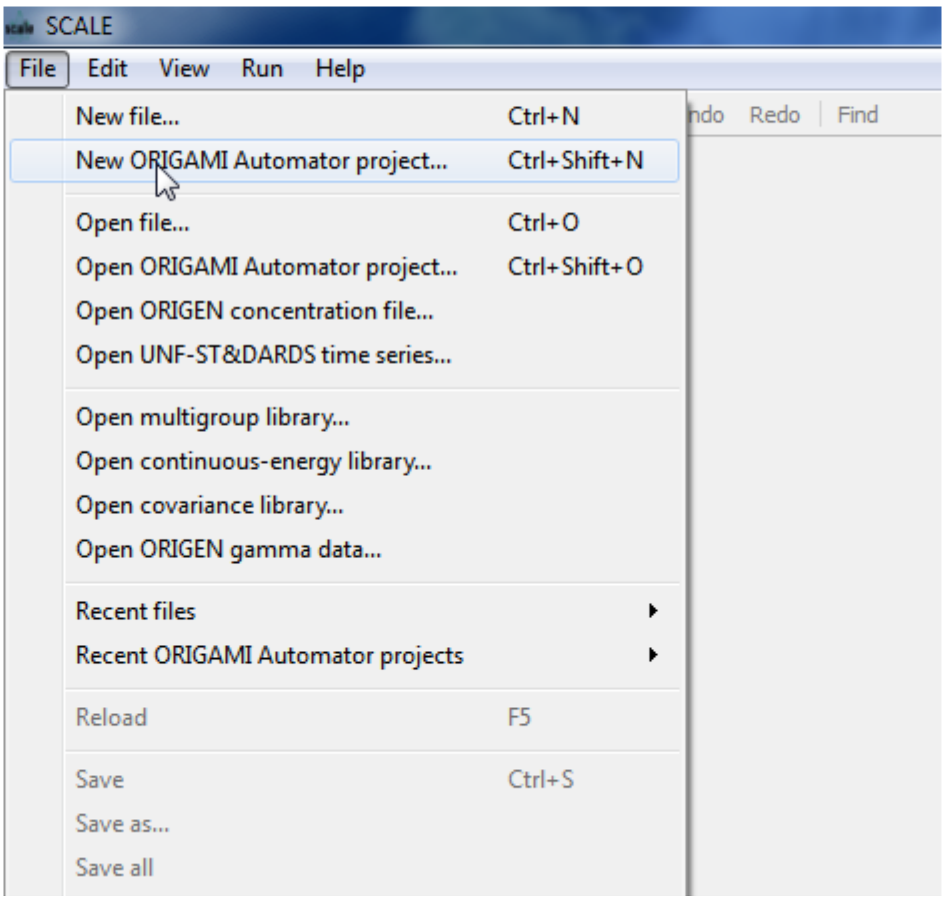


Fig. 21. Create a new project.

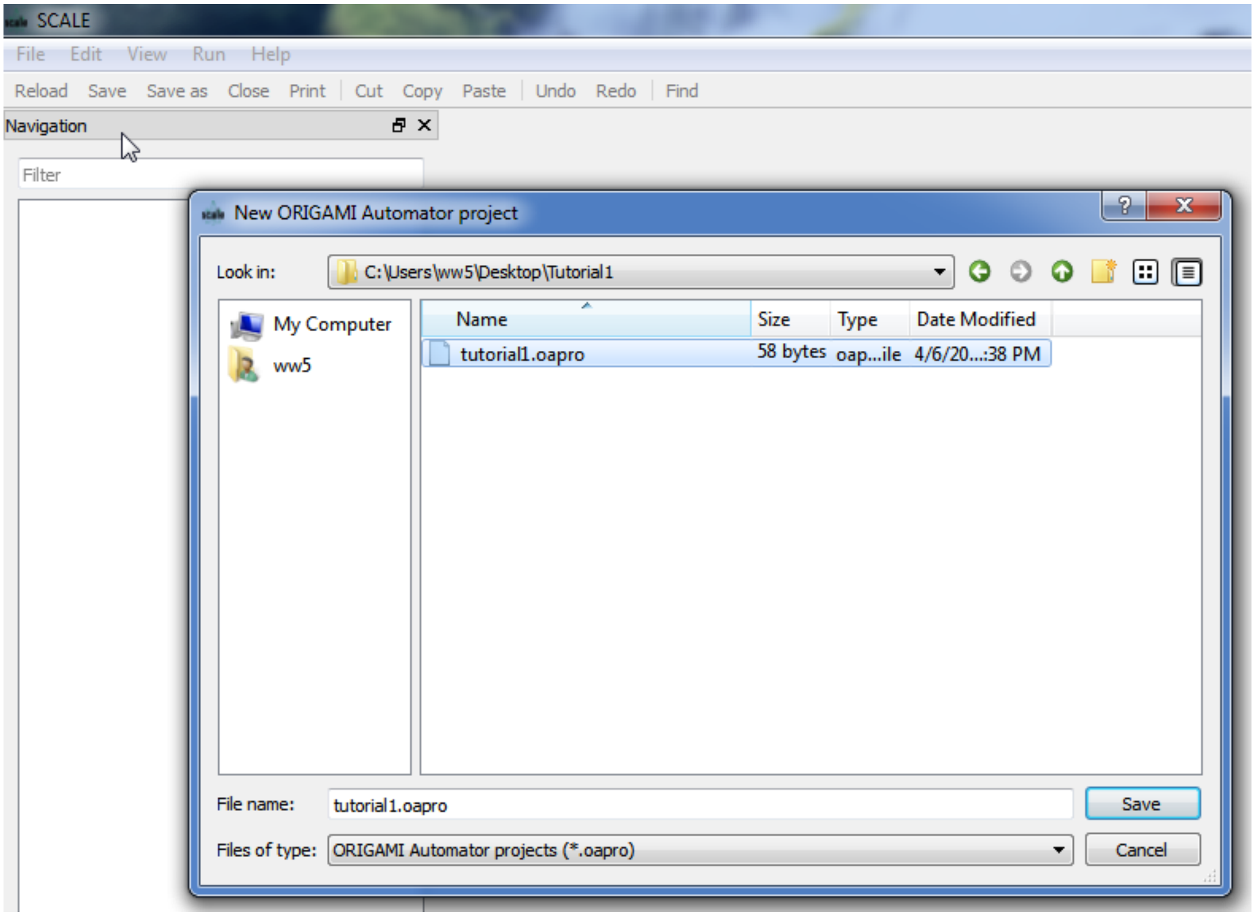


Fig. 22. Save ORIGAMI Automator project file in its own directory.

3.1 REACTORS NAVIGATION ITEM

Once the new project is created, double click on the first navigation item, *Types->Reactors*.Within the *Reactors* navigation item, right click within the empty *Reactor type characteristics* table and select *Add row*. You will be prompted to provide a name (*Reactor type identifier*) for the new reactor type; enter "Westinghouse4Loop" for this example. Then double click on each column of the table to add the required information to the *Westinghouse4Loop* row.

NOTE: You may have to scroll left to fill in all of the information. Enter the following information for this example:

Thermal power (MWth): 3411
Moderator density (g/cc): 0.711
Core size: 15
Number of assemblies: 193

When you specify the core size, the *Reactor type grid layout* table will be populated with the given number of rows; the next step is to fill out this table. The values to use can be seen in Table 1. You can use the navigation item button to move through the table, or you can double click on each entry individually. As you populate the *Reactor type grid layout* table, you will see the *Reactor type grid plot* updated automatically. This plot provides visual feedback that the layout table is correct.[1]

Table 1. Values to use for the *Reactors* grid layout table

Column grid labels	Row grid labels	Assemblies per row
R	1	7
P	2	11
N	3	13
M	4	13
L	5	15
K	6	15
J	7	15
H	8	15
G	9	15
F	10	15
E	11	15
D	12	13
C	13	13
B	14	11
A	15	7

When you have completed the *Reactors* navigation item, ORIGAMI Automator should look like Fig. 23. If the number of assemblies in the *Reactor type characteristics* section equals the sum of the assemblies per row in the *Reactor type grid layout* table, the text at the bottom of the table will be green, as shown in Fig. 23; otherwise, the text will be red.

[1] Currently, the core layout is not used in any calculations. In future versions the locations will be used for bookkeeping and visualization.

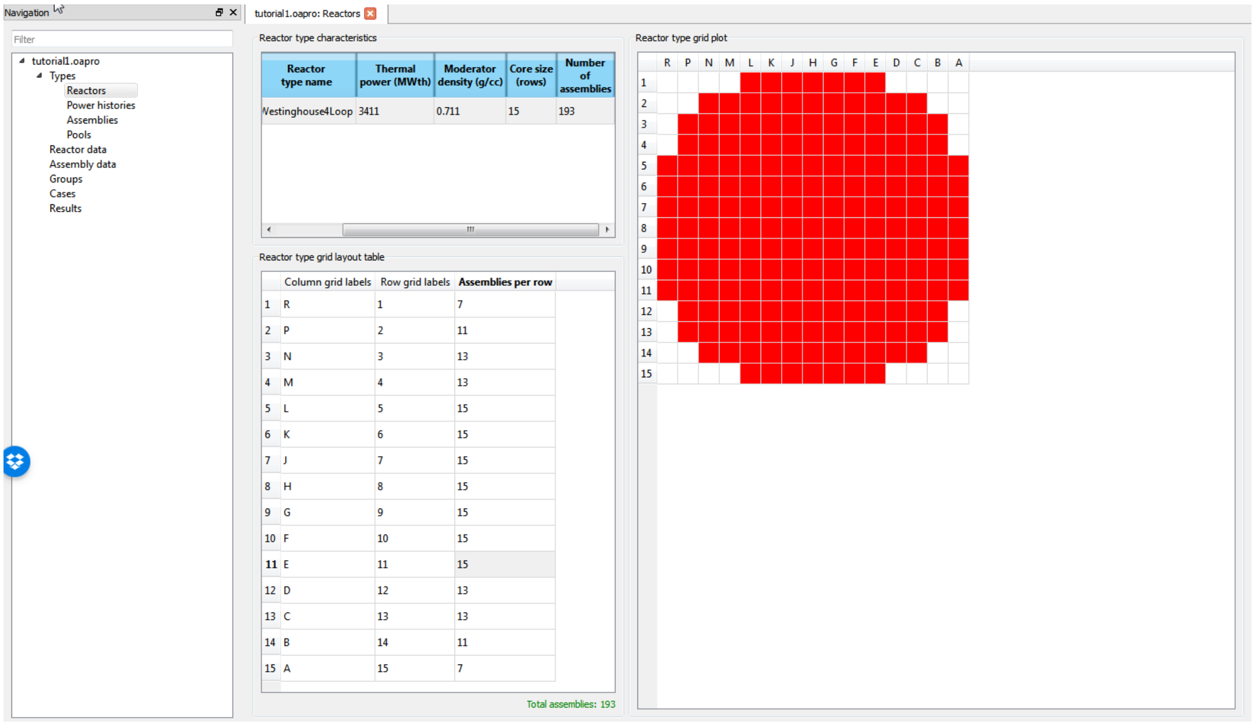


Fig. 23. Final view of the ORIGAMI Automator *Reactors* navigation item.

3.2 POWER HISTORIES NAVIGATION ITEM

Next, double click on the *Power histories* navigation item, and then right click within the empty *Power history profiles* table and select *Add row*. When prompted, enter the term “Three steps” for a new power history profile. Within the *Power history profiles* table, change the *Time steps* column from 0 to 3, and then click outside the *Time steps* field to update the size of the *Power history table* below. The *Power history table* will be populated with three rows. Fill out the *Power history table* with the data listed in Table 2 below.

NOTE: The *Time step* column within the *Power history table* is automatically generated and cannot be modified.

Table 2. Data to be entered into the *Power history* table

Time step	Percent time (%)	Percent full power (%)
1	1	1
2	98	100
3	1	1

Note that a power history simply provides a time-dependent "power multiplier" and will always be scaled to satisfy the user-provided EFPD for each cycle. When all information for the *Power history table* is entered, the *Power history* plot will be generated (see Fig. 24). Using standard mouse movements, the user can zoom in and out and pan on the *Power history plot.* When the *Power histories* navigation item has been completed, the ORIGAMI Automator screen should look like Fig. 24.

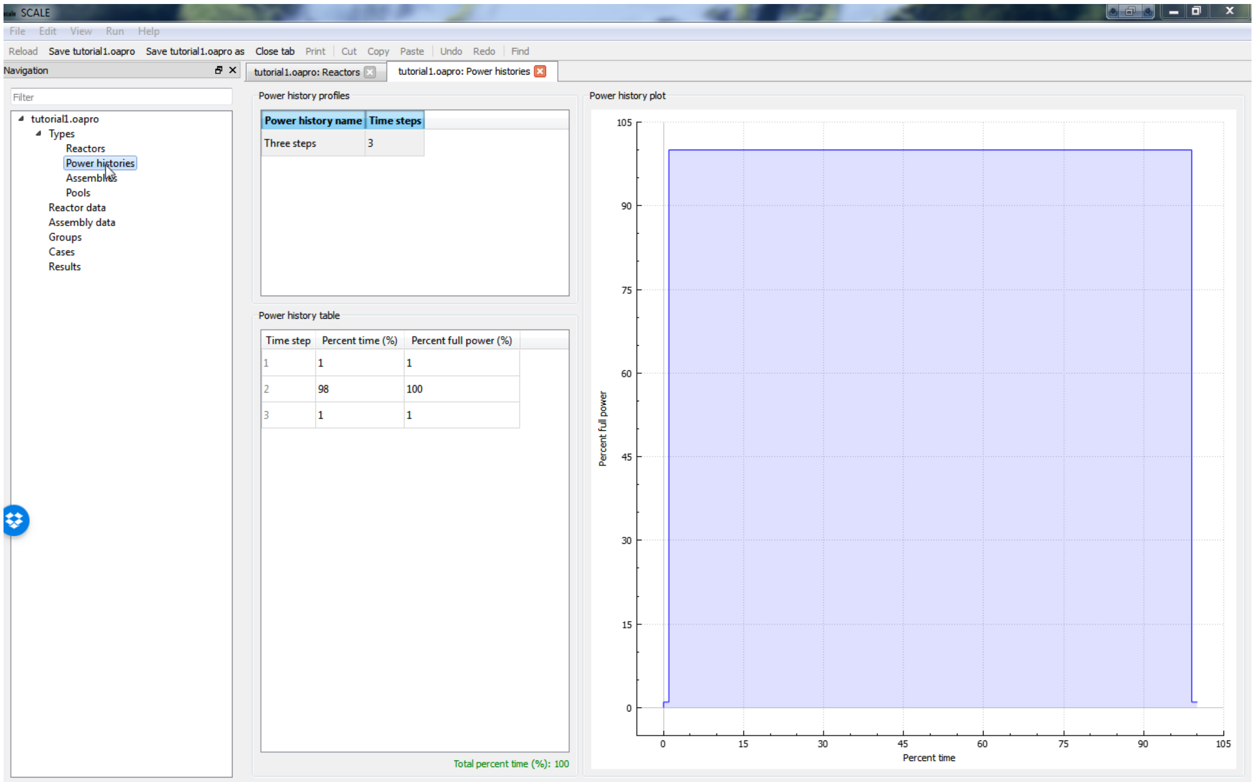


Fig. 24. Final view of the ORIGAMI Automator *Power histories* navigation item.

3.3 ASSEMBLIES NAVIGATION ITEM

Double click on the *Assemblies* navigation item. Right click in the empty *Assembly type characteristics* table within the *Assemblies* navigation item, select *Add row,* and enter "W17x17WL" in the row. Add the following information for the other two columns in the *Assembly type characteristics* table:

- Number of axial nodes: 1
- Total axial height (cm): 365.76

The *Axial profile* table is updated when the number of axial nodes is entered in the *Assembly type characteristics* table. In the *Axial profile table*, enter the following information:

- Axial height (cm): 365.76
- Cross section: w17x17
- Percent mass (%): 100

Note: The *Axial node* column is automatically populated with incremental integers using the data for each row starting with the bottom of the assembly as node 1. The cross section name must be manually entered, and it must match an ORIGEN library name in the ORIGEN cross section directory, which will be specified in the *Cases* navigation item in a subsequent step.

When the axial profile is entered, then the *Axial profile plot* is displayed. When you have completed the *Assembly data* navigation item, the ORIGAMI Automator screen should look like Fig. 25.

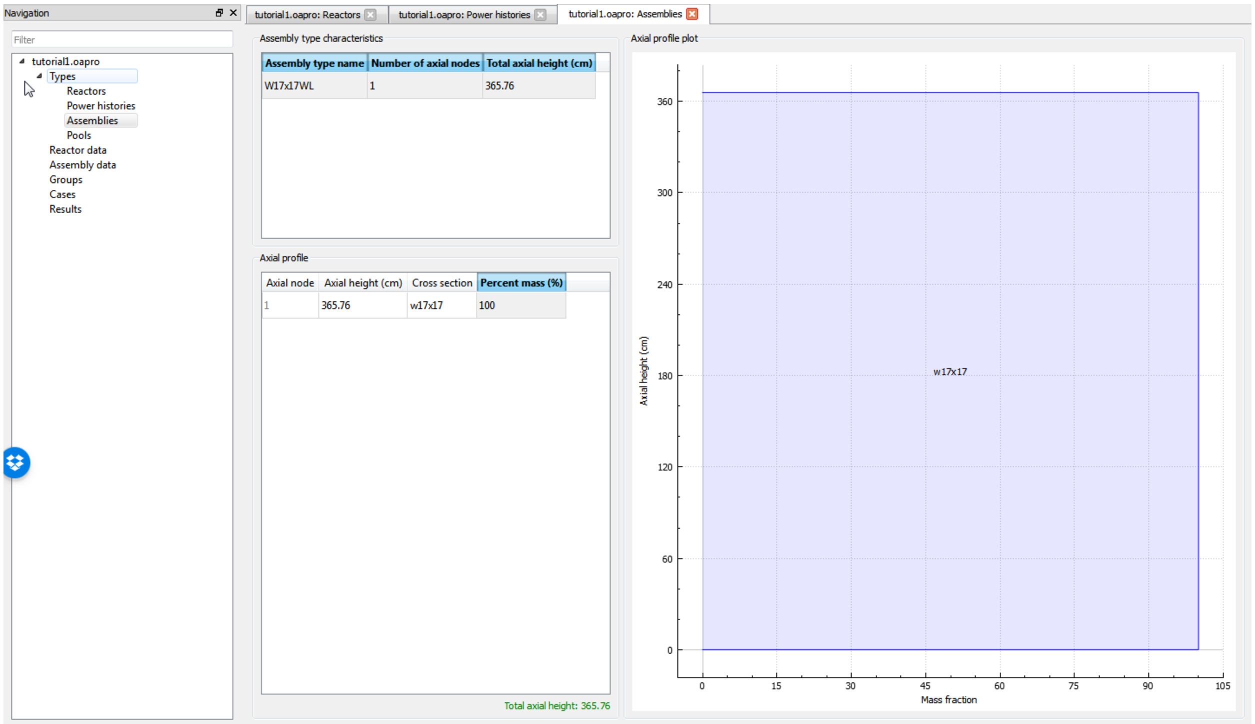


Fig. 25. Final view of the ORIGAMI Automator *Assemblies* navigation item.

3.4 POOLS NAVIGATION ITEM

Double click on the *Pools* navigation item. To create a pool type, right click in the table and select *Add row*. Then create a pool called SFP with 100 rows, 100 columns, and 10,000 assemblies (see Fig. 26).[2]

[2] The pool information is not currently used in any calculations, but in future versions, the locations will be used for bookkeeping and visualization.

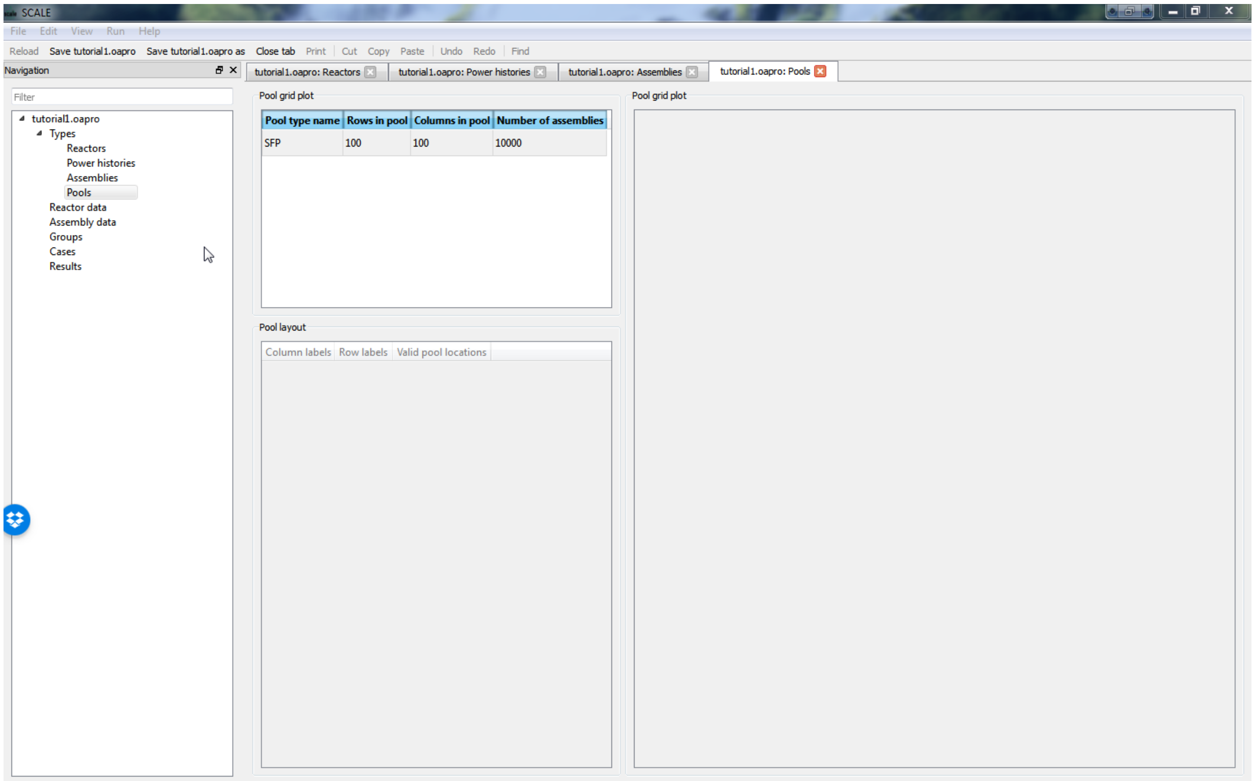


Fig. 26. Final view of the ORIGAMI Automator *Pools* navigation item.

3.5 REACTOR DATA NAVIGATION ITEM

Double click on the *Reactor data* navigation item. To create a reactor name in the *Reactors* panel, right click in the table, select *Add row*, and then enter “Unit1”as the name of the reactor. Once the *Reactor name* is selected, select “Westinghouse4Loop” from the dropdown menu that appears by double clicking in the *Reactor type* column. Deselecting the table will update the *Design power (MWth)* column.

NOTES:

1. The reactors included in the *Reactor type* drop down menu are based on the reactors created in the *Reactors* navigation item (Sect. 3.1).

2. Design power cannot be edited in the *Reactor data* navigation item, but it can be edited in the *Thermal Power (MWth)* column in the *Reactors* navigation item.

Information on power uprates for the reactor can be added by right clicking on the *Power uprate* table and selecting *Add row*. However, no power uprates are specified in this tutorial. To add data in the highlighted *Cycle data* table, right click in the table and select *Add row*. The information to enter into the *Cycle data* table can be seen in Table 3. A new row must be added for each cycle.

NOTE: the *Power history* column is selected from a dropdown menu, with the *Cycle data* information having been defined in the *Power histories* navigation item.

Cycle	Startup date	Shutdown date	Effective full power days	Power history
1	08/21/1987	10/08/1988	410	Three steps
2	01/05/1989	02/23/1990	409.5	Three steps
3	05/20/1990	09/15/1991	480	Three steps

Table 3. Information that should be used for the Cycle data table

When you are finished with the *Reactor data* navigation item, ORIGAMI Automator should look like Fig. 27.

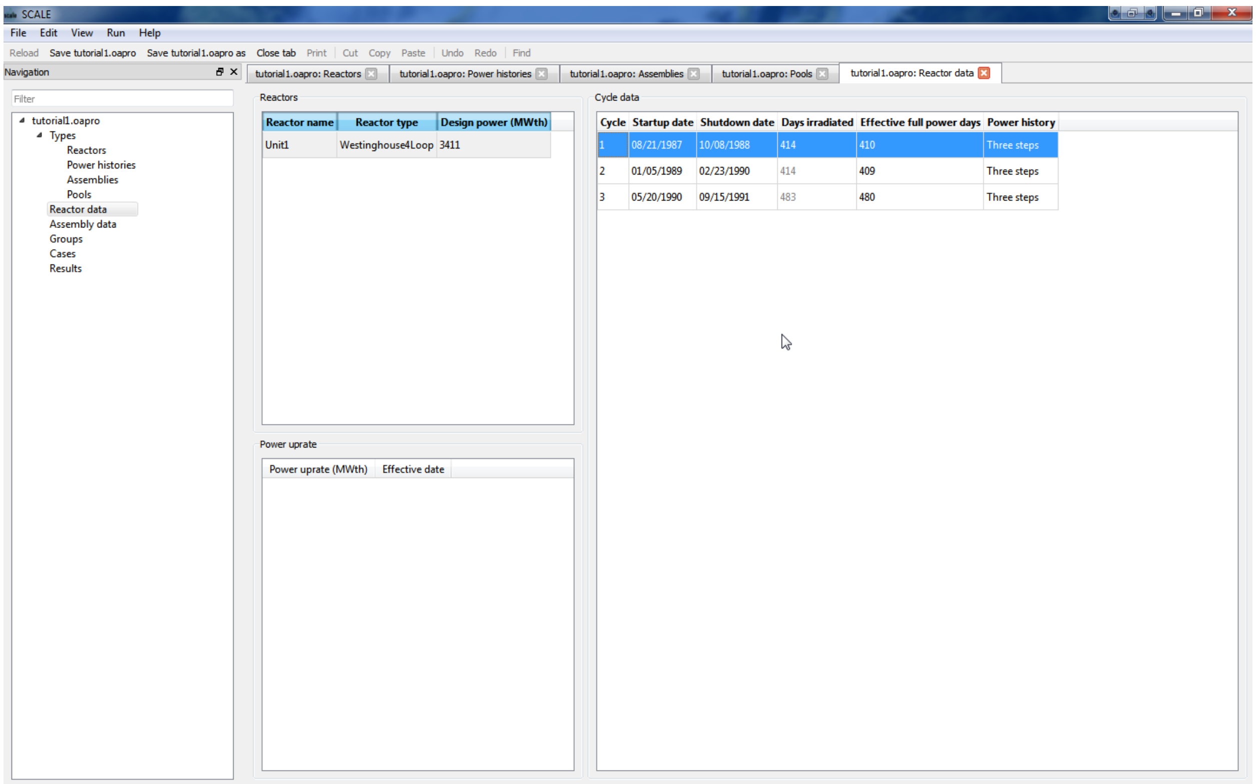


Fig. 27. Final view of the ORIGAMI Automator Reactor *Data* navigation item.

3.6 ASSEMBLY DATA NAVIGATION ITEM

The next navigation item that needs to be populated is the *Assembly data* navigation item. To create a new assembly, right click on the *Assembly inventory* table in the *Assembly data* navigation item, select *Add row*, and enter "5A01" as the *Assembly design identifier*. Enter the remainder of the data using the information from

Table 4. Create each assembly by right clicking on the table and selecting *Add row*.

NOTES:

1. The *Assembly type* column is a dropdown menu from data provided in the *Assemblies* navigation item.

2. The “Composition” column is a dropdown menu that currently has only one option (UO_2).

Table 4. Assembly inventory information for the *Assembly data* navigation item

Assembly ID	Assembly type	Fissile content (%)	Composition	Heavy metal (kg)
5A01	W17x17WL	2.1	UO_2	480
5A02	W17x17WL	2.2	UO_2	460
5A03	W17x17WL	2.3	UO_2	490

The next section to complete in the *Assembly Data* navigation item is the cycle data for each individual assembly. To enter the cycle data for assembly 5A01, double click on assembly 5A01 in the *Assembly inventory* table, right-click on the *Cycle data for assembly 5A01* table, select *Add row*, and enter “Unit1” in the field. Fill out the cycle number, position, and cycle burnup for assembly 5A01 based on the information in Table 5. Repeat the process described above for the other two assemblies.

NOTE: The *Date* column will automatically be populated based on the definition of that cycle from the *Reactor data* navigation item.

Table 5. Cycle data for the three different assemblies defined in the *Assembly data* navigation item

Assembly ID	Reactor or pool	Cycle number	Date*	Position	Cycle burnup (MWd/MTHM)
5A01	Unit1	1	08/21/87	A8	20,000
	Unit1	2	01/05/89	A9	15,000
	Unit1	3	05/20/90	A10	10,000
	SFP		09/15/91	ZZ11	-
5A02	Unit1	1	08/21/87	B8	10,000
	Unit1	2	01/05/89	B9	15,000
	Unit1	3	05/20/90	B10	7,500
	SFP		09/15/91	AAZZ1	-
5A03	Unit1	1	08/21/87	C8	15,000
	Unit1	2	01/05/89	C9	15,000
	SFP		02/02/90	AAZZ2	-

* Cycle dates are automatically generated, and pool dates are entered manually.

When you are finished with the *Assembly data* navigation item, ORIGAMI Automator should look like Fig. 28.

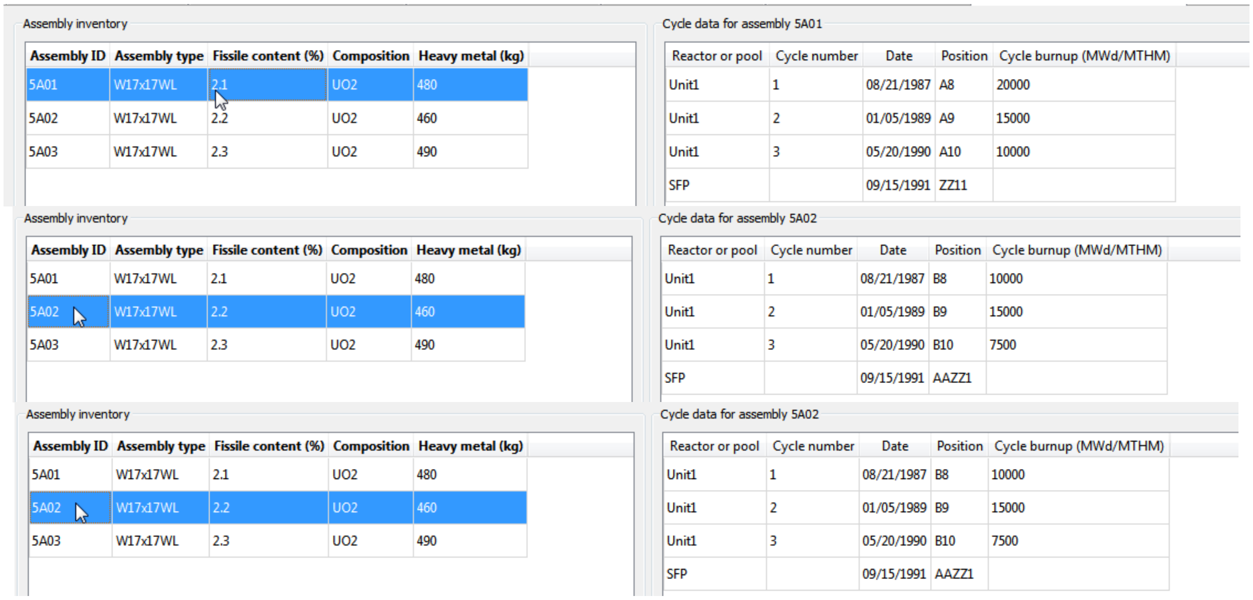


Fig. 28. Final view of the ORIGAMI Automator *Assembly data* navigation item.

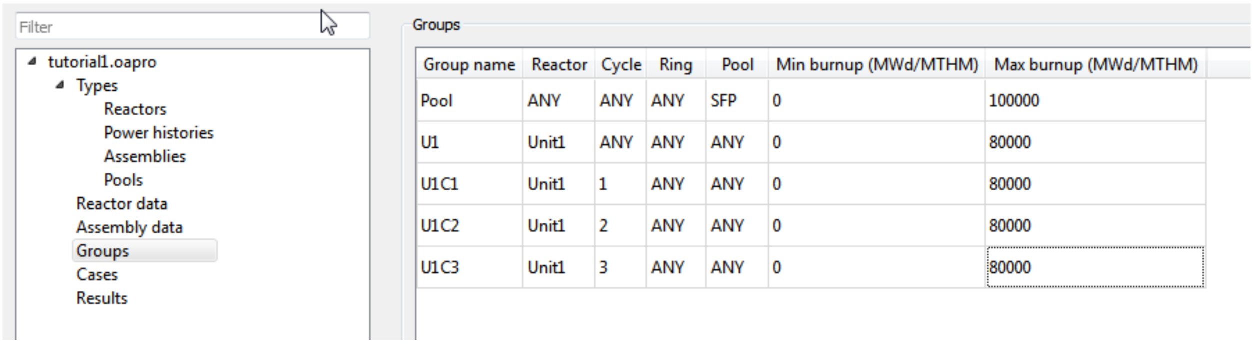
3.7 GROUPING NAVIGATION ITEM

The *Groups* navigation item allows you to specify groups to analyze and/or visualize in the future. Create groups for Unit1 Cycles 1, 2, and 3 and for the SFP. The data for the groups can be seen in Table 6. Note that outside of the start and end date for a cycle, all fuel assemblies are assumed to be in the SFP. In the *Groups* navigation item, right click in the *Groups* table, select *Add row,* and create the "Pool" group to be all assemblies in the SFP. Enter "U1" as the group name and "Unit1" without specifying cycles to create a group to track behavior over all cycles of the given core. Enter "U1C1" as the group name, "Unit1" as the *Reactor or pool,* and Cycle "1" as the *Cycle*. Repeat to create a "U1C2" group for "Unit2" Cycle 2 and "U1C3" group for "Unit1" Cycle 3. In this location, burnup ranges (0 to 80,000 MWd/MTU), which will include all assemblies, are specified. A smaller burnup range may be specified to filter a group.

Table 6. Group data defined in the *Groups* navigation item.

Group name	Reactor or pool	Cycle	Minimum burnup (MWd/MTHM)	Maximum burnup (MWd/MTHM)
Pool	SFP		0	100,000
U1	Unit1		0	80,000
U1C1	Unit1	1	0	80,000
U1C2	Unit1	2	0	80,000
U1C3	Unit1	3	0	80,000

When the *Groups* navigation item is complete, ORIGAMI Automator should look like Fig. 29. Note: *Ring* selection is not available from the GUI in this version.



Group name	Reactor	Cycle	Ring	Pool	Min burnup (MWd/MTHM)	Max burnup (MWd/MTHM)
Pool	ANY	ANY	ANY	SFP	0	100000
U1	Unit1	ANY	ANY	ANY	0	80000
U1C1	Unit1	1	ANY	ANY	0	80000
U1C2	Unit1	2	ANY	ANY	0	80000
U1C3	Unit1	3	ANY	ANY	0	80000

Fig. 29. Final view of the ORIGAMI Automator *Groups* navigation item.

3.8 CASES NAVIGATION ITEM

In the *Cases* navigation item under *Cases to evaluate*, right-click and select *Add row* and enter “Case1” for the *Case identifier*. In the *Case parameters* panel of the *Cases* navigation item, enter/verify the following information:

- **Date evaluated:** 10/18/1991
- **Number of processors:** 1
- **Irradiation template:** C:/SCALE-6.2/etc/Templates/OrigamiAutomator/assembly.tmpl
- **ORIGEN libraries directory:** C:/SCALE-6.2/data/arplibs
- **Calculation results directory:** C:/Users/ww5/Desktop/Tutorial1/Cases/Case1

Note, the above assumes standard Windows installation directories and that the "Tutorial1" folder was created on the desktop of user "ww5". Click the *Run* button to perform the desired calculations. The calculations can be canceled at any point by clicking *Cancel*. A message will be sent to the terminal window as calculations are completed. When the calculations are complete, the *Cases to evaluate* panel should look like Fig. 30. Note the console window that opens with Fulcrum shows the status of the calculation. Also, if *Run* is pressed a second time then a message explaining there is no need to rerun the calculation is displayed.

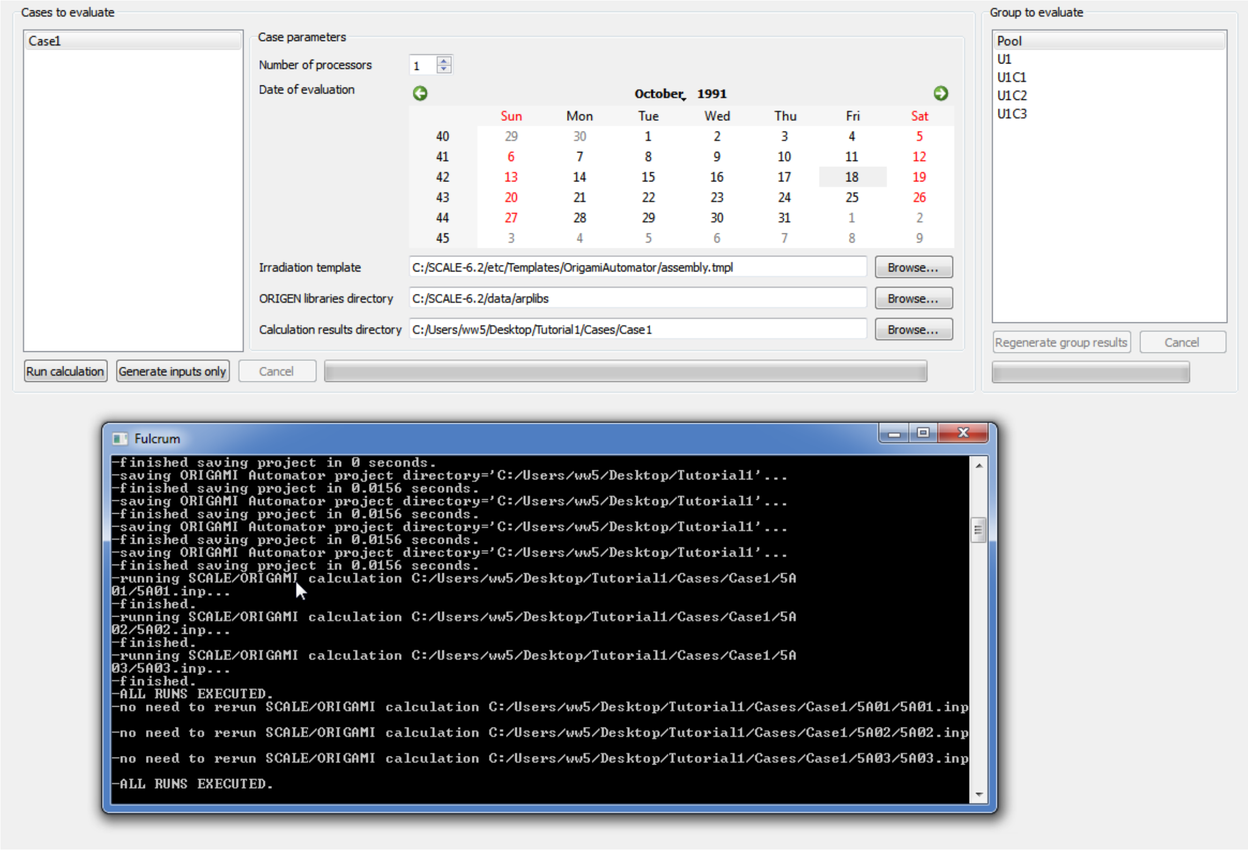


Fig. 30. Final view of the ORIGAMI Automator *Cases* navigation item.

Once all calculations are complete, the groups may be regenerated by clicking on a group name and then clicking on the button *Regenerate group results*, as shown in Fig. 31. Standard shortcuts may be used to highlight multiple groups, e.g. CNTL-A to select all groups.

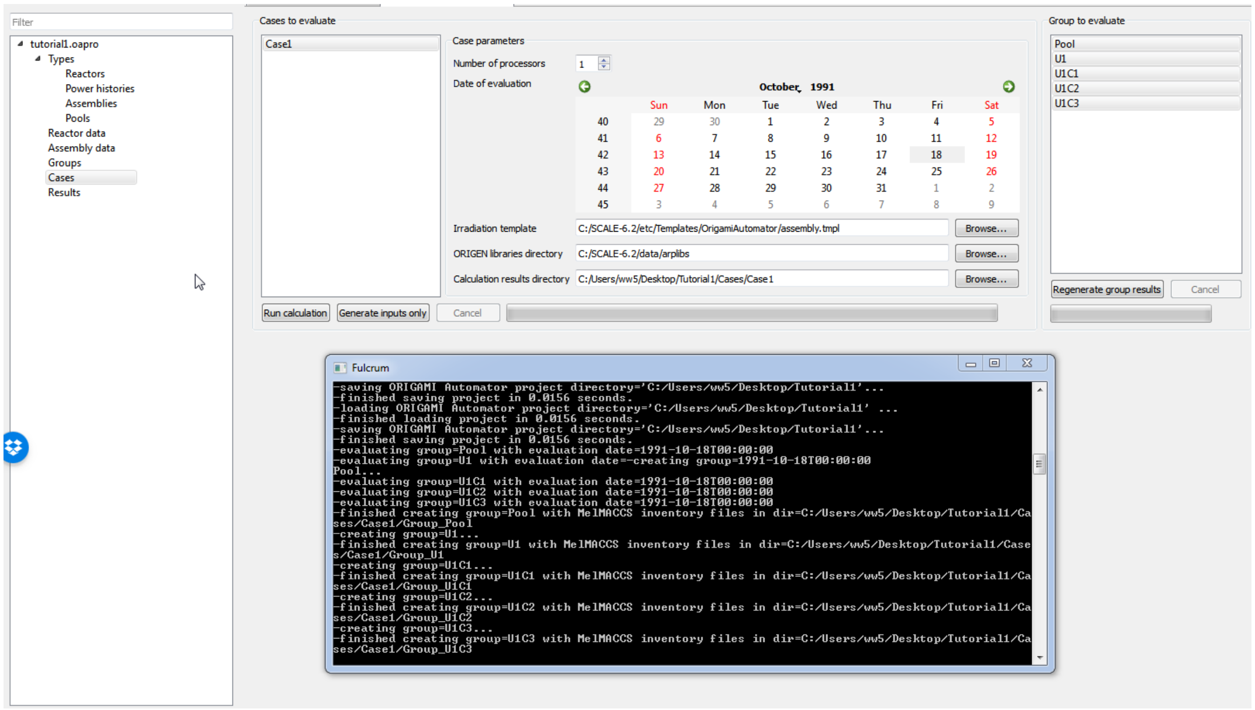


Fig. 31. Final view of the ORIGAMI Automator *Cases* navigation item after regenerating all groups.

3.9 RESULTS NAVIGATION ITEM

As each assembly calculation completes, the results for that particular assembly can be immediately viewed in the *Results* navigation item. Group results are only available once all calculations have been completed and the results have been regenerated for the group.

A navigation tree is in the top-left panel to select a given assembly's results to be displayed. Using the controls in the bottom-left portion of the window, the user can specify parameters that will be used by OPUS to down-select which nuclides' concentrations to view. For example, (1) select assembly *5A01* in the navigation tree, (2) select nuclides *u235*, *pu239*, *pu240*, *pu241*, and *pu242*, (3) plot units in *kilograms*, (4) sort units in *gram-atoms*, and (5) sort time units in *years*. Note that the selected nuclides must be separated by spaces and not commas. The *Plot* panel should match Fig. 32.

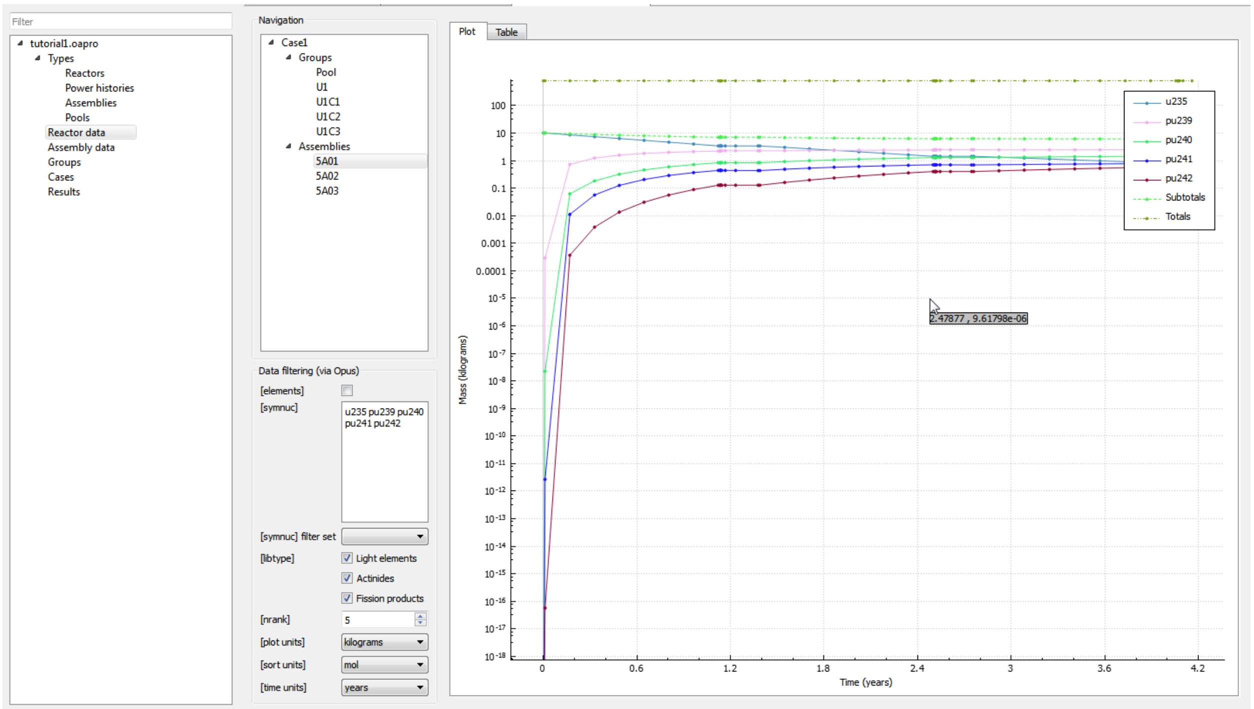


Fig. 32. Final view of the ORIGAMI Automator *Results* navigation item.

As another example, navigate to the group “U1,” which was Unit1 for all cycles, and plot the decay heat ("watts") by element. The *Plot* panel should match Fig. 33. The next objective is to plot the activity in the *U1* and *Pool* groups in Curies only for the fission products: nrank=5 for the *U1* group and nrank=15 for the *Pool* group. The *U1* core group should look like Fig. 34, with clear decay during the shutdown cooling after cycles 1 and 2. There is no decay after cycle 3 because all assemblies were moved to the *Pool* group, as shown in Fig. 35. The *Pool* group actually received its first assembly (5A03) at the end of cycle 2, and then it received the other two at the end of cycle 3. Because the target time was so soon after the end of the last cycle, the final decay period is very short (Fig. 35 at ~4.1 years).

Note: The assembly time axis begins at the time corresponding to the introduction of the fuel assembly and the group time axis begins at the time of the introduction of the first fuel assembly in the simulation. Both the group and assembly time axes end at the end of the specified evaluation period.

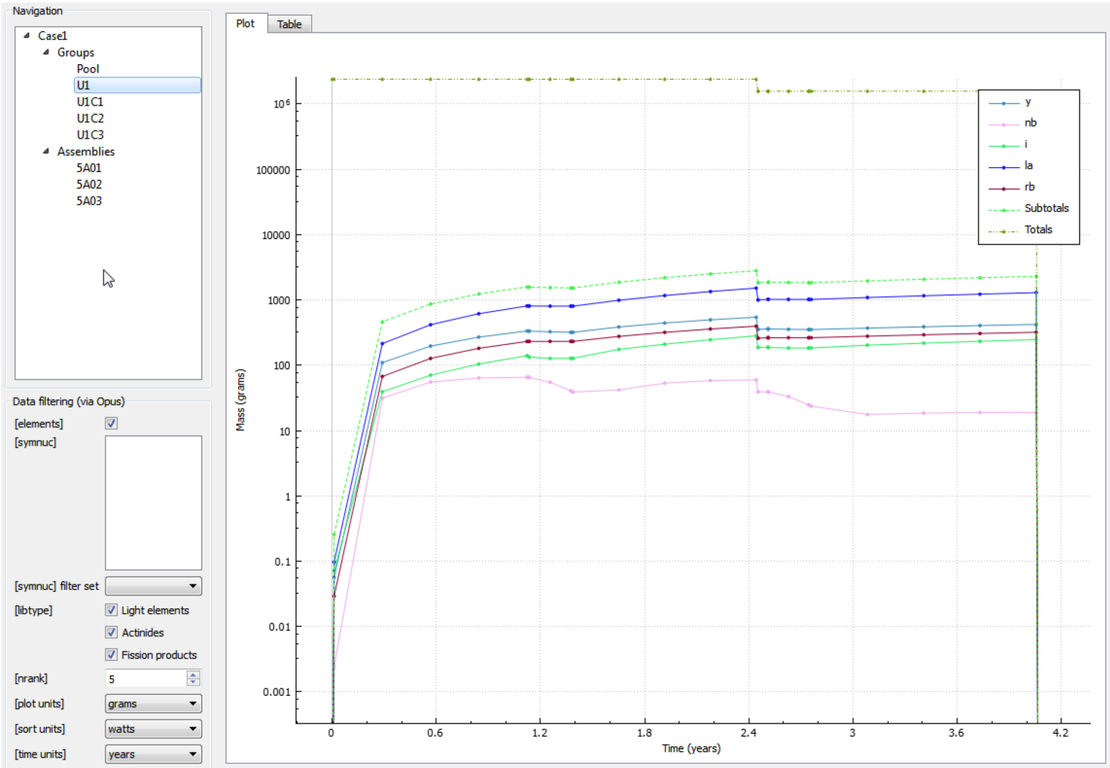


Fig. 33. Plot of decay heat by element for all cycles.

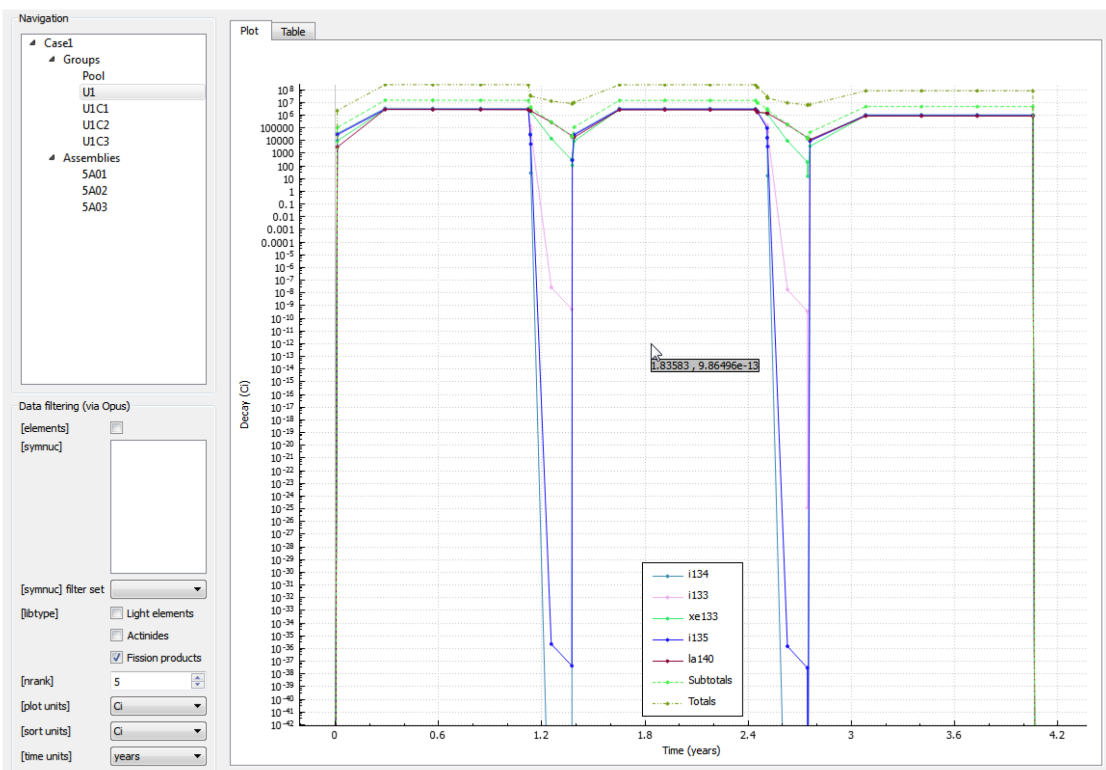


Fig. 34. Plot of activity (Ci) for fission products in *U1* group.

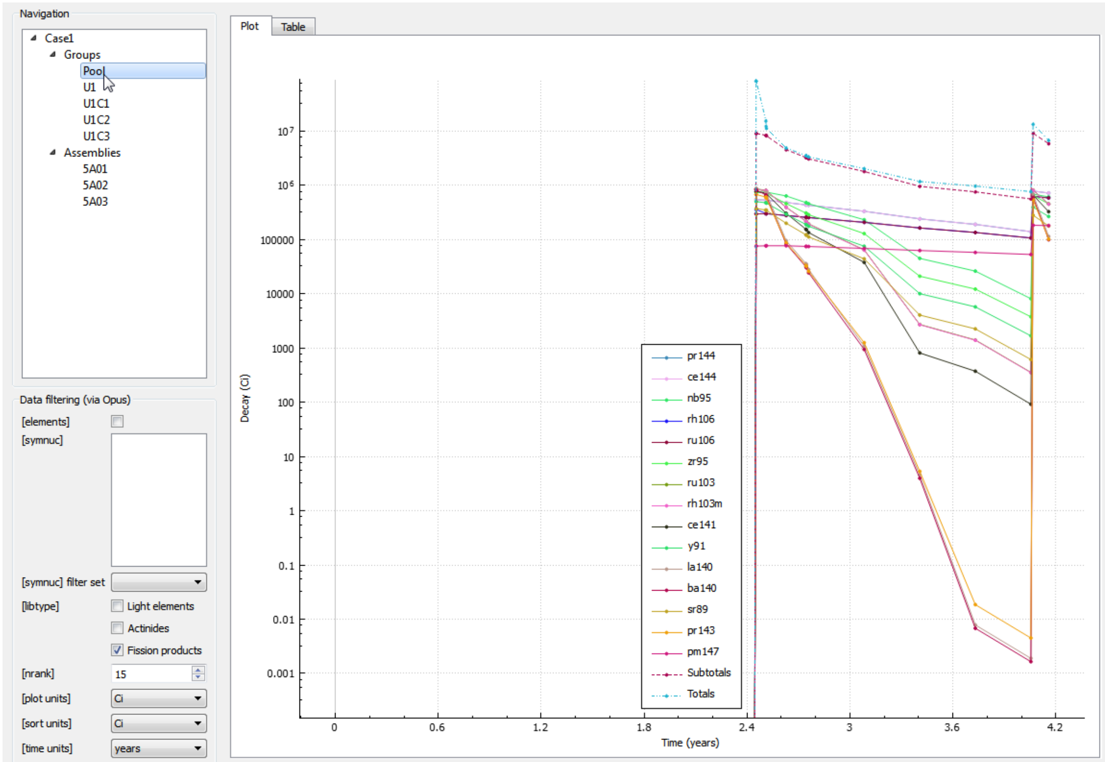


Fig. 35. Plot of activity (Ci) for fission products in *Pool* group.

4. CONCLUSIONS

ORIGAMI Automator provides an easy way for users to analyze SNF isotopic and elemental inventories, radiation source terms, and decay heat loads for operating nuclear power plants.

This primer describes the features and usage of ORIGAMI Automator tool, which is based around a GUI that allows the analyst to perform large-scale SFP analyses using ORIGAMI. The primer provides a user guide for the GUI and a step-by-step tutorial for a simplified scenario. In addition, appendices are included that document the file structures of the underlying data.

5. REFERENCES

1. B. T. Rearden and M. A. Jessee, Eds., *SCALE Code System*, ORNL/TM-2005/39, Version 6.2, Oak Ridge National Laboratory, Oak Ridge, Tennessee (2016). Available from Radiation Safety Information Computational Center as CCC-834.

2. J. M. Scaglione, K. Banerjee, K. R. Robb, and R. A. Lefebvre, "The Used Nuclear Fuel Storage Transportation & Disposal Analysis Resource and Data System," *Proceedings of the Institute of Nuclear Materials Management (INMM) – 55th Annual Meeting*, Atlanta Georgia, USA, July 20–24, 2014.

APPENDIX A. DEFINITION OF JSON DATA FILES

APPENDIX A. DEFINITION OF JSON DATA FILES

An ORIGAMI Automator project directory must be created for each site analysis. The directory structure created to analyze site “SiteAnalysisXYZ” is shown in Fig. A.1. Note that for an initial import of a project, ORIGAMI Automator will scan for any present JSON files and create “site.xml.”

```
SiteAnalysisXYZ
│ └── GenericReactor
├── Types
│ ├── Assembly
│ │ ├── W17x17L.json
│ │ └── W17x17V5.json
│ ├── Core
│ │ └── Westinghouse4Loop.json
│ ├── Pool
│ │ └── SFP.json
│ └──  PowerHistory
│   └──  constant.json
│   └──  coastdown.json
├── Design
│ ├── Assembly
│ │ ├── 1.json
│ │ ├── 2.json
│ │ ├── 3.json
│ │ ............
│ │ └── <N>.json
│ └──  Core
│   └──  1.json
│   └──  2.json
├── Operation
│ ├── Assembly
│ │ ├── 1.json
│ │ ├── 2.json
│ │ ├── 3.json
│ │ ├──............
│ │ └── <N>.json
│ └──  Core
│   ├── 1.json
│   └── 2.json
├── Groups
│ ├── U1.json
│ └── Pool.json
└── Cases
  └──  Case1
    └──  Options.json
```

Fig. A.1. Project directory structure for ORIGAMI Automator site analysis data.

The filenames in the “Types” directory are based on the actual names. For example, coastdown.json is the data for the coastdown power history, whereas the Design and Operation filenames use a simple indexing scheme: the first assembly is 1, the second is 2, and so on. This indexing scheme is chosen because the assembly labels are optional. The operating history defined in Operation/Assembly/1.json corresponds with the assembly defined in Design/Assembly/1.json. Indices are used instead of actual file names so that names can be changed without moving files.

In this appendix, Linux-type paths are used. A simple versioning scheme is used for the JSON data files so the purpose of a file can be recognized, and new content may be added, without losing backward compatibility. Table A.1 lists the currently defined JSON files, their special version identifiers, the relevant GUI panels, the tables containing their detailed definitions, and an illustrated example for each.

Table A.1. Summary of JSON file information in Appendix A

Path	Version	GUI navigation item	Definition	Example
/Types/Assembly	S1.v0	I.a (Sect. 2.2.1)	Table A.2	Fig. A.2
/Types/PowerHistory	S1.v0	I.b (Sect. 2.2.2)	Table A.3	Fig. A.3
/Types/Core	S1.v0	I.c (Sect. 2.2.3)	Table A.4	Fig. A.4
/Design/Core/	S1.v0	II (Sect. 2.2.5)	Table A.5	Fig. A.5
/Operation/Core/	S1.v0	II (Sect. 2.2.5)	Table A.6	Fig. A.6
/Design/Assembly	UOX1.v0	III (Sect. 2.2.6)	Table A.7	Fig. A.7
/Operation/Assembly	S1.v0	III (Sect. 2.2.6)	Table A.8	Fig. A.8

```
{
    "axialHeight" : [ 365.760 ],
    "axialLibraries" : [ "w17x17" ],
    "axialMassFraction" : [ 100.0 ]
}
```

Fig. A.2. Example of a /Types/Assembly JSON file.

```
{
    "colLabels" :
    [
        "R",
        "P",
        "N",
        "M",
        "L",
        "K",
        "J",
        "H",
        "G",
        "F",
        "E",
        "D",
        "C",
        "B",
        "A"
    ],
    "colsPerRow" : [ 7, 11, 13, 13, 15, 15, 15, 15, 15, 15, 15,
13, 13, 11, 7 ],
    "moderatorDensity" : 0.7110,
    "numAssemblies" : 193,
    "rowLabels" :
    [
        "1",
        "2",
        "3",
        "4",
        "5",
        "6",
        "7",
        "8",
        "9",
        "10",
        "11",
        "12",
        "13",
        "14",
        "15"
    ],
    "size" : 15,
    "thermalPower" : 3411.0
```


Fig. A.3. Example of a /Types/Core JSON file.

```
{
    "columns" : 100,
    "numAssemblies" : 10000,
    "rows" : 100
}
```

Fig. A.4. Example of a /Types/Pool JSON file.

```
{
    "powerFraction" : [ 1.0, 100.0, 1.0 ],
    "timeFraction" : [ 1.0, 98.0, 1.0 ]
}
```

Fig. A.5. Example of a /Types/PowerHistory JSON file.

```
{
    "/Types/Assembly" : "W17x17WL",
    "MTHM" : 0.480,
    "assemblyLabel" : "5A01",
    "enrichment" : 2.10
}
```

Fig. A.6. Example of a Design/Assembly JSON file.

```
{
    "/Types/Core" : "Westinghouse4Loop",
    "coreLabel" : "Unit1"
}
```

Fig. A.7. Example of a /Design/Core JSON file.

```
{
    "Locations" :
    [

        {
            "coreLabel" : "Unit1",
            "cycleLabel" : "1",
            "date" : "08/21/1987",
            "deltaBurnup" : 20000.0,
            "positionLabel" : "A8"
        },

        {
            "coreLabel" : "Unit1",
            "cycleLabel" : "2",
            "date" : "01/05/1989",
            "deltaBurnup" : 15000.0,
            "positionLabel" : "A9"
        },

        {
            "coreLabel" : "Unit1",
            "cycleLabel" : "3",
            "date" : "05/20/1990",
            "deltaBurnup" : 10000.0,
            "positionLabel" : "A10"
        },

        {
            "date" : "09/15/1991",
            "deltaBurnup" : 0.0,
            "poolLabel" : "SFP",
            "positionLabel" : "ZZ11"
        }
    ]
}
```

Fig. A.8. Example of an Operation/Assembly JSON file.

```
{
    "Cycles" :
    [

        {
            "/Types/PowerHistory" : "Three steps",
            "EFPD" : 410.0,
            "cycleLabel" : "1",
            "shutdownDate" : "10/08/1988",
            "startupDate" : "08/21/1987"
        },

        {
            "/Types/PowerHistory" : "Three steps",
            "EFPD" : 409.50,
            "cycleLabel" : "2",
            "shutdownDate" : "02/23/1990",
            "startupDate" : "01/05/1989"
        },

        {
            "/Types/PowerHistory" : "Three steps",
            "EFPD" : 480.0,
            "cycleLabel" : "3",
            "shutdownDate" : "09/15/1991",
            "startupDate" : "05/20/1990"
        }
    ],
    "PowerUprates" : []
}
```


Fig. A.9. Example of an Operation/Core JSON file.


```
{
    "/Design/Core" : null,
    "/Types/Pool" : "SFP",
    "cycleLabel" : null,
    "maxBurnup" : 100000.0,
    "minBurnup" : 0.0
}
```


Fig. A.10. Example of a Groups JSON file.

```
{
    "cpus" : 1,
    "date" : "10/18/1991",
    "librariesDirectory" : "/scale/data/arplibs",
    "resultsDirectory" : "/home/usr/Desktop/Tutorial1/Cases/Case1",
    "templateFile" : "/scale/etc/Templates/OrigamiAutomator/assembly.tmpl"
}
```

Fig. A.11. Example of a Cases/Case[n]/Options JSON file.

Table A.2. Content definition for /Types/Assembly JSON file (version S1.v0)

Parameter	Type	Required?	Default	Input validation
axialLibraries	array(string)	yes		
axialHeight	array(float)	no	null	>0
axialMassFraction	array(float)	no	null	>0 and <1

Table A.3. Content definition for /Types/PowerHistory JSON file (version S1.v0)

Parameter	Type	Required?	Default	Input validation
powerFraction	array(floats)	no	null	>0 and <1
timeFraction	array(floats)	no	null	>0

Table A.4. Content definition for /Types/Core JSON file (version S1.v0)

Parameter	Type	Required?	Default	Input validation
thermalPower	float	yes		>0
colsPerRow	array(int)	no	Null	>0
rowLabels	array(string)	no	null	
numAssemblies	int	yes	null	>0
colLabels	array(string)	no	null	
size	int	no	null	>0

Table A.5. Content definition for /Design/Core JSON file (version S1.v0)

Parameter	Type	Required?	Default	Input validation
coreLabel	string	no	sequential numbering	
/Types/Core	pointer	yes		exists

Table A.6. Content definition for /Operation/Core JSON file (version S1.v0)

Parameter	Type	Required?	Default	Input validation
PowerUprates	object(float(newPower), date(effectiveDate))	no		
Cycles	object(startupDate(date), shutdownDate(date), /Types/PowerHistory(pointer), EFPD(float), cycleLabel(string))	no		

Table A.7. Content definition for/Design/Assembly JSON file (version UOX1.v0)

Parameter	Type	Required?	Default	Input validation
enrichment	float	yes		>0 and <100
MTHM	float	yes		>0
/Types/Assembly	pointer	yes		exists
assemblyLabel	string	no	sequential numbering	

Table A.8. Content definition for /Operation/Assembly JSON file (version S1.v0)

Parameter	Type	Required?	Default	Input validation
Locations	array(core: … pool: …)	no		
core	object(			
	cycleLable (string),	yes		must be valid
	positionLabel(string),	no		must be valid
	coreLable(string),	yes		must be valid
	deltaBurnup(float))	yes		>0
pool	object (			
	date(date),	no	last cycle date	
	poolLabel(string),	yes		must be valid
	positionLabel(string))	no		must be valid

APPENDIX B. ORIGAMI INPUT TEMPLATES

APPENDIX B. ORIGAMI INPUT TEMPLATES

One of ORIGAMI Automator's tasks is to create thousands of ORIGAMI input files (one per assembly) with detailed cycle-by-cycle power histories. For maximum interoperability with the UNF-ST&DARDS package [2], the ORIGAMI input files are created in the same way: by combining a template and a JSON file, driven by the UNF-ST&DARDS TemplateEngine. A flow diagram is shown in Fig. B.1.

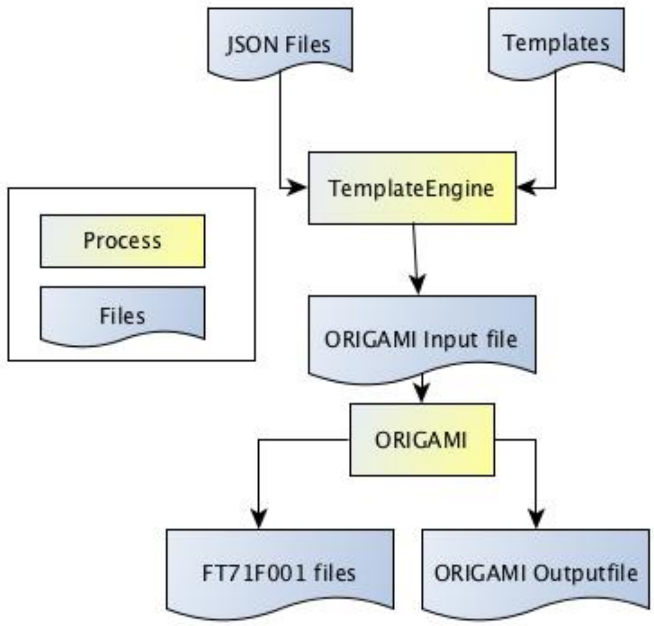


Fig. B.1. Flow chart showing the creation of ORIGAMI input files through the use of JSON files, templates, and the TemplateEngine.

The template system allows fine-grain control of the final ORIGAMI input file and calculation without having to provide access to every single parameter in the GUI. Some possible reasons to modify templates include

- modifying some defaults for convergence checks,
- adding extra header information in comments for quality control, and
- adding additional calls to post-processing modules following ORIGAMI.

The generic templates needed for the TemplateEngine consist of four primary types of data: raw text, attributes to be replaced, evaluations to be performed, and external templates to be imported. These are described below.

- **Raw text** is automatically inserted into the newly created input.

- **Attributes** are specified between the symbols "<>" and are replaced by the JSON value pairs.

- The **evaluations to be performed** include scalar (#eval) and iterative evaluations (#func). The scalar evaluations are used for calculations with the JSON values, whereas the iterative evaluations are a set of values being calculated similar to "for" loops in computer programming.

- **External templates** are imported into the input file. The templates to be imported into the main template are called with the “#import filename” expression. Alternatively, a template can be imported repeatedly using the loop expression “#repeat.”

For each case, ORIGAMI Automator tool processes all internal data, the target time of interest, and other parameters to produce one new type of JSON file describing each assembly. An example of this file is shown in Fig. B.2. Note that this is not one of the Automator’s JSON data files described in Appendix A, but a new file that is the result of evaluating the system at a certain time, taking into account runtime parameters.


```
{
    "assemblyLabel" : "5A01",
    "axial_node_count" : 1,
    "burn_data" :
    [

        {
            "ShutdownDate" : "10/08/1988",
            "StartupDate" : "08/21/1987",
            "burn_day" : 4.140,
            "coreLabel" : "Unit1",
            "cycleLabel" : "1",
            "down_day" : 0.0,
            "nlib" : 1,
            "power_w_per_g" : 0.4928502218318849
        },

        {
            "ShutdownDate" : "10/08/1988",
            "StartupDate" : "08/21/1987",
            "burn_day" : 405.720,
            "coreLabel" : "Unit1",
            "cycleLabel" : "1",
            "down_day" : 0.0,
            "nlib" : 7,
            "power_w_per_g" : 49.28502218318849
        }
    ],
    "burnup_profile[1]" : 1,
    "case_name" : "Case1",
    "evalDate" : "Fri Oct 18 00:00:00 1991",
    "initial_enrichment" : 2.10,
    "initial_uranium_MTHM" : 0.480,
    "libDir" : "C:/SCALE-6.2/data/arplibs",
    "libName" : "w17x17",
    "moderator_density[1]" : 0.7110
}
```


Fig B.2. JSON file created by the ORIGAMI Automator Template system.

The template to which the data in Fig. B.2 are applied is shown in Fig. B.3. The main template file can also import other templates to break down a complex input file. Examples of these are shown in Figs. B.4, B.5, and B.6.

```
' ------------------------------------------------------------------------------
' ASSEMBLY_ID:       <assemblyLabel>
' CASE_ID:           <case_name>
' EVALUATION_DATE:   <evalDate>
' LIBRARY_DIR:       <libDir>
' ------------------------------------------------------------------------------
' OPERATION_TIMETABLE:
' assm_id    unit cycle cycle_start   cycle_end power(W/g) burn(days) cool(days) incr_burnup(MWd/MTHM)
#import print_burn_data.tmpl using <burn_data>
' ------------------------------------------------------------------------------

=origami
title='ORIGAMI Automator calculation for Assembly <assemblyLabel> of Case <case_name> (<evalDate>)'

libs=[<libName>]

fuelcomp{ mix(1){ uox{ enrich=<initial_enrichment> dens=10.4 } } }

options[ % problem parameters
    mtu=<initial_uranium_MTHM>
    fdens=10.4
    stdcomp=yes
    small=no
    nburn=2
    ndecay=2
    temper=293.0
    relnorm=yes
    decayheat=yes
    interp=spline
    output=all
    ft71=all
]

% non fuel components
nonfuel=[ cr=3.366 mn=0.1525 fe=6.309 co=0.0302
  ni=2.366 zr=516.3 sn=8.412 ]

pz=[ % axial power profile array
#repeat burnup_profile_for_node.tmpl using node=1,<axial_node_count>
]

modz=[ % moderator density profile array
#repeat moderator_density_for_node.tmpl using node=1,<axial_node_count>
]

hist[
#import power_burn_down.tmpl using <burn_data>
]

end
```

Fig. B.3. ORIGAMI Automator “assembly.tmpl.”

```
<burnup_profile[<node:fmt=%.0f>]:fmt=%5.5f>
```

Fig. B.4. Example of imported template "burnup_profile_for_node.tmpl."

```
<moderator_density[<node:fmt=%.0f>]:fmt=%5.5f>
```

Fig. B.5. Example of imported template "moderator_density_for_node.tmpl."

```
cycle{
    power=<power_w_per_g:fmt=%.3f>
    burn=<burn_day:fmt=%.3f>
    nlib=<nlib:fmt=%.0f>
    down=<down_day:fmt=%.3f>  }
```

Fig. B.6. Example of imported template "power_burn_down.tmpl."

APPENDIX C. CSV-TO-JSON IMPORT UTILITY

APPENDIX C. CSV-TO-JSON IMPORT UTILITY

A small Perl utility has been created to assist in the import of large data sets in ORIGAMI Automator, as briefly discussed in Section 2.1.1 (shown in Fig. 3 as “import.pl”). The Perl utility reads one or more comma-separated value (CSV) files and produces the necessary JSON files.

The CSV files use a special block identifier that begins with a “/” to identify a particular section. The sections are labeled according to their “path” and “version” identifiers as shown in Table A.2. A “#” is treated as a comment. A skeleton of the CSV format is shown below in Fig. C.1.

```
/Types/PowerHistory:S1.v0
label , timeFraction, powerFraction

/Types/Assembly:S1.v0
label , axialHeight , axialMassFraction , axialLibrary

/Types/Core:S1.v0
label , size, numAssemblies, thermalPower , modDensity

/Types/Pool:S1.v0
label , numAssemblies

/Design/Assembly:UOX1.v0
label , enrichment , MTHM , Assembly.label

/Design/Core:S1.v0
label , Types/Core.label

/Design/Pool:S1.v0
label , Types/Pool.label

/Operation/Core.PowerUprates:S1.v0
Core.label , newPower , effectiveDate

/Operation/Core.Cycles:S1.v0
Core.label , cycleLabel , startupDate , shutdownDate , EFPD , PowerHistory.label

/Operation/Assembly:S1.v0
Assembly.label , pool , poolLabel , NULL , positionLabel , NULL , date
Assembly.label , core , coreLabel , cycleLabel , positionLabel , deltaBurnup , NULL
```

Fig. C.1. Skeleton of the CSV import format.

Because CSV is a flat format, complex, hierarchical JSON data such as the PowerUprates and Cycles sections of the Operation/Core JSON file must be broken into two separate CSV blocks. Note also that every entry in the CSV format is a scalar value. To enter arrays, such as for a power history, the power history label is repeated (see Fig. C.2).

```
/Types/PowerHistory:S1.v0
coastdown, 0.25, 0.50
coastdown, 0.25, 1.00
coastdown, 0.25, 1.00
coastdown, 0.25, 0.90
```

Fig. C.2. Example of a CSV power history.