

SHIELD-HIT12A

User's Guide

Version v1.1.1 - February 11, 2026

INSTITUTE FOR NUCLEAR RESEARCH RAS, MOSCOW, RUSSIA
AARHUS UNIVERSITY, DENMARK

Niels Bassler
Leszek Grzanka
Armin Lühr
David C. Hansen
Nikolai Sobolevsky

- *Поднимите мне веки!*

SHIELD-HIT12A is a Monte Carlo particle transport code for heavy ions, optimized for research in particle therapy. SHIELD-HIT12A is Copyright © 2010 – 2026 by

- Institute for Nuclear Research RAS, Moscow, Russia
<http://www.inr.ru/shield>
- Aarhus University, Aarhus, Denmark
<https://www.au.dk>

Contributing authors:

Niels Bassler^{1,2}, Leszek Grzanka³, Armin Lühr⁴, David C. Hansen⁵ Nikolai Sobolevsky^{6,7}.

¹Department of Physics, Stockholm University, Stockholm, Sweden

²Department of Experimental Clinical Oncology, Aarhus University Hospital, Aarhus, Denmark

³Proton Radiotherapy Group, Institute of Nuclear Physics PAN, Krakow, Poland

⁴OncoRay – National Center for Radiation Research in Oncology, Faculty of Medicine and University Hospital Carl GustavCarus, Technische Universität Dresden, Dresden, Germany

⁵Department of Oncology, Aarhus University Hospital, Aarhus, Denmark

⁶Institute for Nuclear Research RAS, Moscow, Russia

⁷Moscow Institute of Physics and Technology (MIPT), Dolgoprudny, Russia

SHIELD-HIT12A is the property of the copyright holders. For further inquiry, contact

Nikolai Sobolevsky <sobolevs@inr.ru>
Niels Bassler <bassler@clin.au.dk>

The development of SHIELD-HIT12A was supported by grants from the Danish Cancer Society and NovoNordisk Fonden.

This manual is based on SHIELD-HIT12A version v1.1.1 .

Contents

1	Introduction	1
1.1	SHIELD-HIT12A	1
1.1.1	Particle transportation	2
1.1.2	Nuclear physics	2
1.1.3	Atomic physics	3
1.1.4	Scoring of physical quantities	3
1.1.5	Other features	4
2	Quick start guide	5
2.1	Linux: Installation from binary distributions	5
2.2	Windows	5
2.3	Converter script <code>convertmc</code>	5
2.4	Setup a SHIELD-HIT12A run	6
2.5	Example	6
2.6	Analyzing output	9
3	Input files	11
3.1	Overview	11
3.2	<i>mat.dat</i> - Target medium	12
3.2.1	Loading external stopping power files	12
3.2.2	Voxelized structures	13
3.2.3	List of descriptors for <i>mat.dat</i>	14
3.2.4	Examples - Material initialization	15
3.3	<i>beam.dat</i> - Projectile, statistics, seeds...	17
3.3.1	List of descriptors for <i>beam.dat</i>	18
3.3.2	USECBEAM - External source file	21
3.3.3	USEPARLEV - External parameter file	23
3.3.4	USEBMOD - Beam modulator	23
3.3.5	Examples - Beam initialization	26
3.4	<i>geo.dat</i> - Geometry	27
3.4.1	Title	27
3.4.2	Bodies	27
3.4.3	Voxelized Body	28
3.4.4	Zones	29
3.4.5	Media	30
3.4.6	Examples - Geometry	30
3.5	<i>detect.dat</i> - Scoring	34
3.5.1	Geometry Section	35
3.5.2	Filter Section	37

3.5.3	Settings Section	40
3.5.4	Output Section	42
3.5.5	Complete Examples	48
3.6	<i>detect.dat</i> - Scoring (Old Style)	51
3.6.1	GEOMAP - Mapping the geometry	51
3.6.2	ZONE - Scoring by zone	52
3.6.3	CYL - Cylindrical scoring	53
3.6.4	MSH - Cartesian scoring	54
3.6.5	PLANE - Scoring by 2D plane	55
3.6.6	DZONE, DCYL, DMSH and DPLANE - Differential scoring	56
3.6.7	TRACE - Dump particle tree	57
3.6.8	VOXSCORE - Scoring within loaded CT-cube	58
3.6.9	Available detectors for scoring	59
3.6.10	Uncertainties	60
3.6.11	Examples - Scoring	61
4	Output files	63
4.1	Auxiliary output files	63
5	Auxiliary programs	65
5.1	Helper scripts	65
5.2	Format conversion	65
5.2.1	<i>convertmc</i> - Convert <i>.bdo</i> files to ASCII or similar	65
5.2.2	<i>shield2fluka</i> - Convert <i>geo.dat</i> to FLUKA format	65
5.2.3	<i>fluka2shield</i> - Convert FLUKA <i>.inp</i> files to SHIELD-HIT12A input files	65
5.2.4	<i>shield_dEdx</i> - Prepare an external stopping power table	66
6	Parallelization	67
6.1	Torque	67
6.2	LSF	68
6.3	Condor	68
A	User's reference tables and lists	71
A.1	Command line options and arguments	71
A.2	Units	72
A.3	Particle codes	73
A.4	Nuclear targets with neutron cross sections	74
A.5	Material ICRU_ID	76
A.6	Bodies	81
A.7	Input and output files.	84
A.8	PARLEV parameters	85
B	Additional background information	89
B.1	Optional antiproton annihilation corrections APCORR	89
B.2	History	90
C	The BDO Fileformat	93
C.1	Overview	93
C.2	Token Description	94
C.3	Compressed Format - <i>.bdz</i>	95

C.4 Tag IDs	95
C.5 Payload Element Type Description	97
C.6 Geometry and Detector Identifiers	97
C.7 Detector Identifiers	97
C.8 Unit IDs	98
C.9 Fileformat IDs	99
Bibliography	101

List of Tables

3.1	List of available geometry types.	35
3.2	Table of keywords for the available three different geometry types.	36
3.3	List of available filter rules.	37
3.4	List of particle IDs which are used for filter rules.	38
3.5	List of available filter operators.	38
3.6	List of available filter rules.	40
3.7	List of available media for scoring neutron Kerma	40
3.8	List of available keywords for the Output section.	42
3.9	List of available file formats	42
3.10	List of available detectors	43
3.11	Table of keywords for the available three different geometry types.	44
3.12	List of differential scorers.	45
A.1	Units	72
A.2	Particles	73
A.3	Nuclear target identifiers	74
A.4	ICRU materials	76
A.5	Input and output files	84
A.6	PARLEV parameters	87

Chapter 1

Introduction

The Monte Carlo particle transport code SHIELD-HIT¹ is designed to precisely simulate therapeutic beams of protons and ions in biological tissue relevant for ion beam cancer therapy. SHIELD-HIT (Heavy Ion Therapy) evolved from the common SHIELD code that models interactions of hadrons and atomic nuclei in complex extended targets in the energy range up to 1 TeV/nucleon.

SHIELD and SHIELD-HIT employ the same models to simulate nuclear interactions, which were developed at JINR, Dubna² and INR RAS (Moscow)³. The models are grouped together in the MSDM-generator (Multi Stage Dynamical Model) which allows to simulate a whole nuclear reaction in the exclusive approach. Contrary to the inclusive approach, every generated secondary particle is tracked and processed until it stops, decays, leaves the simulation universe or interacts destructively. Parameters of all secondaries including residual nuclei are retained, thereby fulfilling all conservation laws. Neutron transport below 14.5 MeV is simulated using the 28 group neutron data system ABBN [?] both in SHIELD and SHIELD-HIT.

SHIELD calculates ionization losses of charged hadrons and nuclear fragments according to the Bethe-Bloch equation. The heavy ion version SHIELD-HI contains also ATIMA stopping powers [?]. The "medical" version SHIELD-HIT includes various models and data sets to compute mean ionization loss, fluctuation of ionization losses and multiple Coulomb scattering. In addition, external stopping power tables (including MSTAR and ICRU tables) can be provided using the libdEdx [?] stopping power library.

A more elaborate history of the various releases of SHIELD-HIT is given in section B.2.

1.1 SHIELD-HIT12A

SHIELD-HIT12A was forked from SHIELD-HIT08 and its source code includes massive changes. The physics engine was tuned to new experimental nucleus-nucleus interaction data, many performance improvements were made and generally the entire code base was restructured, now eliminating the need for users to recompile code for typical usage applications. Several bug fixes from the original SHIELD-HIT branch are forwarded and included in SHIELD-HIT12A. Further development of the Physics engine and bug fixes from the original SHIELD-HIT branch enter continually in SHIELD-HIT12A and vice versa.

Characteristic features of SHIELD-HIT12A can be summarized in the following way:

¹Official homepage: <http://www.inr.ru/shield>

²<http://www.jinr.ru/>

³<http://www.inr.troitsk.ru/>

1.1.1 Particle transportation

- Transport of neutrons, pions, kaons, atomic nuclei, and a number of anti particles in energy range up to 1 TeV/nucleon for SHIELD-HIT. SHIELD-HIT12A is limited to 2 GeV/A. Lower energy cutoff is $E_{\text{cut}} = 25$ keV/nucleon except for atomic nuclei with $Z > 20$ in SHIELD-HIT12A where $E_{\text{cut}} = 1$ MeV/nucleon.
- Available initial beam configurations: Gaussian, flat square or flat circular in any direction. Beam divergence can be Gaussian or flat beam; focused or defocused beams can be specified.
- Geometric configuration of the target as an arbitrary combination of geometric bodies bounded by the second order surfaces (Combinatorial Geometry compatible) [?, ?].
- Arbitrary chemical and isotopic composition of materials in target zones can be defined based on the table of available isotopes shown in table A.3.
- Full memorization of the extra-nuclear cascade tree during simulation without any loss of physics information.
- Formation of neutrons ($E_n < 14.5$ MeV) as well as electrons/positrons and γ -rays, which were created during simulation of extra-nuclear cascade and π^0 decay. However, only neutrons are transported.
- The possibility to switch on/off various physics processes (energy straggling, multiple scattering, nuclear interactions) on user request.

1.1.2 Nuclear physics

- Simulation of inelastic hadron-nucleus and nucleus-nucleus interaction in exclusive approach using a Multi Stage Dynamical Model (MSDM-generator) [?]. The total and inelastic cross sections of the hadron-nucleus and nucleus-nucleus interaction are calculated according to [?, ?, ?] with modifications by [?, ?, ?]. These cross sections are used for sampling of the nuclear interaction path length as well as for choice of the interaction type (inelastic/elastic). The MSDM-generator describes all stages of the nuclear reaction in the exclusive approach. Current versions of known Russian nuclear models are interfaced in the MSDM-generator:
 - Fast, cascade stage of the nuclear reaction
 - * Intranuclear cascades is handled by the Dubna Cascade Model (DCM) [?]
 - * Independent quark-gluon string model (QGSM) [?, ?, ?]
 - * The coalescence model [?]
 - Precompound emission of nucleons and lightest nuclei [?]
 - Equilibrium de-excitation of the residual nucleus
 - * Fermi break-up of light nuclei [?]
 - * Evaporation/Fission competition [?, ?]
 - * Multifragmentation of highly excited nuclei (SMM) [?]
- Neutron transport ($E_n < 14.5$ MeV) on the basis of the 28-group neutron data ABBN [?]. A phase space file of these neutrons can be exported for use in other programs such as MCNP or MCNPX [?].

- Two- and three-particle modes of decay of pions and kaons.
- Neutron transport below 14.5 MeV is simulated by the original neutron transport code LOENT (Low Energy Neutron Transport) [?] using the 28 group neutron data system ABBN [?]. The LOENT code may be used both separately and as a part of the SHIELD code, since SHIELD and LOENT use the same geometry parser (Combinatorial Geometry) [?, ?].

The LOENT code uses the following tables from the neutron data system ABBN:

- st - total cross section
- sf - fission cross section (n,f)
- n - mean number of fission neutrons
- sc - capture cross section (n,c)
- sin - inelastic scattering cross section (n,n'), including the reaction (n,2n)
- se - elastic scattering cross section (n,n)
- m - mean cosine of the angle of the elastic scattering
- sin(g,g+k) - matrix of inter group transitions at the inelastic scattering.

The LOENT code gets neutrons from an external neutron source and follows them, one by one, until the end of the neutron trajectory. Multiplication of neutrons is possible via the reactions (n,2n) and (n,f). Each neutron has its statistical weight attached and a cumulative timer, which accumulates the time from the beginning of the neutron transport history. After transition of the neutron to the thermal group, its energy does not change in further collisions.

1.1.3 Atomic physics

- Ionization losses of charged hadrons and nuclear fragments according to the Bethe-Bloch equation and the Lindhard Scharff equation at low energies.
- Multiple Coulomb scattering simulated with Moliere's or a Gaussian model, similar to that implemented in Geant [?].
- Fluctuations of the ionization energy loss (energy straggling) simulated by a fast implementation (article in preparation) of Vavilov's model [?, ?] or Gaussian distribution.
- Possibility to load external mass stopping power tables to override internal ones. A script interfacing to libdEdx [?] is available.

1.1.4 Scoring of physical quantities

A generic scoring system was developed—that is specified by the user—which provides:

- Detectors for a wide range of physical quantities, which can be made sensitive to specific particles or particle groups.
- Arbitrary scoring in Cartesian or cylindrical scoring grids.
- Alternatively, detectors can be assigned to zones that constitute the target geometry.
- Routines to merge results from multiple runs from parallel processing system.

In addition to the generic scoring system, the legacy scoring system is retained, providing:

- Scoring of the production rates of radioisotopes in the entire system.
- Track Length Estimation (TLE) of differential (differential in energy) fluences and energy fluence of secondary particles and nuclear fragments averaged over each geometric zone of the target.
- Scoring of contributions to the energy deposition from various types and from different generations of particles and nuclear fragments separately.

However, the post-processing scripts available for this scoring system cannot merge the ASCII output from parallel runs.

1.1.5 Other features

- Reading and processing of beam source and ripple filter files (see sections 3.3.2 and 3.3.4, respectively).
- Support for parallelization (see chapter 6).

Chapter 2

Quick start guide

2.1 Linux: Installation from binary distributions

SHIELD-HIT12A should come in a tarball of the form: *shield_hit12a_vXXX.tar.gz*, where *XXX* refers to the release version. Untar the file:

```
$ tar xvfz shield_hit12a_vXXX.tar.gz
$ cd shield_hit12a_vXXX
$ sudo make install
```

which will copy the binaries to */usr/local/bin*. This should work on Ubuntu type systems. If you wish to install to other directories, you may have to edit the *Makefile* accordingly. On non-sudo type systems (such as Debian or RedHat based Linux systems) one can become root with **su** instead of **sudo** and the last line then reads:

```
$ su
# make install
```

2.2 Windows

SHIELD-HIT12A was tested on Windows 10 and may in principle also run on older Windows systems. SHIELD-HIT12A is provided as a zip file for 64 bit systems. Once you unzipped it, you will find a ready-to-run *shieldhit.exe* file.

2.3 Converter script convertmc

The output from SHIELD-HIT12A is in a binary format, which is not human readable. In order to convert the output to something sensible, such as ASCII data, or perhaps a plot, you must use the *convertmc* script, which is provided by the *pymchelper* package [?] you can find on <https://github.com/DataMedSci/pymchelper>. On Linux systems, make sure you have the python package manager *pip* installed (Ubuntu and debian: **apt-get install pip** Fedora/RedHat: **dnf install pip** or **yum install pip**). Then simply run

```
$ pip install pymchelper
```

to install it as a user. For system wide installation you can also run the same command as root user. If you have already installed *pymchelper* you may want to upgrade it:

```
$ pip install pymchelper --upgrade
```

2.4 Setup a SHIELD-HIT12A run

You can run one of the examples found in base directory, simply by entering:

```
$ shieldhit examples\simple
```

or change into the directory and run `shieldhit` without arguments.

Windows users can open a shell (Run - "Enter a command" - cmd) and change to the directory of where the *shieldhit.exe* file is found. To run the example, simply type:

```
$ shieldhit examples\simple
```

or simply drop the *shieldhit.exe* file into the folder you want to run, and double click on *shieldhit.exe*.

2.5 Example

Here is a quick description of the most important files in the example *examples/simple* supplied with the distribution:

geo.dat

The file *geo.dat* describes the simulated geometry. Here we want to simulate a target which is a cylinder where the center of its base is located in origin (0,0,0) cm. The height of the cylinder is described by a vector along the Z axis, and is here set to 30 cm. Finally the cylinder radius is set to 10 cm. The cylinder is surrounded by a universe consisting of vacuum which does not interact with the particle. Finally that vacuum is surrounded by a black hole medium. Any particle hitting this medium will be disintegrated, and not tracked any further.

All this information is formatted in "cards". Cards are lines which are no longer than 80 characters long—a legacy from last centuries punch card systems. Only ASCII characters are allowed; as a white character only space is allowed while tabs are not.

Each card has a number of arguments, which are data being passed to the code. The position and lengths of the arguments varies, thus one should must make sure that the exact positions are used.

Lines beginning with * are ignored, and can be used for comments. So, in the case of the simulation described earlier, our file looks like this:

```
*---><---><-----><----->
      0      0          C12 200 MeV/A, H2O 30 cm cylinder, r=10, 1 zone
*---><---><-----><-----><-----><-----><-----><----->
RCC      1          0.0      0.0      0.0      0.0      0.0      30.0
          10.0
RCC      2          0.0      0.0      -5.0      0.0      0.0      35.0
          15.0
RCC      3          0.0      0.0      -10.0      0.0      0.0      40.0
          20.0
END
*.-><--->..<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->
```

```

001          +1
002          +2      -1
003          +3      -2
END
  1      2      3
  1 1000      0

```

The first line, is a comment line, which is ignored. It is inserted here, to better see where the fields are which are interpreted by SHIELD-HIT12A. The following line specifies a title of the geometry setup. The title is preceded by two integers which are described later in section 3.4. After another comment line, the next two lines describe the geometry of the aforementioned target cylinder. The RCC card selects a body (here a Right Circular Cylinder). The body is assigned to a number 1. The x,y,z coordinates of the base follow, along with a vector which spans to the top of the cylinder. Since the cylinder is following the Z axis, all entries are zero, except for the height of the cylinder, i.e. the z-component of the vector. The following card is a continuation card of the RCC card, and holds only one value, namely the radius of the cylinder. All values are in cm. A list of default units in SHIELD-HIT12A is provided in section A.2.

Afterwards two more cylinders follow, each expanded by an additional 10 cm in all three directions.

The geometry section, which only contains one body here, must be terminated by the END card.

The lines

```

*.<-->---->..<---->OR<---->OR<---->OR<---->OR<---->OR<---->OR<---->OR<---->
001          +1
002          +2      -1
003          +3      -2

```

assigns the bodies we just specified to zones. For the first body (our target cylinder) the body and the zone are identical. The surrounding vacuum is represented by the second cylinder, but subtracted the inner cylinder, since this is already described by zone 001. Similarly zone 003 consists of the largest body (number 3), subtracted the body number 2. There is no reason to subtract body number 1 here, as it is already entirely contained in body number 2. The point is that a zone can be any Boolean composition of previously defined bodies, which is described in detail later in section 3.4. Any point inside the black hole must be assigned to a zone, and only one zone.

The section describing the zones must also be terminated with the END card.

Finally, two lines follow: the first specifies a list of zones, and the second assigns a material number to that zone. In this example, each line consists only of three numbers since we have three zones. In the second line we tell which materials each zone consists of. The numbers 0 and 1000 are predefined, any other number will be specified in the *mat.dat* file. 0 is black hole, 1000 is vacuum and the inner most cylinder is made of material with the id# 1.

mat.dat

The file *mat.dat* specifies the medium of the zone we just specified in *geo.dat*.

We define medium number 1 to be water. The most simple way to do this is by writing:

```

MEDIUM 1
ICRU 276
END

```

The first line tells SHIELD-HIT12A that we now are going to assign **MEDIUM** number one. The second line says that we want to use a default ICRU material with material number 276. The numbers of predefined materials (as given by ICRU) are listed in table A.4 in section A.5 with 276 being water. Each **MEDIUM** section must be terminated with an **END** line for SHIELD-HIT12A to initialize the material. The specification materials, other than those listed in table A.4, will be explained in section 3.2.

beam.dat

For *beam.dat* we should initially only worry about a few lines. Fairly at the top, **JPART0** specifies the number of the primary particle according to table A.2.

```
JPART0                25      ! Incident particle type
HIPROJ                12.0      6.0      ! A and Z of heavy ion
```

Here we request a beam of heavy ions (**JPART0** = 25), which is specified further with the **HIPROJ** card. By stating $A = 12$ and $Z = 6$ we select carbon-12 ions to be our primary particle.

The energy is specified with the **TMAX0** card, here set to 391.0 MeV/nucleon.

```
TMAX0                391.0      0.0      ! Incident energy; (MeV/nuc1)
```

The statistics, i.e. the amount of primary particles to be simulated is specified in the **NSTAT** line:

```
NSTAT                2000      1000      ! NSTAT, Step of saving
```

With no further input we get a default pencil beam with $\sigma_x = 0.0$ $\sigma_y = 0.0$ cm which starts at origin (0,0,0).

detect.dat

Finally, we can specify detectors to score a variety of quantities. The file *detect.dat* may be in a "new" or in an "old format", as described later in sections 3.5 and 3.6. In this example, we will use the old format, and want to score a depth dose-curve along the water target.

We can setup a cylinder, following the Z axis, located inside the water target. It does not need to match any geometry specified in *geo.dat*. Note, this file requires fixed format.

```
*----0---><----1---><----2---><----3---><----4---><----5---><----6--->
CYL                0.0      0.0      0.0      10.0      7.0      30.0
                  1        1        300      -1      ENERGY      ex_cyl
```

The first card specifies cylindrical scoring, starting from point (0,0,0), and with the radius 10 cm, covering 2π (by specifying "7"¹) of the circumference of the cylinder, and with a height of 30 cm along the Z axis. Currently, the **CYL** scoring cannot be rotated, and is always placed parallel to the X -axis. In the next line the first three arguments say that we want 1 bin for the radius, 1 bin for the angle spanning the cylinder² and 300 bins along the height. The result will then be a one-dimensional file, where the scored quantity is listed in steps of 30.0 cm / 300 bins (i.e. 1 mm). The next three arguments tell that we want the **ENERGY** to be scored of the particle -1 which means the energy deposited by all particles crossing the bins. **ex_cyl** is the name of the output file.

¹Any number larger than 2π means we do not subdivide the cylinder along its radial lines, i.e. we do not cut the cylinder in pieces of pie.

²I.e. scoring along the entire circumference and not into pie pieces.

Alternatively we could also specify a Cartesian scoring mesh. The two next lines specifies a mesh with one bin from -5 to 5 cm along the X and Y axis, and then 300 bins from 0 cm to 30 cm along Z . The result is saved in the file *ex_zmsh*.

```
MSH          -5.0      -5.0      0.0      5.0      5.0      30.0
              1         1        300      -1      ENERGY  ex_zmsh
```

We can here also score along another axis if we want. The next example scores 100 bins from -5 cm to 5 cm along the Y axis.

```
MSH          -1.0      -5.0      10.0      1.0      5.0      12.0
              1        100         1      -1      ENERGY  ex_ymsh
```

or if we want to see a 100 x 300 pixel 2-D map of the energy deposited along the beam path in the target:

```
MSH          -5.0      -5.0      0.0      5.0      5.0      30.0
              1        100        300      -1      ENERGY  ex_yzmsh
```

2.6 Analyzing output

After SHIELD-HIT12A terminated the run you find several new files in your directory. Here the results from the four detectors, defined in *detect.dat*, are stored in the files *ex_zmsg.bdo*, *ex_ymsh.bdo*, *ex_yzmsh.bdo* and *ex_cyl.bdo*. These files are in binary form and the script `convertmc` can be used to convert them to ASCII format. Stepping into the directory and applying,

```
$ convertmc txt ex_zmsh.bdo
```

will produce a human-readable output file *ex_zmsh.txt*, which can be plotted. You can also directly produce a plot in .png format

```
$ convertmc image ex_zmsh.bdo
```

`convertmc` is also capable generating plots directly, please see the documentation of `pymchelper` at <https://pymchelper.readthedocs.io/>

Uncertainties are briefly covered in section 3.6.10.

Chapter 3

Input files

3.1 Overview

The SHIELD-HIT12A transport code works with at least three input files.

mat.dat – chemical composition of materials in target zones.

beam.dat – several parameters, explained below (like seed, projectile, statistics, etc.).

geo.dat – geometry of the target using combinatorial geometry (CG), similar to FLUKA.

Optionally, the user can include an additional file:

detect.dat – for simple scoring of physical quantities in independent geometries. There are currently two available scoring systems, which will be selected depending on what format the file *detect.dat* has.

These four files are described in more detail in the sections to follow.

As extension of the file *beam.dat* up to three optional ASCII formatted files (with user-specified names) can be supplied that specify: a ripple filter, a beam source file, and parameters for nuclear interaction models. Examples for these three files are shown adjacent to the description of file *beam.dat* in 3.3. If chosen by the user, SHIELD-HIT12A may also look for external ASCII files containing stopping power data as described in section 3.2.

All input files must be grouped in the same folder. Its location can be specified by providing a command line argument to the SHIELD-HIT12A executable. All output files are placed into the same directory. The SHIELD-HIT12A executable itself can be run from anywhere if the exact path to the directory with the input files is given. If the executable is run from the directory containing the input files the specification of the path can be omitted.

3.2 *mat.dat* - Target medium

The *mat.dat* file defines the materials that are used in the simulation. It consists of cards (lines starting with a descriptor followed by possible arguments) and comment lines. Any line starting with a `*` or a `<` or a blank line are regarded as a comment, which is ignored and may appear at any place. The ordering of the cards has to obey certain rules. Some cards can be omitted, and SHIELD-HIT12A will assign the default values given in the descriptor list below. It is recommended to specify all parameters explicitly since default values may change in future releases of SHIELD-HIT12A. While the ordering of the arguments must be as specified for the respective card, they can be positioned in any column (free format).

Each parseable line starts with a descriptor, which specifies what parameters are to be described. After this descriptor, none, one or two arguments follow, depending on the type of the descriptor. The descriptor and each argument must be separated with at least one space or tab. The rest of the line is not parsed. It can be used for comments if it is separated from the last argument with at least a space or tab. Only the first 128 characters are read; any characters beyond will be ignored.

Each material specification starts with the **MEDIUM** card. A medium can consist of a single element or of a compound. A compound again can be a chemical compound, or simply a mix of elements and/or isotopes, e.g.:

- Pure medium - element with one fixed Z (e.g. O, Fe, Cu, etc.).
- Chemical composition, e.g. water H_2O .
- Isotope mixture, if isotopes of the same Z have different neutron properties at low energies, e.g. mixture of ^{235}U and ^{238}U .
- a chemical compound containing isotopes, e.g. ^6LiF .

The maximal total number of different media **NUMMED** is limited to 100, excluding internal and outer vacuum which are predefined as medium numbers 1000 and 0, respectively. Furthermore, if CT-images are imported using the **VOX** card, the available medium slots are reduced depending on the Hounsfield unit to medium segmentation algorithm applied. This may be up to 40 media, leaving 60 media for the user.

The user can select one of the predefined media listed in the ICRU material table in section A.5. Alternatively, the user can define a **MEDIUM** by specifying the chemical elements or isotopes, which can be found in this medium. The constituent elements are limited to a total number of 14. A chemical element is described by the variable **NUCLID**. Usually, **NUCLID**= Z selects the element from the periodic table corresponding to the atomic number Z ($1 < Z < 100$). Further isotopes, which differ in the number of neutrons, are predefined with values of **NUCLID** > 100 and are listed in table A.3. If an isotope does not appear in the list, no low energy neutron cross section data are included in SHIELD-HIT12A. In this case SHIELD-HIT12A will not start the simulation, if the neutron transport cutoff energy (**NEUTRLCUT** in *beam.dat* see section 3.3) is set below 14.5 MeV.

Note that the corresponding neutron cross sections are either for natural isotope mixtures or for a specific isotope.

3.2.1 Loading external stopping power files

The user can provide external files containing mass stopping power data by specifying the **LOADDEDX** card with an appropriate value from the material list in table A.5. Each file contains data for the lightest 18 ions from $^1\text{H}^+$ (proton) to $^{40}\text{Ar}^{18+}$ interacting with one specific target material.

These external files are expected to be ASCII formatted, space or tab separated columns, and should follow the ICRU 73 [?] energy grid (53 energy nodes from 25 keV/u up to 1 GeV/u). SHIELD-HIT12A expects the grid to contain the first 18 ions, i.e. from protons ($Z=1$) to Argon ions ($Z=18$). The mass stopping power data must be in units of MeV cm²/g. Comment lines starting with *** are allowed anywhere in the file, but no line (either comment or data) may be longer than 512 characters. External stopping power files may contain data different from ICRU, e.g. ATIMA [?] data, but the energy grid must be compatible to ICRU 73. The libdEdx [?, ?] computer library can serve as a source for stopping power data. A script *shield_dEdx*, described in section 5.2.4, can generate these tables for most ICRU materials using libdEdx.

Examples of external stopping power files are: *Water.txt*, *Air.txt*, *A-150.txt*, *Kapton.txt*, which are supplied along with the SHIELD-HIT12A distribution. In table A.5 the naming convention of the external stopping power files is listed.

3.2.2 Voxelized structures

A CT cube loaded with the **VOX** card in *geo.dat* must be assigned to a medium number, which is specified as a special voxel type medium in *mat.dat*. This is done using the **VOXMED** card in *mat.dat*. Specifying this card, will prepare a list 24 materials, based on a paper by Schneider et al. [?], which is valid for Hounsfield units between -1000 to +1600. Additionally, a density scale is created, and assigned to each voxel, depending on the Hounsfield units.

$$\rho(\text{HU}) = \begin{cases} 1.03091 + 1.0297 \cdot 0.001\text{HU} & : \text{HU} < -98 \\ 1.0018 + 0.893 \cdot 0.001\text{HU} & : -98 \leq \text{HU} < 14 \\ 1.03 & : 14 \leq \text{HU} < 23 \\ 1.0003 + 1.169 \cdot 0.001\text{HU} & : 23 \leq \text{HU} < 100 \\ 1.017 + 0.592 \cdot 0.001\text{HU} & : 100 \leq \text{HU} \end{cases}$$

with ρ in units of [g/cm³]. Materials with densities below 0.5 will automatically have the **STATE** set to 1 (gaseous), and 2 (condensed) otherwise.

User-specified Hounsfield conversion tables may be implemented upon request to the developers.

3.2.3 List of descriptors for *mat.dat*

The recognized descriptors are listed alphabetically, and not by relevance.

AMASS : 1 argument. Optional card which can override the atomic mass A of an element when specified *after* a **NUCLID** card.

END : No argument. Mandatory card that terminates the description of a medium that was started by the **MEDIUM** card.

ICRU : 1 argument. Optional card which will select a material from the ICRU list in section A.5. This card may not be used together with the **NUCLID** card for a **MEDIUM**, as it may lead to unpredictable behaviour.

IVALUE : 1 argument. Optional card that can override the default mean excitation energy (I -value) of the element that was specified with the preceding **NUCLID** card. Units are eV.

LOADDEDX : 1 optional argument. Optional card which will trigger **SHIELD-HIT12A** to look for an external stopping power file following the naming scheme stated in the table in section A.5. The argument must be the ICRU number of the material to be loaded. However, if used after the **ICRU** card, the ICRU number is already known, and the argument can be omitted. The external stopping power files should contain mass stopping power data for the medium for ions ranging from $^1\text{H}^+$ (proton) to $^{40}\text{Ar}^{18+}$. See section 3.2.1 for a description of the format.

MEDIUM : 1 argument which specifies the medium number. The numbering must be sequential, starting at 1 for the first medium which is specified.

NUCLID : 2 arguments. This card adds a single element to the medium. First argument is the **NUCLID** number specifying the element or isotope from table A.3. You can also use a Z value that is missing in this list but there are no neutron cross section data attached to it. Second argument is the relative stoichiometric fraction, which will automatically be normalized by **SHIELD-HIT12A**. Up to 13 elements can be defined per **MEDIUM**. **SHIELD-HIT12A** will refuse to start, if there are no neutron cross sections for the requested element, and the lower neutron energy cut is set below 14.5 MeV.

RHO : 1 argument which specifies the density of the medium in g/cm^3 . This card is mandatory in conjunction with the **NUCLID** card. It should be stated after the **MEDIUM** card and before the first **NUCLID** card. In the case that the material is setup using the **ICRU** card, **RHO** will override the default value.

STATE : 1 arguments which specifies the state of the medium. Appropriate mean excitation energies (I -values) are selected for each element in a compound, following the ICRU49 guidelines [?]. This card is mandatory in conjunction with the use of a **NUCLID** card. It should be stated after the **MEDIUM** card and before the first **NUCLID** card.

SHIELD-HIT12A discriminates between a chemical compound and just a mix of elements. If **STATE** 0, then **SHIELD-HIT12A** uses I -values for elements. If **STATE** 1 or **STATE** 2 **SHIELD-HIT12A** uses I -values for compounds in gas or liquid/condensed phase, respectively. In the case that the material is setup using the **ICRU** card, **STATE** will override the default state of the **ICRU** material.

VOXMED : Specifies that this medium will be a voxel structure, prepares a material database consisting of 24 materials following Schneider *et al.* [?]. Density will be scaled to Hounsfield units, as described in section 3.2.2.

3.2.4 Examples - Material initialization

Example 1: Specifying five materials

An example of *mat.dat* with five materials is given below

1. Water, ICRU default setup. External stopping powers will be loaded from *Water.txt*.
2. Air, ICRU default setup.
3. PMMA (also known as Perspex or Lucite), is defined. It is a compound in solid state. In this example we assume slightly different parameters, i.e. we override the default density with a density of 1.20 g/cm³. Also the default I-value of hydrogen is changed to 21.9 eV. (This example is merely intended for illustration, and is not motivated to mimic a real situation.)
4. Alanine, ICRU default setup, but overriding the density.
5. Lithium-6 Fluoride (LiF), which is composed of Lithium-6 isotopes which has a very high thermal neutron cross section. External stopping power tables are loaded from *LiF.txt*.

```
* ----- WATER -----
MEDIUM      1
ICRU         276
LOADDEDX
END
* ----- AIR -----
MEDIUM      2
ICRU         104
END
* ----- PMMA -----
MEDIUM      3
STATE       2
RHO         1.20
NUCLID      1      8
IVALUE      21.9      ! Override I-value of hydrogen
NUCLID      6      5
NUCLID      8      2
END
* ----- NPL ALANINE -----
MEDIUM      4
ICRU         105
RHO         1.23      ! Override density, since NPL alanine is less dense
END
* ----- Li6-Fluoride -----
MEDIUM      5
STATE       2
RHO         2.635
NUCLID      105     1
NUCLID      9       1
LOADDEDX    185      ! Better stopping powers from external lib.
END
```

Example 2: Voxelized CT structure

Another example, where a voxelized structure is assigned to medium number 2:

```
* ----- WATER -----  
MEDIUM 1  
ICRU 276  
END  
* ----- VOXEL STRUCTURE-----  
MEDIUM 2  
VOXMED  
END
```

3.3 *beam.dat* - Projectile, statistics, seeds...

The *beam.dat* file characterizes the particle beam transport by defining: the primary beam properties; the required statistics and random seed; details of the physics engine; and additional options.

The format is free, meaning that the arguments can be given in any columns, but they must be ordered as in the description of the respective card. After the last argument, any text string for comments are allowed. Cards can be arbitrarily ordered. Any card (with exception of HIPROJ, see below) can be omitted, and SHIELD-HIT12A will then use default values. The default values are given in the descriptor list below, but may change unnoticed in future releases of SHIELD-HIT12A. Therefore it is recommended to avoid using default parameters, if they are important to the simulation.

In *beam.dat* any line starting with a `*` or a `<` is regarded as a comment and is ignored. Blank lines will be ignored too.

Each parsable line starts with a descriptor that specifies what parameters are to be described. After this descriptor, one to three arguments follow, depending on the type of the descriptor. The descriptor and each argument must be separated with at least one space or tab. The rest of the line is not parsed, and can be used for comments. However it is important that the comments are separated too from the last argument with a space or tab. Only the first 128 characters are read, any characters beyond will be ignored.

3.3.1 List of descriptors for *beam.dat*

The recognized descriptors are listed alphabetically, and not by relevance.

APCORR : Optional card. 1 argument that switches the antiparticle physics correction on and off. This will affect the annihilation physics of antiprotons, π^- and K^- . The argument can be either 0 or 1. 0 for using the default settings – calculating the capture probability based on the Fermi-Teller Z-law [?], and using the default antiproton cross sections. 1 for scaling the antiproton cross section by a factor of 1.08 and using a more physical correct theory for the capture probability. See section B.1.

If this card is not specified, **APCORR** is set to zero.

BEAMDIR : Optional card. 2 arguments that specify the beam direction: the polar and azimuth angles θ and ϕ , both in degrees, as explained in figure 3.1.

Initially, the beam divergence and shape is setup assuming transport along Z axis. No matter where the card is specified, the actual rotation of the beam into the requested beam direction will always take place as a last step, before transportation begins. If this card is not specified, then beam transport along the Z axis is assumed.

If this card is not specified, θ and ϕ are set to zero. That is, by default the beam points in Z direction.

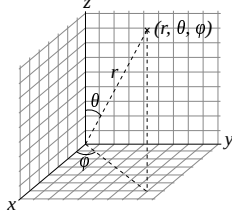


Figure 3.1: Coordinate system as used by SHIELD-HIT12A for describing the beam direction. θ is the angle relative to the Z axis. ϕ is the angle of the orthogonal projection of the direction onto XY plane measured from the X axis.

BEAMDIV : Optional card. 3 arguments that specify the divergence and focal point of the beam. First two arguments define the divergence half-angles as Gaussian sigmas in X and Y direction assuming beam transport in Z direction. (A subsequent rotation of the beam can be specified with the **BEAMDIR** card.) The beam divergence must be in mrad. Negative values will generate a flat distribution instead. The distributions are weighted along the polar angle, so requesting a flat divergence distribution will result in a flat beam spot.

The third argument sets the distance from the beam source position to the focal point k in cm at which the undisturbed beam (i.e. transport in vacuum) has the width and shape specified by the **BEAMSIGMA** card. A positive value $k > 0$ describes a defocused beam with the focal point upstream the beam at a distance k , while a negative value $k < 0$ results in a focused beam with the focal point downstream, relative to the source position of the beam. If $k = 0$ the beam has its focus coinciding with the beam source position.

The beam model follows equation (25) in reference [?], which gives a beam with a slant emittance ellipse. The same reference describes the Gaussian case only, however k affects any beam shape in SHIELD-HIT12A.

In case of a beam divergence larger than 2π , i.e. > 6283.2 mrad, then an isotropic distribution is assumed. Here, random sources are simulated located on a sphere with a radius given by the third argument. Each initialized particle will have a random direction. The second argument has no effect.

If this card is not specified, all values are set by default to zero.

BEAMPOS : Optional card. 3 arguments that specify the start position of the beam in (x,y,z) coordinates, respectively. All units are in cm.

By default the beam starts at the origin (0,0,0).

BEAMSAD : (UNDER DEVELOPMENT, DO NOT USE) Optional card. 1 or 2 arguments which specify the source-to-axis distance for scanning beam lines. SAD is the distance from the last bending dipole to the isocenter position. If one value is given, this will be used for both X and Y deflection. If two values are given, then these will be used for X and Y respectively. This parameter is intended to be used when a list of spot positions are given in an external spot file (see **USECBEAM**). The given spot positions listed in the **USECBEAM** file are assumed to be at the isocenter, and the SAD is then used as the distance from the corresponding bending magnet to the isocenter position. The **BEAMSAD** will then project these spot positions to the user-given to the position where the beam is started (typically at the beam nozzle), as specified with the **BEAMPOS** card.

Default: SADx and SADy set to infinity (= a parallel beam).

BEAMSIGMA : Optional card. 2 arguments that specify the lateral extension of a Gaussian-shaped beam by the sigmas in cm along X and Y, respectively. If **both** values are negative then a flat square distribution is generated, where the two arguments represent *half the width* of the sides. If σ_x is larger than 0 **and** σ_y is smaller than 0, then a flat circular distribution is generated with radius $\text{abs}(\sigma_y)$. The σ_x value is unused here, but it must be larger than 0.0. If the focus is specified in the optional **BEAMDIV** card then the beam size and shape are obtained at the focal point.

Default values are zero sigma in X and Y direction.

BMODMC : Optional card. 1 argument that specifies the mode of the beam modulator simulation: 0 for modulus beam modulator version retaining the spatial period information of the beam modulator (such as in ripple filters); 1 for Monte Carlo sampling of beam modulator material thickness—useful for simulating range modulator wheels or when generating beam kernel files (e.g. *.ddd* and *.spc* files for TRiP98).

Default value is 0.

BMODTRANS : Optional card. 1 argument that specifies the interpretation of data in the input files loaded with the **USEBMOD** card: 0 lists of material thicknesses (default), 1 lists of vacuum thicknesses.

Default value is 0.

DELTAE : Optional card. 1 argument that specifies the relative mean energy loss per transportation step, that is a finite fraction smaller than unity. This value times the particle energy at a given step determines the absolute energy loss.

Default is 0.03. (Meaning 3%.)

DEMIN : Optional card. 1 argument that specifies the minimum allowed energy loss per transportation step in MeV/n.

Some models are only valid if the absolute magnitude of energy loss (as calculated by **DELTAE**) is larger than a minimal value, such as Moliere multiple scattering.

Default value is 0.025 MeV/n.

EMTRANS : Future switch for photon/electron/positron transportation. Not implemented, default is 0 (off).

EXTSPEC : Specifies if external source file to be read with **USECBEAM** is an array of monoenergetic beams (**EXTSPEC** 0) or a binned spectrum (**EXTSPEC** 1). If a binned spectrum, then the energies are assigned to the lower energy of that bin. The last bin marks the endpoint of the energy spectrum, and its weighting value is thus ignored.

This card is optional, and default is 0 (no spectrum). The card has only effect if **USECBEAM** is called *after* **EXTSPEC**.

HIPROJ : If **JPART0=25** then the of this card 2 arguments specifies the number of nucleons A and the charge Z of the beam particles. Z has to be larger than 2, since nucleons with $Z = 1$ and $Z = 2$ have their own particle ID **JPART0**, as listed in table A.2.

If this card is omitted, then default values are $Z = 6$ and $A = 12$.

JPART0 : Optional card. 1 argument that specifies the primary particle ID, see table A.2. If **JPART0=25**, then the **HIPROJ** card can be called as well. If **HIPROJ** is not called, then carbon-12 is assumed by default.

If **JPART0** is not specified, then the default particle 2 (protons) are assumed.

MAKELN : Optional card. 1 argument that invokes phase-space output of all secondary neutrons with energies below 14.5 MeV which are created within the target: 1 output is written to *for028*, see section 4.1, 0 no output of secondary neutrons.

Default value is 0.

MSCAT : Optional card. 1 argument that switches between the available types of multiple scattering: 1 for Gaussian- and 2 for Moliere-type multiple scattering.

Moliere multiple scattering (2) is activated by default.

NEUTRFAST : Optional card. Toggles fast neutron transport for energies above 14.5 MeV. If set to 0, no neutrons above 14.5 MeV are transported.

Default is 1, that is fast neutron transport is activated.

NEUTRLCUT : Optional card. Lower energy cut-off value for neutron transport in MeV. Must be lower than 14.5 MeV but larger or equal 0.0 MeV. (This was previously called **OLN** in older versions of **SHIELD-HIT**.)

Default value is 0.0. In this case, neutrons are transported until they are absorbed or exit the simulation universe.

NSTAT : Optional card. 2 arguments that specify the requested total number of primary particles to be simulated and the number of transported primaries after which an intermediate save to the scorers will be invoked, respectively.

Default values are 10000 primary particles simulated, and intermediate saving after 5000 transported primary particles.

Setting the first argument of to -1 will cause an infinite amount of particles to be transported (or until integer overflow). This is useful in combination with the `-time` comandline option.

Setting the second argument to 0 or a negative number will disable saving the results, until either `-time` or the requested total number of primary particles have been simulated.

NUCRE : Optional card. 1 argument that switches nuclear reactions on or off: 1 with nuclear reactions; 0 all nuclear reactions are turned off. (This card was called **INUCRE** in earlier versions of SHIELD-HIT.)

Default value is 1, i.e. nuclear reactions are activated.

RNDSEED : Optional card. 1 argument that specifies the random number seed. Using the same seed for an identical simulation should yield the same results.

Default value is 89736501.

STRAGG : Optional card. 1 argument that switches between the available models for energy straggling: 1 for Gaussian- and 2 for Vavilov-type energy straggling; 0 for no straggling.

Vavilov straggling (2) is activated by default.

TCUTO : Optional card. 2 arguments that specify the upper and lower possible initial energy interval for the primary beam, respectively. This is useful for truncating the initial energy distribution given by **TMAX0**, representing the lowest and highest energy available from an accelerator. Depending on **TMAX0**, this can be either energy [MeV] or [MeV/nucleon], or momentum [MeV/c]. (Negative values are here treated as positive values, whether it is momentum or energy depends on the sign of **TMAX0**.) Default values are 0.0 and 1.0e32.

TMAX0 : Optional card. 2 arguments that specify the initial energy of the primary particle in MeV/nucleon and the energy spread in MeV/nucleon, expressed as one standard deviation of a Gaussian distribution (1σ).¹ If a negative value is given, then it is treated as momentum [MeV/c].

Default values are 250 MeV/nucleon and zero energy spread.

USEBMOD : Optional card. 2 arguments that specify the zone number (integer) and an external beam modulator file. See section 3.3.4. No file name is specified by default, and by default an invalid zone number is specified.

USECBEAM : Optional card. 1 argument that specifies the filename of an optional external beam source file. See section 3.3.2.

No file name is specified by default

USEPARLEV : Optional card. 1 argument that specifies the file name of an external file. The file may contain model parameter values that override the default **PARLEV** model parameters. See section 3.3.3.

No file name is specified by default

3.3.2 USECBEAM - External source file

If a file name is specified, e.g. *sobp.dat*, a projectile will be sampled from the file. Beam settings specified in the input file *beam.dat* such as **BEAMSIGMA** and **TMAX0** will be overridden. The weighting is typically taken as the particle number for the individual beam components (or spots). The total particle number is passed on in the header of the *.bdo* in case a **VOXSCORE** card is specified in *detect.dat*.

Example of a typical beam file, which was generated from a hadron therapy carbon ion control file:

¹Note that [MeV/nucleon] is different from [MeV/amu].

```
*ENERGY(GEV) X(CM) Y(CM) FWHM(cm) WEIGHT
0.270550 -1.00 1.00 0.48 9.3700e+06
0.270550 -0.90 1.00 0.48 9.3700e+06
0.270550 -0.80 1.00 0.48 9.3700e+06
0.270550 -0.70 1.00 0.48 9.3700e+06
0.270550 -0.60 1.00 0.48 9.3700e+06
0.270550 -0.50 1.00 0.48 9.3700e+06
0.270550 -0.40 1.00 0.48 9.3700e+06
0.270550 -0.30 1.00 0.48 9.3700e+06
0.270550 -0.20 1.00 0.48 9.3700e+06
0.270550 -0.10 1.00 0.48 9.3700e+06
(...)
```

Notice that the energy is in GeV to maintain compatibility with other Monte Carlo codes. Lines starting with * are ignored and can be used for comments.

If 6 columns are specified, the following format is assumed:

```
*ENERGY(GEV) X(CM) Y(CM) FWHMx(cm) FWHMy(cm) WEIGHT
0.270550 -1.00 1.00 0.48 0.44 9.3700e+06
0.270550 -0.90 1.00 0.48 0.44 9.3700e+06
0.270550 -0.80 1.00 0.48 0.44 9.3700e+06
0.270550 -0.70 1.00 0.48 0.44 9.3700e+06
0.270550 -0.60 1.00 0.48 0.44 9.3700e+06
0.270550 -0.50 1.00 0.48 0.44 9.3700e+06
0.270550 -0.40 1.00 0.48 0.44 9.3700e+06
0.270550 -0.30 1.00 0.48 0.44 9.3700e+06
0.270550 -0.20 1.00 0.48 0.44 9.3700e+06
0.270550 -0.10 1.00 0.48 0.44 9.3700e+06
(...)
```

The FWHM parameter is the Gaussian FWHM (and not sigma), contrary to the BEAM-SIGMA card² However, if FWHM is negative (both x and y in the 2-d case), then a box with similar side length is generated. If only one of the two values are negative in the 2-d case, then a flat circular distribution is assumed, with FWHMy as the circular radius. FWHMx is currently ignored, but may be used for ellipsoid beamlets in future.

If 7 columns are specified, the following format is assumed:

```
*ENERGY(GEV) SigmaTO(GEV) X(CM) Y(CM) FWHMx(cm) FWHMy(cm) WEIGHT
0.270550 1.1e-03 -1.00 1.00 0.48 0.44 9.3700e+06
0.270550 1.1e-03 -0.90 1.00 0.48 0.44 9.3700e+06
0.270550 1.1e-03 -0.80 1.00 0.48 0.44 9.3700e+06
0.270550 1.1e-03 -0.70 1.00 0.48 0.44 9.3700e+06
0.270550 1.1e-03 -0.60 1.00 0.48 0.44 9.3700e+06
0.270550 1.1e-03 -0.50 1.00 0.48 0.44 9.3700e+06
0.270550 1.1e-03 -0.40 1.00 0.48 0.44 9.3700e+06
0.270550 1.1e-03 -0.30 1.00 0.48 0.44 9.3700e+06
0.270550 1.1e-03 -0.20 1.00 0.48 0.44 9.3700e+06
```

²Thereby compatibility to FLUKA can be retained.

```
0.270550  1.1e-03   -0.10    1.00    0.48    0.44    9.3700e+06
0.270550  1.1e-03    0.00    1.00    0.48    0.44    9.3700e+06
(...)
```

where SigmaT0 marks the energy spread of the primary energy assuming a Gaussian distribution. The energy spread is 1σ standard deviation in GeV/nucleon. This will override the second argument specified with the TMAX0 card in *beam.dat*.

3.3.3 USEPARLEV - External parameter file

The predefined values of the PARLEV parameters can be redefined by those given in the optional PARLEV input file thereby overwriting the carefully selected and benchmarked default values. Further information on PARLEV parameters can be found in A.8 and the default values are listed in Table A.6.

It is recommended to use the default values. They have been tested against available data. Most likely you want to use the default values to remove the USEPARLEV card in the *beam.dat* file entirely.

Example of typical PARLEV input file with two columns. The first column specifies the ID of a PARLEV parameter while the second specifies its new value (either space or tab separated).

```
* ID Value
33 1.0
34 1.0
39 1.20
40 1.20
```

Lines starting with * are ignored and can be used for comments.

3.3.4 USEBMOD - Beam modulator

A beam modulator is a special geometry, which transports a particle hitting it by a certain amount in the Z-direction, following a user specified transportation table. The amount of transportation can be sampled randomly from that table, or be linked to a certain position along X and/or Y axis.

The beam modulator geometry is useful for simulating complex geometries such as

- Ripple filters [?] (RiFi).
- Ridge filters
- Range modulator wheels

A beam modulator can be inserted into the beam at an arbitrary position. This works by specifying a zone in *geo.dat*, and attaching a user generated, external modulator file to that particular zone using the USEBMOD command in *beam.dat*.

The beam modulator file is structured as a series of bins, holding a certain thickness. The first line contains the start position of the first bin, and the subsequent lines contain the end position of the bin and its respective thickness.

Shown below are the first lines of a typical user generated (1D) displacement map:

```

*Y coordinate   Z displacement
-0.797654000000
-0.797653867871      0.299843308865
-0.797583286279      0.296589497459
-0.790976849238      0.292032749859
-0.790878035008      0.28747741389
-0.784271597967      0.28292066629
-0.771129305477      0.277060982496
-0.771072840203      0.274457933371
-0.764466403162      0.269901185771
-0.751366459627      0.26599378882
-0.744760022586      0.26143704122
-0.738167701863      0.257531055901
(... many more data points... )

```

SHIELD-HIT12A assumes by default that the beam modulator file lists the thickness of the ripple filter as a function of position. If a displacement in vacuum is stated then the **BMODTRANS** card must be set in *beam.dat*.

Line breaks and lines starting with * are ignored and can be used for comments.

The beam modulator can either be one or two dimensional. If the beam modulator is one dimensional, the periodic structure changes along the Y axis only. The structure is repeated over the entire beam modulator on the y-position of the incoming particle. Alternatively, the structure can be sampled by a random number using the **BMODMC** card in *beam.dat*. This way the spatial structure is lost, but pencil beams hitting the same spot on the modulator will still be modulated over the entire structure, where the Y-bins are used as weights instead. This is useful for generating TRiP98 depth dose kernels, which require a full modulated pencil beam.

The data must be either space or tab delimited and the maximum number of steps is set to 200. The size of the beam modulator is specified in the *geo.dat* file. In the example shown below, the beam modulator is represented by the zone number 3. Successful application of the ripple filter will be stated at run time when the first particle hits the ripple filter surface.

Important: The thicknesses or displacements of the external file should not exceed the zone dimensions along the z-axis, as this may lead to undefined behaviour of the code.

Example: *geo.dat* with ripple filter (body and zone number 3).

```

      0      0      C-12 on water with RiFi  21.10.2011
RPP    1    -100.0    100.0    -100.0    100.0      0.0    100.0
RPP    2    -150.0    150.0    -150.0    150.0    -150.0    150.0
RPP    3      -8.0      8.0      -8.0      8.0     -1.300     -1.0
RPP    4    -200.0    200.0    -200.0    200.0    -200.0    200.0
END
TAR    1      +1
AIR    2      +2    -1    -3
RIF    3      +3
OUT    4      +4    -2
END
      1      2      3      4
      1      2      3      0

```

The *beam.dat* links the zone number 3 to a certain beam modulator file, by adding the line

```

USERIFI      3    rifi.dat

```

Additional information about the implementation of the beam modulator as a ripple filter can be found in [?] and [?].

Two-dimensional beam modulators can be specified as well. SHIELD-HIT12A detects this automatically if the specified file contains three columns of data instead of two. The supplied file should be a mesh of data, describing a full period of the ripple filter. Either the X coordinate can increment fastest followed by the Y position, or vice versa, yet the data must be ordered in increasing order along X and Y. The third column gives the thickness in Z direction. Again, X and Y denote the end-positions of the bins, except for the first line which defined the starting positions of the first bin. Line breaks and lines starting with * are ignored. The maximum number of allowed lines is hard coded to be 200x200 steps (i.e. 40.000 data points). The table below shows an excerpt of a 2D ripple filter file example:

```
* 2-D ripple filter example.
* X coordinate [cm]   Y coordinate [cm]   Z displacement [cm]
0.0000   0.0000
0.0000   0.0030   0.0000
0.0000   0.0060   0.0000
0.0000   0.0090   0.0000
0.0000   0.0120   0.0000
0.0000   0.0150   0.0058
0.0000   0.0180   0.0139
(...many lines skipped here...)
0.0060   0.1350   0.0278
0.0060   0.1380   0.0175
0.0060   0.1410   0.0082
0.0060   0.1440   0.0000
0.0060   0.1470   0.0000
0.0060   0.1500   0.0000

0.0090   0.0000   0.0000
0.0090   0.0030   0.0000
(...many lines skipped here...)
0.1500   0.1470   0.0000
0.1500   0.1500   0.0000
```

3.3.5 Examples - Beam initialization

Example 1: A SOBP carbon ion beam with ripple filter

Example of typical *beam.dat* file:

```
*          Input file beam.dat for the SHIELD-HIT Tansport Code
RNDSEED          89736501      ! Random seed
JPARTO           25          ! Incident particle type
HIPROJ           12.0         6.0      ! Carbon ions
TMAXO            400.0         0.0      ! Incident energy; (MeV/nucl)
BEAMSIGMA        2.0          2.0      ! SigmaX, SigmaY at focus point
BEAMPOS          0.0          0.0      -1.50 ! Beam XYZ start pos
NSTAT            20000        5000     ! NSTAT, Step of saving
NEUTRLCUT        0.0          ! Cutoff for neutron transport
MAKELN           0           ! 1 - Make neutron phase space file
DELTAE           0.020       ! Delta E (relative share ~0.1)
DEMIN            0.030       ! Minimum Energy step 0.025 (MeV/n)
STRAGG           2           ! Straggling: 0-Off 1-Gauss, 2-Vavilov
MSCAT            2           ! Mult. scatt 0-Off 1-Gauss, 2-Moliere
NUCRE            1           ! Nucl.Reac. switcher: 0-Off, 1-On
USEBMOD          3   rifi.dat  ! Zone# and file name for beam modulator
USECBEAM         sobp.dat    ! Filename of beam sourcefile
```


3.4 *geo.dat* - Geometry

The SHIELD-HIT code uses known Combinatorial Geometry (CG), similar to the Monte Carlo particle transport codes MORSE, FLUKA.

The geometry is defined by a number of so called *bodies* which are primitive objects such as boxes and cylinders, which by Boolean logic are combined to *zones*. For instance a square slab with a round hole can be constructed using two primitives, i.e. a box subtracted with a cylinder. Similarly a hollow box can be described by a larger box subtracted with a smaller box inside it.

The zones must then be assigned to a material identifier, which is an integer defined by the user in the material definition file *mat.dat*, see section 3.2. Every point in the simulated universe must be assigned to a zone, else errors may occur. So, if a hollow box is constructed, one must remember also to assign the void inside the box to a material code (e.g. internal vacuum 1000). One should also understand that a single body cannot directly be assigned to a material, but must be assigned to a zone first, even if the zone just consists of a single body.

The definition of the bodies, zones and the material assignments are set in the *geo.dat* file, in exactly that order.

Anywhere within *geo.dat* lines starting with *** are skipped.

3.4.1 Title

The first line of *geo.dat* contains two integers JDBG1, JDBG2 and a string of characters describing the title of the geometry. The format of the title card is:

- 5 columns (1-5): integer JDBG1, see below
- 5 columns (6-10): integer JDBG2, see below
- 10 columns (11-20): unused
- 60 columns (21-80): any title of the geometry

The first integer JDBG1 selects whether the file *for017* containing the geometry debugging information should be kept (0) or deleted (1) after the geometry parser was initialized. The second integer JDBG2 describes the lower cutoff value of transportation step size in powers of 10, i.e. $10^{-|\text{JDBG2}|}$. If set to zero, the minimum step size is set to 10^{-4} cm. If any of the numbers are omitted, they are interpreted as (0). The fields for both numbers are five columns wide.

3.4.2 Bodies

A list of bodies describing the geometry follows the title card explained in the section before. A body is a primitive object with a given geometrical configuration. The available primitive objects are listed section A.6. A “top level” body (which is mandatory in the MC code FLUKA) can optionally be added which contains all other bodies, which will be the simulation universe, but this is not strictly necessary in SHIELD-HIT12A. There are no requirements to how this body must be named or where it is placed in the body list, but for convenience it can be either the first or last body. The list of bodies must be terminated with the **END** card. The format of a body card is:

- 2 columns (1-2): unused, e.g. two empty spaces
- 3 columns (3-5): name of the body (see sec. A.6)
- 1 column (6): unused

- 4 columns (7-10): an integer describing the number of the body
- 6 x 10 columns (11-70): 6 arguments describing the body as mentioned in A.6.

A continuation card may be necessary, depending on what body is described. A continuation card must be blank in the first 10 columns (i.e. no name and no body number). The body section must be terminated with an **END** card.

3.4.3 Voxelized Body

The **VOX** body, is a special body, which will load a CT-cube formatted as the **VOXELPLAN** format used by **TRiP98**³. A CT-cube in the **VOXELPLAN** format consists of two files:

1. a header file with the *.hed* suffix, containing human readable information on the properties of the CT-cube, such as number of slices, number of bins along the X and Y axis, voxel dimensions etc.
2. a CT-cube file, containing the voxel Hounsfield units in binary floating point format.

The **VOX** body is treated as if it would be a normal **BOX**, however the dimensions are automatically extracted from the header file. However, **SHIELD-HIT12A** still needs to know where to place the CT-cube, and whether it should be rotated. **SHIELD-HIT12A** always assumes that the point (0,0,0) in the simulated universe coordinate system will be the isocenter. Thus, the isocenter position in **SHIELD-HIT12A** should be marked inside the CT-cube. This is conveyed to **SHIELD-HIT12A** using the first five arguments of the **VOX** card:

1. X - simulated universe isocenter x-position inside CT cube coordinate system [cm]
2. Y - simulated universe isocenter y-position inside CT cube coordinate system [cm]
3. Z - simulated universe isocenter z-position inside CT cube coordinate system [cm]
4. Couch angle [degrees]
5. Gantry angle [degrees]
6. Optional planned target dose as specified in **TRiP98** [Gy]

The last optional argument needs more clarification: when the **VOXSCORE** card is used in *detect.dat*, then **TRiP98** formatted data cubes may be created during postprocessing⁴. These **TRiP98** formatted cubes store the dose value units relative to the planned PTV dose. **SHIELD-HIT12A** calculates however the dose per simulated primary particle. So, in order to convert to the relative **TRiP** dose, **SHIELD-HIT12A** thus needs to know 1) The planned target dose (given by the last argument in the **VOX** card), and 2) the total particle budget. The total particle budget are derived from the external beam specifications loaded with the **USECBEAM** card, assuming these were crafted with the actual particle numbers deposited in each beam spot. Note, that specifying this option does not change any scoring or transport behaviour of **SHIELD-HIT12A**. Any raw output *.bdo* will still contain the values in their respective units, the difference is just, that the information on the target dose is shipped along in the header of the output files, providing useful information to any external postprocessing routines.

Arguments for the second card:

³See <http://bio.gsi.de/DOCS/TRiP98/PRO/DOCS/trip98fmt.html>

⁴See the *pymchelper* project on GitHub <https://github.com/DataMedSci/pymchelper>

1. here the filename is specified, relative to the working directory specified when invoking the `shieldhit` command. The filename must be specified without the `.hed` or `.ctx`, only the basename of the file must be supplied. The full line can be used for specifying the filename, free formatting.

From this point on, the VOX body may be treated like a BOX, however the user must make sure, that the zone logic is still valid. This may in particular become tricky close to the body boundaries in the case the cube is rotated. Note also that in principle it is possible to insert metal implants inside the CT-cube.

A word of caution must be mentioned, as voxel cubes allocate large amounts of memory. In particular running on parallel structures, nodes may run low on RAM and possibly begin to use swap space, which would effectively prevent SHIELD-HIT12A ever to complete. SHIELD-HIT12A prints out the memory to be allocated during detector initialization. A remedy is to reduce the amount of VOXELSCORE cards active in the simulation, and instead run the simulation several times with different VOXELSCORE cards.

3.4.4 Zones

The zone description follows afterwards. In order to construct different zones out of the bodies defined at the beginning of *geo.dat*. Boolean logic is used. Examples of how to construct zones out of two bodies are shown in figure 3.2.

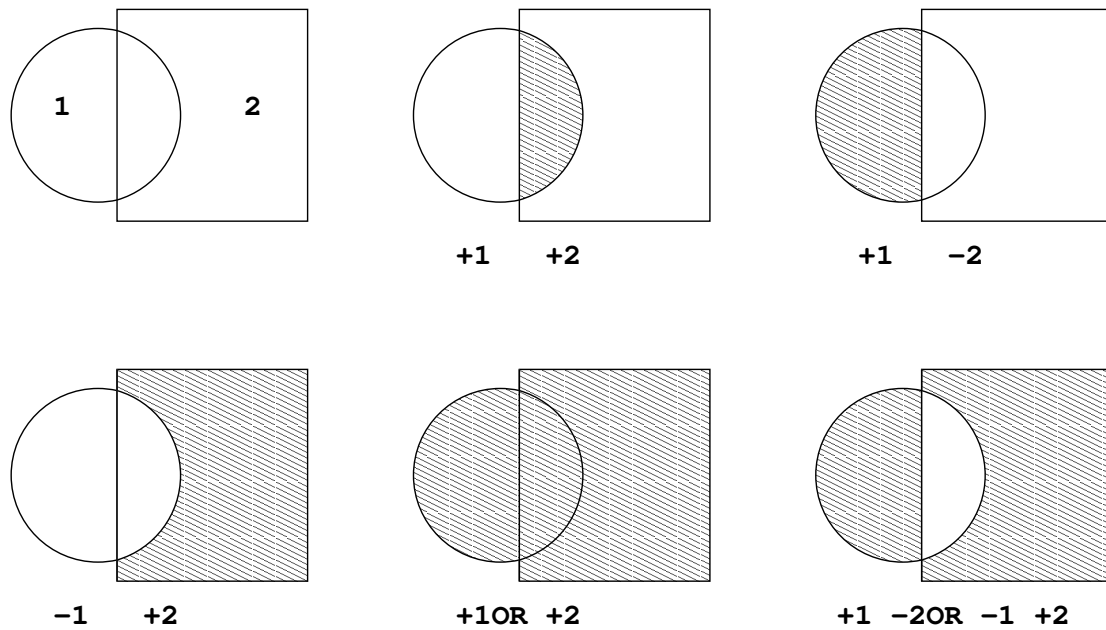


Figure 3.2: Two bodies (marked as 1 and 2) describing 5 different zones.

A line in the zone description consists of:

- 2 columns (1-2): unused, e.g. two empty spaces
- 3 columns (3-5): name of the zone
- 5 columns (6-10): an optional integer describing the number of the zone

- 9 x (2+5) columns (11-72): where each set of 9 columns are composed of two components. The two first columns hold the Boolean operator **OR** or are left blank if needed. The following 5 columns contain a signed (+,-) integer body number. It is essential that the alignment is strictly kept.

If more than 9 bodies are used for the zone description additional continuation cards can be added directly afterwards. These continuation lines must not have a zone name, i.e. there must be no entry at columns 3 - 5. The zone section must also be terminated with **END** card.

Example:

```
*.<-><--->..<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->
001          +1      -2      -3      -4      -5
002          +2
003          +3
004          +4
005          +5
006          +6      -1
END
*.<-><--->..<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->
```

It is important that any point (except for zone boundaries) where a particle may go, is described by exactly one zone, no more, no less. SHIELD-HIT12A may produce unexpected or random behavior if particles cross areas where two zones overlap, or a zone which is not defined. To some extent SHIELD-HIT12A will warn the user in such cases, but SHIELD-HIT12A may not catch all cases, therefore a careful assignment of zones is essential.

3.4.5 Media

Finally, what follows is the assignment of media to the various zones. External vacuum behaves like a black hole, any particle hitting it will be disintegrated and not transported further. The “top level” zone must be assigned to external vacuum (id = 0). Inside this zone you can place a zone with internal (i.e. real physical) vacuum (id = 1000) or any other material. In internal vacuum regular particle transport takes place, as opposed to the external vacuum. Even if not strictly mandatory, the external vacuum should encompass the entire system is to prevent infinite loops where stray particles are transported forever, and to avoid unintended behavior of the code. The media assignment is done in two equally formatted lists. The first list is a list of zones:

- 14 x 5 columns (1-70): list of integer zone numbers

The second list are the material numbers:

- 14 x 5 columns (1-70): list of integer media identifiers

If one line is not enough, then it is simply continued on the following line, until all zones are described. All zones must be assigned to medium, thus, both lists must be equally long.

3.4.6 Examples - Geometry

Example 1: Water tank surrounded by air

A simple example of the input geometric file *geo.dat* is given below.

```

0      0      simple water tank  20.09.2011
RPP    1    -100.0    100.0    -100.0    100.0    0.0    100.0
RPP    2    -150.0    150.0    -150.0    150.0   -150.0    150.0
RPP    3    -200.0    200.0    -200.0    200.0   -200.0    200.0
END
*,<->---->..<---->OR<---->OR<---->OR<---->OR<---->OR<---->OR<---->OR<---->OR<---->
TAR    1      +1
AIR    2      +2      -1
OUT    3      +3      -2
END
1      2      3
1      2      0

```

Example 2: Proton therapy beam line

A more complex example is a model of the beam line at the cyclotron facility in Clatterbridge, UK. It features two zones with external vacuum, where the second one is used to collimate the beam:

```

0      0      COO beam line Clatterbridge  13.05.2011
RCC    1      0.0      0.0      -25.6      0.0      0.0      1.00
15.0
RCC    2      0.0      0.0      -25.6      0.0      0.0      1.00
0.3
RCC    3      0.0      0.0      -22.6      0.0      0.0      0.003
15.0
RCC    4      0.0      0.0      -0.66      0.0      0.0      0.660
0.286
RCC    5      0.0      0.0      0.0      0.0      0.0      0.002
15.0
RCC    6      0.0      0.0      5.0      0.0      0.0      0.005
15.0
RCC    7      0.0      0.0      -30.0      0.0      0.0      35.005
15.0
RCC    8      0.0      0.0      27.0      0.0      0.0      0.001
20.0
RCC    9      0.0      0.0      55.0      0.0      0.0      1.00
20.0
RCC   10      0.0      0.0      55.0      0.0      0.0      1.00
2.0
RCC   11      0.0      0.0     115.00      0.0      0.0      0.002
20.0
RCC   12      0.0      0.0     115.002      0.0      0.0      0.002
20.0
RCC   13      0.0      0.0     175.0      0.0      0.0      7.80
20.0
RCC   14      0.0      0.0     175.0      0.0      0.0      7.80
1.70

```

```

RCC 15      0.0      0.0      182.80      0.0      0.0      0.80
          20.0
RCC 16      0.0      0.0      182.80      0.0      0.0      0.80
          1.5
RCC 17      0.0      0.0      5.005      0.0      0.0      178.995
          25.0
RCC 18      0.0      0.0      184.0      0.0      0.0      30.0
          20.0
RCC 19      0.0      0.0      -100.0      0.0      0.0      500.0
          100.0
RCC 20      0.0      0.0      -200.0      0.0      0.0      1000.0
          200.0
RCC 21      0.0      0.0      -26.0      0.0      0.0      1.00
          15.0
RCC 22      0.0      0.0      -26.0      0.0      0.0      1.00
          0.3
END
*.<-><--->..<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->
TAR  1      +18
BR1  2      +1      -2
BR2  3      +9      -10
BR3  4      +13     -14
BR4  5      +15     -16
HL1  6      +2
HL2  7      +10
HL3  8      +14
HL4  9      +16
SF1 10      +3
SF2 11      +5
STP 12      +4
KWD 13      +6
*.<-><--->..<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->OR<--->
VAC 14      +7      -1      -3      -4      -5      -6      -21
WHL 15      +8
ICM 16      +11
ICA 17      +12
CAR 18      +17     -8      -9      -11     -12     -13     -15
OAR 19      +19     -7     -17     -18
OUT 20      +20     -19
IBH 21      +21     -22
IHL 22      +22
END
  1   2   3   4   5   6   7   8   9  10  11  12  13  14
15 16 17 18 19 20 21 22
  1   2   2   2   2   2   2   3   3   9   9   2   5 1000
  8   6   4   3   3   0   0 1000

```

Example 3: Voxelized CT cube

This example demonstrates how a TRiP formatted CT cube is loaded. The couch angle is set to -90 degrees, and the iso-center (which is the (0,0,0) point of the SHIELD-HIT12A coordinate system) is located at (15.5, 15.5, 8.7) cm in the CT cube coordinate system.

```

*---><---><-----><----->
      0      0      Voxelplan test
*---><---><-----><-----><-----><-----><----->
* sphere - black body
  SPH    1      0.0      0.0      0.0    10000.0      0.0      0.0
*---><---><-----><-----><-----><-----><----->
* sphere - vacuum
  SPH    2      0.0      0.0      0.0     1000.0      0.0      0.0
*---><---><-----><-----><-----><-----><----->
* box/vox under consideration, water or CT
  VOX    3      15.5      15.5      8.6     -90.0      0.0
          ../res/TST001/tst001000
*---><---><-----><-----><-----><-----><----->
  END
  001      +1      -2
  002      +2      -3
  003      +3
  END
    1      2      3
* 0 - black hole, 1000 vacuum, 1 - water
    0 1000      1

```

3.5 *detect.dat* - Scoring

SHIELD-HIT12A needs to know what information from the particle transport the user is interested in. The process of recording relevant results generated at runtime is referred to as *scoring*. The file `detect.dat` defines the *scorer*, which is how the user can decide on what information SHIELD-HIT12A must record. SHIELD-HIT12A has currently two scoring systems available, which is referred to *the new* and *the old scoring system*. This section covers the *new scoring system*, which has replaced the *old scoring system* which is described later in section 3.6.

What data is to be scored is specified in the *detect.dat* file, which simply is an ASCII text file arranged in four sections:

1. **Geometry**: One or more sections, which describe the region where we wish to record data.
2. **Filter**: Optionally, one or more filters can be defined, which limits what particles should be included for scoring. These can later be attached to various detectors.
3. **Settings**: Optionally, additional settings can be defined in one or more **Settings** sections. These settings can then be attached to the detectors, if needed.
4. **Output**: One or more outputs, which may describe a list of quantities to be scored. **Output** are also referred to estimators. The quantities to be scored in this estimator are described with the **Quantity** key, and are followed by the detector name and one or more optional filters or settings which will be attached to this detector.

In other words, each *detect.dat* file must at least contain one **Geometry** and one **Output** section in order to produce a meaningful result.

These three sections will be described in more detail below, but before this a few general features of the *detect.dat* must be mentioned:

- A line in *detect.dat* consists of a keyword, possibly followed by one or more arguments, i.e.: *keyword argument_1 argument_2 argument_3*
- Keywords are not case sensitive.
- User given names and filenames *are* case sensitive.
- Comments are marked with leading # (preferred), * or !.
- The *detect.dat* file may contain tabs or spaces.
- Tabs and spaces are allowed.
- Any indentations are allowed.
- Each argument is separated by at least one or more spaces.

The output file format can be chosen, and the options are described below. By default, the results are saved in binary *.bdo* format, which can be read by the reader provided by *pymchelper*⁵. Pymchelper provides also averaging functionality needed when running SHIELD-HIT12A on computer clusters [?].

⁵<https://github.com/DataMedSci/pymchelper>

3.5.1 Geometry Section

To understand the geometry section, it is easiest to see a simple example:

```
Geometry Mesh
  Name MyMesh
    X -10.5  10.5    1
    Y -10.5  10.5    1
    Z  0.0   40.0   800
```

In the above example, the scoring geometry is a box 21 x 21 x 40 cm³, where one surface is centered at the point (0,0,0). This box is then divided into 800 bins along z. This would correspond to a 1-D scoring, e.g. an integrated depth-dose (idd) curve may be obtained this way. The entire geometry is named **MyMesh**.

The **Geometry** keyword signals that a new geometry definition is to follow. The **Geometry** keyword takes exactly one argument, which is the *geometry type*. Available geometry types are listed in table 3.1.

<i>geometry type</i>	<i>description</i>
Mesh	scoring using a Cartesian mesh geometry
Cyl	scoring using a cylindrical mesh geometry
Zone	scoring within one or more zones as specified in the <i>geo.dat</i> file.
All	scoring within the entire universe, only useful with certain detectors (e.g. Trace).

Table 3.1: List of available geometry types.

Somewhere within a single geometry section, the geometry must be given a name so it can be referred to later in the file. This is done with the **Name** keyword which takes exactly one argument. The user given geometry name should be a single word and contain no white spaces. In the example above, the geometry given the name **MyName**. The name is case sensitive.

Depending on the geometry type, the lower and upper coordinate boundary must be specified, as well as the amount of bins. The keywords for the different geometry types as well as their arguments are given in table 3.2.

For zone scoring a **Volume** keyword is needed, since SHIELD-HIT12A has no algorithm implemented to automatically calculate the zone volumes. If a range of zones is specified, this volume will be applied equally to all zones (this may change in a future release.) For the **All** geometry, the default is 1.0 cm³, if the the volume is not given by the user.

Example 1 - 2D Scoring

```
Geometry Mesh
  Name MyMesh
    X -10.0  10.0  400
    Y -0.5   0.5   1
    Z  0.0   20.0  400
```

Defines a 1 cm thick 2D scoring volume in the X-Z plane divided into 400 x 400 pixels.

<i>geometry</i>	<i>keyword</i>	<i>argument_1</i>	<i>argument_2</i>	<i>argument_3</i>
Mesh	X	x_{\min}	x_{\max}	<i>nr. of bins</i>
	Y	y_{\min}	y_{\max}	<i>nr. of bins</i>
	Z	z_{\min}	z_{\max}	<i>nr. of bins</i>
Cyl	R (radius)	r_{\min}	r_{\max}	<i>nr. of bins</i>
	Z (depth along z axis)	z_{\min}	z_{\max}	<i>nr. of bins</i>
Zone	Zone	<i>first zone nr.</i>	<i>(optional) last zone nr.</i>	
	Volume	<i>volume of zone</i> <i>in [cm³]</i>		

Table 3.2: Table of keywords for the available three different geometry types.

Example 2 - 3D Scoring

```

Geometry Mesh
  Name MyMesh
    X -10.0  10.0  400
    Y -10.0  10.0  400
    Z  0.0   20.0  400

```

Defines a 3D scoring volume where a 20 x 20 x 20 cm³ cube sliced into 400 x 400 x 400 voxels.

Example 3 - Cylindrical Scoring

```

Geometry Cyl
  Name MyCyl
    R  0.0   0.5   1
    Z  0.0  20.0  400

```

Defines a 1 cm diameter cylinder, 20 cm along the z-axis divided into 400 bins.

Example 4 - Zone Scoring

```

Geometry Zone
  Name MyZone
    Zone 3
    Volume 13.37

```

Scoring is averaged over entire zone number 3, which has the volume 13.37 cm³.

3.5.2 Filter Section

In the filter section, filter rules can be defined by the user, which then later can be applied to the detectors for scoring.

The filter section is optional, and is started with the **Filter** keyword. In the filter section, there must be one occurrence of the keyword **Name** followed by a user given name of that filter as an argument to the **Name** keyword.

Name : used to specify the name of the filter. It takes one argument, which is a user given name. The name should be a single word and not contain any whitespace. The filter name can be linked to a **Quantity** in the **Output** section. The name is case sensitive.

Filter Rules

The user may define one or more rules for filtering the particles to be scored. The filter rule is permissive, that is, the rules define what particles will be *accepted* for scoring. The rule parser is very simplistic. Each rule is contained in a single line, which consists of a keyword, followed by an operator, and then a numeric value, such as:

ENUC > 100.0

which tells the scorer only to include particles with energies above 100 MeV/nucleon.

Available filter keywords are listed in table 3.3 below.

<i>keyword</i>	<i>description</i>
A	creates a filter rule on particle nucleon number <i>A</i> .
AMASS	creates a filter rule on particle mass <i>m</i> , measured in [MeV/c ²].
AMU	creates a filter rule on particle mass <i>m</i> , measured in atomic units.
E	creates a filter rule on kinetic energy, measured in [MeV].
EKIN	same as E .
ENUC	creates a filter rule on kinetic energy per nucleon, measured in [MeV/n].
EAMU	creates a filter rule on kinetic energy per amu, measured in [MeV/amu].
ID	creates a filter rule on particle id, see table 3.4.
JPART	[NOT IMPLEMENTED YET] creates a filter rule on particle type JPART , following the definition in A.3.
GEN	creates a filter rule on the generation number of the particle. Generation 0 is the primary particle. Generation 1 is the first secondary particle, etc. . . .
NPRIM	Primary particle number currently simulated. This is useful together with the TRACE detector to select what particles to record.
Z	creates a filter rule on particle charge <i>Z</i> .

Table 3.3: List of available filter rules.

Filter Operators

Available filter operators are listed in table 3.5 below.

<i>particle ID</i>	<i>description</i>
1	Hadrons
2	Anti-hadrons
3	π^- pi- meson
4	π^+ pi+ meson
5	π^0 pi0 meson
6	(do not use)
7	(do not use)
8	K^-
9	K^+
10	K^0
11	K^\sim
12	γ -ray
13	electron
14	positron
15	μ^-
16	μ^+
17	ν_e electron neutrino
18	$\bar{\nu}_e$ electron anti-neutrino
19	ν_μ mu neutrino
20	$\bar{\nu}_\mu$ mu anti-neutrino

Table 3.4: List of particle IDs which are used for filter rules.

<i>operator</i>	<i>description</i>	<i>operator</i>	<i>description</i>
<	less than	<=	less or equal than \leq
>	greater than	>=	greater or equal than \geq
= (or ==)	equal	!=	not equal

Table 3.5: List of available filter operators.

Example 1 - Proton Scoring

Filter for only scoring protons.

```
Filter
  Name MyFilter
  Z = 1
  A = 1
```

Example 2 - Primary Ions

Filter for only scoring primary C-12 ions with more than 100 keV energy:

```
Filter
  Name MyFilter
  Z = 6
  A = 12
  E > 0.1
  GEN = 0
```

3.5.3 Settings Section

A detector can also accept certain settings. For example if dose-to-water is needed even if the medium in which the particle is transported is different than water, then a scoring medium can be defined. This is done internally by multiplying with the mass stopping power ratio of water and medium. Within a single settings section, the settings must be given a name so it can be referred to later in the file. This is done with the **Name** keyword which takes exactly one argument. The user given settings name should be a single word and contain no white spaces. The name is case sensitive.

All available settings keywords are listed in table 3.6 below.

<i>keyword</i>	<i>description</i>
Medium	override scoring medium with the given medium number (from <i>mat.dat</i>). Works for dose and LET scorers.
Name	Gives this section a name which is used to attach it to a Quantity
NKMedium	[Only for NKERMA and NEqvDose detector] Selects medium for the neutron kerma detector. For available media, see table 3.7. If not set, Medium defaults to Water for the NKERMA detector.
Offset	adds an offset, so $y' = y \cdot \text{Rescale} + \text{Offset}$.
Primaries	Converts <i>per primary</i> detectors to absolute units by multiplying with exactly this amount of primaries. Useful to calculate absolute dose.
Rescale	rescale detector with this factor. Useful for converting units.
SiteDiameter	define site diameter for microdosimetric scorers, in [nm]. default is 500 nm.

Table 3.6: List of available filter rules.

<i>Medium</i>	<i>description</i>
A150	Tissue-equivalent A-150 plastic.
AIR	Air, sea-level dry.
BONE	Bone tissue.
Muscle	Muscle tissue.
TE.Methane	Tissue-equivalent gas, methane based.
TE.Propane	Tissue-equivalent gas, propane based.
WATER	Water medium.

Table 3.7: List of available media for scoring neutron Kerma **NKERMA**. This is based on ICRU 63 [?].

Example 1 - Dose-to-water

Monte Carlo transport codes report on dose-to-medium by default, where medium refers to the material present in the scoring volume. However, in clinical settings dose-to-water is typically reported. Here we assume that material number 3 in *water.dat* is water:

Settings

```
Name MyWater
```

```
Medium 3      # assuming that material 3 is given in mat.dat as water.
```

and `MyWater` can then be added to a detector, just like a filter (see full examples at the end of sec.3.5).

3.5.4 Output Section

The **Output** keyword signals that a new geometry section is to follow. The **Output** keyword has no arguments. The keywords specific to the **Output** section are listed in table 3.8.

<i>Output keyword</i>	<i>description</i>
Filename	Sets the filename for output.
. Fileformat	Format will be guessed from the given file extension, see table 3.9. Optional. Select output file format, see table 3.9. This will override any automatic format discovery by file extension.
Geo	Attaches a user-defined geometry for scoring. If no geometry is given, then and All geometry is attached. If no All geometry was specified by the user, then one will be generated with a volume of 1 cm ³ .
Quantity	argument_1: Selects what quantity is to be scored, see table 3.10. (optional) following arguments attaches user-defined filters and settings by their name. The filter and settings list must be space separated. Filters and settings must be unambiguously named.

Table 3.8: List of available keywords for the **Output** section.

<i>Fileformat</i>	<i>File extension</i>	<i>description</i>
ASCII	.txt	Output results as human readable ASCII text.
BDO	.bdo	Binary Data Object. This is the default format.
BDZ	.bdz	Compressed BDO.
CSV	.csv	[NOT IMPLEMENTED]
TEXT	.txt	as ASCII
TXT	.txt	as ASCII
DAT	.dat	as ASCII

Table 3.9: List of available file formats. If **Fileformat** is not set, these will be automatically be discovered by the file extension given to the filename.

- **DoseEqv** can be used to derive operational quantities such as personal dose-equivalent ($H_p(d)$) ambient dose-equivalent ($H^*(d)$) or directional dose-equivalent quantities ($H'(d, \Omega)$).
- **DDD** scorer is used for the TRiP98 .ddd files, as described in ⁶.
- **EqvDose** is calculated using the ICRP 103 weighting coefficient w_R . Anti-particles (positrons, antiprotons,...) are treated as their respective counterpart, even if they are not mentioned in ICRP 103. Uncharged particles (neutrons) are ignored since they deposit no dose.
- **NEqvDose** is calculated using the ICRP 103 weighting coefficient w_R for neutrons. Neutron kerma is calculated as with **NKERMA**, so **NKMedium** setting must be supplied by attaching a **Settings** section, just as in the case using **NKERMA**, otherwise **Water** is assumed. This card does $H_T = K_n \cdot w_R$.
- The **NKERMA** detector uses the kerma coefficients listed in table C.3 in ICRU Report 63 [?], which is only defined for neutron energies between 11 eV to 150 MeV. As an approximation,

⁶<http://bio.gsi.de/DOCS/TRiP98/PRO/DOCS/trip98fmtddd.html>

<i>Detector</i>	<i>description</i>
1MeVNEq	1-MeV neutron equivalent fluence [cm^{-2}]. Only scored for neutrons, protons and pions. Multiply with 2.037e-3 to get D_{NIEL} in [MeV / g].
Alanine	Alanine quenching model for ions, scores Dose * RE(z,E). Based on Bassler et al. NIMB;2008;266(6);929-936.
AvgBeta	Track-averaged β , where $\beta = v/c$.
AvgEnergy	Average kinetic energy of the particle in [MeV/nucleon].
Count	count transport steps or point-like events in this volume.
CountPoint	Count point-like events in this volume (e.g. below transport threshold).
DDD	as Dose , but specially for TRiP98 depth-dose kernel file generation.
Dose	Dose [MeV/g]
DoseEqv	Dose-Equivalent (see notes below) [Sv].
DoseGy	Dose [Gy].
dLET	Dose-averaged LET [MeV/cm].
dQ	Dose-averaged Q .
dQeff	as dZeff2Beta2 .
Energy	Total amount of energy deposited [MeV].
EqvDose	Equivalent dose (see notes below) [Sv].
Fluence	Fluence [cm^{-2}].
MATERIAL	Maps material ID as assigned in <i>geo.dat</i> . Useful for debugging geometries.
NEqvDose	Equivalent dose from neutron kerma (see notes below) [Sv].
NormCount	as Count , but normalized per primary particle.
NormCountPoint	as CountPoint , but normalized per primary particle.
NKERMA	Neutron Kerma in [Gy], based on ICRU 63 [?].
NMEDIUM	alias as MATERIAL .
NZONE	Maps zone number. Useful for debugging geometries.
Q	alias for dQ .
Qeff	as dZeff2Beta2 .
Rho	Maps material density as assigned in <i>geo.dat</i> and <i>mat.dat</i> [g/cm^3]. Useful for debugging geometries.
tQ	Track-averaged Q .
tQeff	as tZeff2Beta2 .
tLET	Track-averaged LET [MeV/cm].
TRACE	Dumps all steps to a <i>.csv</i> file. This detector must stand alone in an Output section, as it is incompatible with other detectors. By default 10 primaries are traced. Use an userdefined filter, if you need a certain number of projectiles.
User1	User-defined scoring, for users with source-code access. Defaults to Fluence .
User2	User-defined scoring, for users with source-code access. Defaults to Fluence .
ZONE	Alias for NZONE .
Z2Beta2	alias for dZ2Beta2 .
dZ2Beta2	Dose-averaged z^2/β^2 .
tZ2Beta2	Track-averaged z^2/β^2 .
Zeff2Beta2	alias for dZeff2Beta2 .
dZeff2Beta2	Dose-averaged z_{eff}^2/β^2 .
tZeff2Beta2	Track-averaged z_{eff}^2/β^2 .

Table 3.10: List of available detectors for the **Quantity** keyword.

SHIELD-HIT12A sustains the closest kerma coefficient when the neutron energy is out of bounds. `NKMedium` setting must be supplied by attaching a `Settings` section, otherwise `Water` is assumed.

- Effective charge is calculated as $z_{\text{eff}} = z(1 - \exp(-125\beta z^{-2/3}))$, see [?].
- `Q` is based on publication [?]. Notice that this scorer as well as `dQ` and `tQ` are divergent at low energies. Adding an energy cutoff is recommended for better control of this scorer. Particles below transportation threshold are ignored, irrespectively of any user setting. For `Q` scoring and averaging, only charged particles are included.
- `Qeff` is based on publication [?]. It is not sensitive to a threshold, unlike `Q`.
- `Z2BETA2` and the dose and track-averaged derived scoreres are also divergent, hand handled the same way as the `Q` scorer. For `Z2BETA2` scoring and averaging, only charged particles are included.
- For `ZEFF2BETA2` scoring and averaging, only charged particles are included.
- Particles falling below the transportation threshold are gradually slowed down (without further transport) for the detectors `DLET`, `TLET`, `DZEFF2BETA2`, `TZEFF2BETA2` and `DOSEEQV`. For the detectors `ENERGY`, `DOSE`, `DDD`, `DOSEGY` and `EQVDOSE` the remaining kinetic energy is dumped on the spot where the particle falls below transport threshold. For all other detectors, the remaining energy is ignored.
- More notes on LET averaging please see [?].

Differential Scoring

It is possible to add single or double differential scoring to a detector set by the `Quantity` keyword. In the single differential case, this is done by adding two lines with the keywords `Diff1` and `Diff1type`, respectively, immediately after the line with the "Quantity" keyword. The `Diff1` keyword describes the binning dimensions with three arguments following the keyword, as shown in table 3.11. The binning is in linear steps by default. Logarithmic binning can be enabled by setting the `log` flag as the 4th argument.

<i>keyword</i>	<i>arg-1</i>	<i>arg-2</i>	<i>arg-3</i>	<i>arg-4</i>
<code>Diff1</code>	<i>lower limit</i>	<i>upper limit</i>	<i>nr. of bins</i>	(optional) <code>log</code>
<code>Diff1type</code>	Type of differential scoring, see table 3.12.			

Table 3.11: Table of keywords for the available three different geometry types.

In the double differential case, two more keywords are added `Diff2` and `Diff2Type`, which functions fully analogous.

<i>DiffType</i>	<i>description</i>
ANGLE	differential in angle [°]
DEDX	differential in unrestricted electronic stopping power [MeV/cm]
E	differential in kinetic energy [MeV]
EAMU	differential in kinetic energy per amu [MeV/amu]
EKIN	as E
ENUC	differential in kinetic energy per nucleon [MeV/n]
LET	as DEDX
MDEDX	differential in unrestricted electronic mass stopping power [MeV cm ² /g]
TL	differential in track length [cm]
Z	differential in projectile charge
Zeff	differential in effective projectile charge $z_{\text{eff}} = z(1 - \exp(-125\beta z^{-2/3}))$
Z2Beta2	differential in z^2/β^2
Zeff2Beta2	differential in z_{eff}^2/β^2

Table 3.12: List of differential scorers.

Example 1 - Energy Scoring

Score energy deposited in an earlier defined geometry. Output results into *NB_msh_energy.bdo*:

Output

```
Filename NB_msh_energy.bdo
Geo MyMesh
Quantity ENERGY
```

Example 2 - Energy and Filtered Fluence

Score energy and fluence for all particles, but also fluence only including those which pass the filter defined in MyFilter. All results will be written to a single files which then contains three pages:

Output

```
Filename NB_msh_fluence.bdo
Geo MyMesh
Quantity ENERGY
Quantity FLUENCE
Quantity FLUENCE MyFilter_C12 MyFilter_Gen0 # add a second page,
                                                # but with a filter applied
```

Example 3 - Scoring Dose- and Track-averaged LET

Score energy, fluence, dose, dose-averaged LET and track-averaged LET. Output results as a human-readable formatted text file *NB_msh.dat*

Output

```
Filename NB_msh.dat
Fileformat TEXT
Geo MyMesh
Quantity Energy
Quantity Fluence
Quantity Dose
Quantity DLET
Quantity TLET
```

Example 4 - Double Differential Scoring

Score double-differential fluence while filtering.

Output

```
Filename NB_msh_diff2_fluence.bdo
Geo MyMesh
Quantity      FLUENCE ProtonFilter
Diff1         0.1   500   10      log # triggers diff. scoring in log bins
Diff1Type     ENUC                  # Energy / nucleon number
Diff2         0    90    6          # triggers double differential scoring
Diff2Type     ANGLE                  # angle
```

Example 5 - Settings

An example using rescale (even if unnecessary since DoseGy does the same), multiplying with total number of primaries to get absolute dose, and score dose-to-water.

Settings

```
Name AbsoluteDoseToWater
Rescale 1.602e-10      # MeV/g -> J/kg
Primaries 1.34e9       # total number of primaries
Medium 3               # Assuming that material 3 in mat.dat is water
```

Output

```
Filename NB_dose_1Gy.dat
Fileformat Text
Geo MyMesh
Quantity Dose          # in [MeV/g/primary]
Quantity DoseGy        # in [Gy/primary]
Quantity Dose AbsoluteDoseToWater # [Gy]
```

3.5.5 Complete Examples

Complete *detect.dat* examples are listed below.

Example 1 - Typical Scoring Example

The example below demonstrated depth-dose scoring along the z-axis. This happens both in Cartesian and cylindrical coordinates. Apart of dose, also energy, fluence, dose- and track-average LET is recorded for *all* transported particles.

```
# Very simple test of new scoring system
```

```
Geometry Mesh                                # Cartesian geometry type
  Name MyMesh
  X -10.5  10.5    1
  Y -10.5  10.5    1
  Z  0.0  40.0   800

Geometry Cyl                                # Cylindrical geometry type
  Name MyCyl
  R  0.0  11.84798  1
  Z  0.0  40.0   800
```

Output

```
Filename NB_msh.dat
Fileformat TEXT
Geo MyMesh
Quantity Energy
Quantity Fluence
Quantity Dose
Quantity DLET
Quantity TLET
```

Output

```
Filename NB_cyl.dat
Fileformat TEXT
Geo MyCyl
Quantity Energy
Quantity Fluence
Quantity Dose
Quantity DLET
Quantity TLET
```

Example 2 - Neutron Kerma in Muscle Tissue

This example demonstrates a narrow scoring area, where the total fluence as well as fluence from neutrons only is recorded. Furthermore, the neutron Kerma in muscle tissue is recorded, using the kerma-factors from ICRU 63 table C.3 [?]. Additionally, the equivalent-dose from neutron kerma is calculated. (This is done just as a technical example, even if the 1-mm slab volumes

do not correspond to an organ. A slightly example would be a sphere which would correspond to an organ, over which NEQVDOSE is scored.)

Geometry Mesh

```
Name MyMesh          # geometry type
X -0.5  0.5    1
Y -0.5  0.5    1
Z  0.0  40.0  400
```

Filter

```
Name MyNeutrons      # Name
Z = 0
A = 1
```

Settings Set_A150

```
NKMedium A150
```

Output

```
Filename NB_neutrons.dat
Fileformat Text
Geo MyMesh
Quantity FLUENCE
Quantity FLUENCE MyNeutrons
Quantity NKERMA Set_A150      # in Gy per primary particle
Quantity NEQVDose Set_A150    # in Sv per primary particle
```

Example 3 - Scoring with Traces

This demonstrates how the TRACE scorer is applied. Note that no Geometry is attached to the trace scorer, thereby it will be active in the entire simulation universe. The trace scorer needs its own file to write to (here *NB_trace.csv*), and can therefore not be used together with the detectors which outputs to the *.dat* file.

Geometry Mesh

```
Name MyMesh          # geometry type
X -0.5  0.5    1
Y -0.5  0.5    1
Z  0.0  40.0  400
```

Filter

```
Name MyFilter
NPrim <= 20          # only score the first 20 primaries.
                     # do not set this too large, as the file size will grow very fast.
```

Output

```
Filename NB_dose.dat
Fileformat TEXT
Geo MyMesh
Quantity DoseGy
Quantity Fluence          # for comparison to the next two.
Quantity User1            # HomeMade detector
```

```
Quantity User2                # HomeMade detector

Output
  Filename NB_trace.csv
  Quantity Trace  MyFilter    # results will be dumped into a csv file.
```

Example 4 - Generate Treatment Planning Depth-Dose Kernels

The treatment planning system TRiP98 requires depth-dose kernels to generate treatment plans. These are in a *.ddd* format⁷, and can be directly generated by *Pymchelper*⁸ from the bdo files generated by SHIELD-HIT12A.

```
Geometry Cyl
  Name MyCyl                # geometry type
  R  0.0  20.0  200
  Z  0.0  40.0  400

Output
  Filename protons_ddd.bdo
  Geo MyCyl
  Quantity DDD
```

⁷<http://bio.gsi.de/DOCS/TRiP98/PRO/DOCS/trip98fmtddd.html>

⁸<https://github.com/DataMedSci/pymchelper>

3.6 *detect.dat* - Scoring (Old Style)

This section refers to the "old" scoring system, which gradually will become obsolete, since it is too difficult to maintain. The old scoring system will gradually be replaced by the new scoring system as previously described in section 3.5. Currently, there is little reason left to use this scorer, and there are some bugs in it which are fixed in the new scoring system. It is only kept for backwards compability.

In this scoring system, different geometrical scoring types, called *estimators*, are available for auxiliary scoring: GEOMAP, ZONE, DZONE, CYL, MSH, PLANE DCYL, DMSH, and DPLANE. Each estimator can be invoked with a scoring card and possible following cards in the file *detect.dat*. Most estimators need additional specification of a *detector*, which controls what kind of quantity is scored. The available detectors are described afterwards in section 3.6.9.

Each field is exactly 10 chars long, except for the output filename, which may be 4096 characters long. Only ASCII characters are allowed. Similar to *geo.dat*, tabs are not allowed and will lead to segfaults, space must be used instead. The output of each estimator is written, in binary form, to a *.bdo* file, which is specified by the user. The *.bdo* suffix is automatically attached, if missing. The recommended post-processing scripts *convertmc* is open-source and is currently not included in the SHIELD-HIT12A distribution. It must be installed separately from the *pymchelper* package [?].

<https://github.com/DataMedSci/pymchelper>.

3.6.1 GEOMAP - Mapping the geometry

GEOMAP : Mapping the geometry. This card is useful for debugging the geometry. Using the arguments below, a 1,2 or 3-D mesh can be created, where the points will hold either the zone number, medium number specified in *mat.dat* or the density of the medium in [g/cm³]. For voxelized structures, the actual density in the individual voxels will be scored.

The card is only invoked during initialization, and has therefore no affect on run-time performance of SHIELD-HIT12A. The resulting *.bdo* files are independent of any particles transported, i.e. when the user is debugging the geometry, the particle number can be set to zero (either by using the *-n* command line option or the *NSTAT* card in *beam.dat*, in order to quickly generate the output files.

In case of parallelization using the *-N* flag, the **GEOMAP** card will be ignored and *.bdo* files are not generated, as it makes no sense to clutter the directory with multiple data cubes with identical data in.

The arguments for the **GEOMAP** files are:

1. X_{min} : lowest X position.
2. Y_{min} : lowest Y position.
3. Z_{min} : lowest Z position.
4. X_{max} : highest X position.
5. Y_{max} : highest Y position.
6. Z_{max} : highest Z position.

Arguments for the second card:

1. X_{bin} : number of bins in X direction.
2. Y_{bin} : number of bins in Y direction.

3. Z_{bin} : number of bins in Z direction.
4. `unused`
5. `ZONE/MEDIUM/RHO`: Quantity to be mapped. This can be either the number of the zone (`ZONE`), or the number of the medium (`MEDIUM`), or the density of the material (`RHO`)
6. `outputfile`: output file name.

3.6.2 ZONE - Scoring by zone

ZONE : Scoring by zones w_i . Zones are defined from bodies in *geo.dat*. A single or a range of zones can be specified. Arguments are:

1. w_i : first zone of a range of zones to be scored.
2. w_j : optional last zone of a range of zones. If this is not specified, then only a single zone w_i will be scored. Else every zone within the interval $[w_i, w_j]$ will be scored.
3. `unused`.
4. `JPART`: particle type. See list in section A.3. Setting this value to -1 scores all particles. If heavy ions are scored (`JPART=25`) then a continuation card is required, where the first argument is an integer representing the particle charge Z , and the second argument is an integer representing the particle mass A .
5. `DETECTOR`: quantity that should be scored. **Important:** Zone scoring works straightforwardly only with the detectors `ENERGY`, `CROSSFLU`, `DLET`, `COUNTER` and `PET` as described in section 3.6.9. All other detectors can be applied as well, but require knowledge of the zone volume, which is not known by `SHIELD-HIT12A` (unlike the `MSH` and `CYL` scorer). Therefore the user must divide the scoring result with the corresponding zone volume (in cm^3) for all other detectors than those listed above. This may be changed in a future release (see ticket #177).
6. `outputfile`: output file name.

Analogue to `CYL` and `MSH`, a continuation card is only required if `JPART=25` or if `DETECTOR` is specified as `LETFLU` (and only `LETFLU`). Arguments for the continuation card are:

1. Z : an integer specifying the charge of the particle to be scored.
2. A : an integer specifying the number of nucleons of the particle to be scored. Setting this value to -1 scores all isotopes.
3. `MEDIUM`: an integer specifying the medium for LET weighted scoring. If left unspecified, the default is the medium material that the bin is made of. Any other integer from the medium list specified in *mat.dat* is valid here. In that case, scoring will apply the stopping powers of the medium specified here. Transport will irrespective of the scoring always be done in the as medium specified in *geo.dat*. This is useful for calculating stopping power ratios needed by various cavity theories in radiation dosimetry.
4. `unused`.
5. `unused`.
6. `unused`.

3.6.3 CYL - Cylindrical scoring

CYL : Cylindrical scoring. Cylindrical coordinates are used. Currently, scoring can only be done along the Z axis. This card requires a second succeeding card. Arguments are:

1. R_{min} : inner radius of scoring cylinder. If $R_{min} > 0$ then the scoring volume is a cylinder shell.
2. Set this to 0.0
3. Z_{min} : start position of cylindrical scoring in Z direction.
4. R_{max} : outer radius of scoring cylinder.
5. Set this to 7.0
6. Z_{max} : end position of cylindrical scoring in Z direction.

Arguments for the second card:

1. R_{bin} : number of bins in radial direction.
2. Set this to 1
3. Z_{bin} : number of bins in Z direction.
4. **JPART**: particle type. See list in section A.3. Setting this value to -1 scores all particles. If heavy ions are scored (**JPART**=25) then a third continuation card is required, where the first argument is an integer representing the particle charge Z , and the second argument is an integer representing the particle mass A .
5. **DETECTOR**: quantity that should be scored. A list of available detectors is given in section 3.6.9.
6. *outputfile*: output file name.

Analogue to **ZONE** and **MSH** a continuation card is only required if **JPART**=25 or if **DETECTOR** is specified as **LETFLU** (and only **LETFLU**). Arguments for the third card are:

1. Z : an integer specifying the charge of the particle to be scored.
2. A : an integer specifying the number of nucleons of the particle to be scored. Setting this value to -1 scores all isotopes.
3. **MEDIUM**: an integer specifying the medium for LET weighted scoring. If left unspecified, the default is the medium material that the bin is made of. Any other integer from the medium list specified in *mat.dat* is valid here.
4. unused.
5. unused.
6. unused.

3.6.4 MSH - Cartesian scoring

MSH : Cartesian mesh scoring card. This card requires a second succeeding card. Arguments for the first card:

1. X_{min} : lowest X position.
2. Y_{min} : lowest Y position.
3. Z_{min} : lowest Z position.
4. X_{max} : highest X position.
5. Y_{max} : highest Y position.
6. Z_{max} : highest Z position.

Arguments for the second card:

1. X_{bin} : number of bins in X direction.
2. Y_{bin} : number of bins in Y direction.
3. Z_{bin} : number of bins in Z direction.
4. **JPART**: particle type. See list in section A.3. Setting this value to -1 scores all particles. If heavy ions are scored (**JPART**=25) then a third continuation card is required, where the first argument is an integer representing the particle charge Z , and the second argument is an integer representing the particle mass A .
5. **DETECTOR**: quantity that should be scored. A list of available detectors is given in section 3.6.9.
6. *outputfile*: output file name.

Analogue to **ZONE** and **CYL** a continuation card is only required if **JPART**=25 or if **DETECTOR** is specified as **LETFLU** (and only **LETFLU**). Arguments for the third card are:

1. Z : an integer specifying the charge of the particle to be scored.
2. A : an integer specifying the number of nucleons of the particle to be scored. Setting this value to -1 scores all isotopes.
3. **MEDIUM**: an integer specifying the medium for LET weighted scoring. If left unspecified, the default is the medium material which the bin is made of. Any other integer from the medium list specified in *mat.dat* is valid here.
4. unused.
5. unused.
6. unused.

3.6.5 PLANE - Scoring by 2D plane

PLANE : Scoring by a single 2-D infinite plane s , independently of *geo.dat*. The plane is specified by a random point which must lie in the plane, and the normal vector of that plane. The normal can either be specified in this card, or - if omitted - the normal vector will automatically be calculated from the beam direction specified by the **BEAMTHETA** and **BEAMPHI** cards in *beam.dat*.

Arguments are:

1. S_x : X coordinate of point in plane.
2. S_y : Y coordinate of point in plane.
3. S_z : Z coordinate of point in plane.
4. n_x : x component of normal vector.
5. n_y : y component of normal vector.
6. n_z : z component of normal vector.

Arguments for the second card:

1. unused.
2. unused.
3. unused.
4. **JPART**: particle type. See list in section A.3. Setting this value to -1 scores all particles. If heavy ions are scored (**JPART**=25) then a third continuation card is required, where the first argument is an integer representing the particle charge Z , and the second argument is an integer representing the particle mass A .
5. **DETECTOR**: quantity that should be scored. A list of available detectors is given in section 3.6.9.
6. *outputfile*: output file name.

Analogue to **CYL** and **MSH**, a continuation card is only required if **JPART**=25 or if **DETECTOR** is specified as **LETFLU** (and only **LETFLU**). Arguments for the continuation card are:

1. Z : an integer specifying the charge of the particle to be scored.
2. A : an integer specifying the number of nucleons of the particle to be scored. Setting this value to -1 scores all isotopes.
3. **MEDIUM**: an integer specifying the medium for LET weighted scoring. If left unspecified, the default is the medium material that the bin is made of. Any other integer from the medium list specified in *mat.dat* is valid here. In that case, scoring will apply the stopping powers of the medium specified here. Transport will irrespective of the scoring always be done in the as medium specified in *geo.dat*. This is useful for calculating stopping power ratios needed by various cavity theories in radiation dosimetry.
4. unused.
5. unused.
6. unused.

3.6.6 DZONE, DCYL, DMSH and DPLANE - Differential scoring

DZONE, DCYL, DMSH and DPLANE : Differential zone, cylindrical and mesh scoring. Same as ZONE, CYL, MSH and PLANE respectively, but with an extra card that specifies the differential dimension of the scoring. The differential scoring card must be inserted before the heavy ion selection card, if that is used as well.

Arguments for the extra card:

1. Δ_{start} : Start value of differential binning.
2. Δ_{stop} : Stop value of differential binning.
3. Δ_n : Number of bins. If this value is negative, then binning is done in logarithmic steps, and the Δ_{start} parameter must be larger than zero.
4. internal use, keep this field clear.
5. **Type**: Type of differential scoring. Currently, **ENERGY**, **LET** and **ANGLE** are available, being differential in MeV/n, MeV/cm or radians, respectively.
6. unused.

The optional third card for CYL and MSH is the fourth card when using DCYL and DMSH.

If applying differential scoring on pions, their energies will be converted to specific energies (MeV/n) by multiplying with 0.1498 (π^+ and π^- mass divided by proton mass)⁹.

⁹In the case of π^0 it is 0.1448 which the π^0 mass divided by proton mass. However, SHIELD-HIT12A does not transport π^0 as it decays instantaneously.

3.6.7 TRACE - Dump particle tree

TRACE : A single card, which instructs SHIELD-HIT12A to dump all transport steps of all particles into a comma delimited file. This estimator is useful for debugging and visualizing particle trajectories across the simulated universe. Each line contains the start and stop position of a particle step [cm], the JPART number, charge (Z) and nucleon (A) numbers, the kinetic energy [MeV/nucleon] and the energy loss ΔE for this step [MeV/nucleon]. If the start and stop positions are equal, then a point-wise scoring happened (e.g. at the end of a particle trajectory).

Only one **TRACE** card is allowed within *detect.dat*.

Since a lot of output is generated which is written to disk, it will slowdown the calculation massively. It is thus not recommended to record more than 100 particle trajectories.

Arguments are:

1. first particle to be scored
2. last particle to be scored
3. unused.
4. unused.
5. unused.
6. *outputfile*: output file name. *.csv* suffix will be appended automatically.

Analogue to **CYL** and **MSH**, a continuation card is only required if **JPART=25** or if **DETECTOR** is specified as **LETFLU** (and only **LETFLU**). Arguments for the continuation card are:

1. *Z*: an integer specifying the charge of the particle to be scored.
2. *A*: an integer specifying the number of nucleons of the particle to be scored. Setting this value to -1 scores all isotopes.
3. **MEDIUM**: an integer specifying the medium for LET weighted scoring. If left unspecified, the default is the medium material that the bin is made of. Any other integer from the medium list specified in *mat.dat* is valid here. In that case, scoring will apply the stopping powers of the medium specified here. Transport will irrespective of the scoring always be done in the as medium specified in *geo.dat*. This is useful for calculating stopping power ratios needed by various cavity theories in radiation dosimetry.
4. unused.
5. unused.
6. unused.

3.6.8 VOXSCORE - Scoring within loaded CT-cube

VOXSCORE : Scoring within a voxelized structure, which is loaded with the **VOX** card in *geo.dat*. **VOX** functions similar to **MSH**, however the Cartesian mesh will always be aligned with the loaded CT-cube, automatically taking care of any translation and rotation of the CT-cube. Arguments are:

1. unused.
2. unused.
3. unused.
4. **JPART**: particle type. See list in section A.3. Setting this value to -1 scores all particles. If heavy ions are scored (**JPART**=25) then a continuation card is required, where the first argument is an integer representing the particle charge Z , and the second argument is an integer representing the particle mass A .
5. **DETECTOR**: quantity that should be scored. **Important:** Zone scoring works straightforwardly only with the detectors **ENERGY**, **CROSSFLU**, **DLET**, **COUNTER** and **PET** as described in section 3.6.9. All other detectors can be applied as well, but require knowledge of the zone volume, which is not known by **SHIELD-HIT12A** (unlike the **MSH** and **CYL** scorer). Therefore the user must divide the scoring result with the corresponding zone volume (in cm^3) for all other detectors than those listed above. This may be changed in a future release (see ticket #177).
6. *outputfile*: output file name.

Analogue to **CYL** and **MSH**, a continuation card is only required if **JPART**=25 or if **DETECTOR** is specified as **LETFLU** (and only **LETFLU**). Arguments for the continuation card are:

1. Z : an integer specifying the charge of the particle to be scored.
2. A : an integer specifying the number of nucleons of the particle to be scored. Setting this value to -1 scores all isotopes.
3. **MEDIUM**: an integer specifying the medium for LET weighted scoring. If left unspecified, the default is the medium material that the bin is made of. Any other integer from the medium list specified in *mat.dat* is valid here. In that case, scoring will apply the stopping powers of the medium specified here. Transport will irrespective of the scoring always be done in the as medium specified in *geo.dat*. This is useful for calculating stopping power ratios needed by various cavity theories in radiation dosimetry.
4. unused.
5. unused.
6. unused.

3.6.9 Available detectors for scoring

Different detector types are available for the auxiliary scoring. For each geometrical estimator invoked in the file `detect.dat` one type of detector has to be specified for `DETECTOR` in the second card. But not every geometrical estimator type can be combined with each of the available detectors. A detector type may itself invoke an additional card. The output is given in SHIELD-HIT12A standard units (see table A.1). The quantities that scale with particle number are normalized per simulated primary particle. The available detectors are:

ENERGY : Absolute energy deposited in each bin.

FLUENCE : Track length fluence Φ in each bin.

CROSSFLU : Absolute number of particles entering each bin.

FLU-CHAR : Charged particle track length fluence Φ_{charged} in each bin.

FLU-NEUT : Neutral particle track length fluence Φ_{neutral} in each bin.

DOSE : Energy deposited in each bin divided by total mass (volume \times density) of the bin.

LETFLU : Sum of the products between fluence and unrestricted linear energy transfer (LET) over particle type i ,

$$\bar{\Phi}_{\text{LET}} = \sum_i \phi_i(E, Z) \frac{dE}{dx}(E, Z)_i, \quad (3.1)$$

for a medium specified by card 3, argument 3 which might be different from the material of the bin. This argument is a number m that corresponds to the m^{th} material in *mat.dat*. The product for one specific particle i can be selected with arguments 1 and 2 in card 3. All particles are considered if these two arguments are left blank.

1MEVNEQ : 1 MeV equivalent neutron fluence, based on silicon. The tables used are based on reference [?]. So far, only protons, neutrons and ions are supported. Heavier ions will thus not contribute with dose from non-ionizing energy loss (NIEL). Note that these tables have no data point at exactly 1 MeV for neutrons. When interpolating, the normalized damage function $D(E) / 95 \text{ MeV mb}$ yields a factor of 1.05, instead of the expected 1.00.

DLET : Dose (D) averaged unrestricted stopping power (LET) in MeV/cm. Since SHIELD-HIT12A 0.7.0 the scoring method used, is the algorithm “C” (equation 12) mentioned in [?].¹⁰ Only charged particles are considered, since neutral particles have no electronic stopping power. For each particle i :

$$\overline{LET}_D = \frac{\sum_i D_i(E, Z) \frac{dE}{dx}(E, Z)_i}{\sum_i D_i(E, Z)}. \quad (3.2)$$

TLET : Track length (τ) - averaged unrestricted stopping power (LET) in MeV/cm. TLET Uses similar scoring method as for DLET, but for each charged particle i :

$$\overline{LET}_\tau = \frac{\sum_i \tau_i(E, Z) \frac{dE}{dx}(E, Z)_i}{\sum_i \tau_i(E, Z)}. \quad (3.3)$$

which equals the fluence (Φ) averaged unrestricted stopping power, since $\Phi = \tau/V$.

¹⁰Earlier versions of SHIELD-HIT12A employed the algorithm “A” from the same reference.

AVG-ENERGY : Track length averaged energy for charged and neutral particles. For each particle i :

$$\overline{E}_\tau = \frac{\sum_i E_i(Z) \tau_i(E, Z)}{\sum_i \tau_i(E, Z)}. \quad (3.4)$$

AVG-BETA : Track length averaged relativistic velocity $\beta = v/c$, applied to charged and neutral particles. For each particle i :

$$\overline{\beta}_\tau = \frac{\sum_i \beta_i(Z) \tau_i(E, Z)}{\sum_i \tau_i(E, Z)}. \quad (3.5)$$

Q : Armin's Q-value. Q is a radiation quality specifier similar to LET which is useful to express variations of observables such as the relative biological effectiveness (RBE) or relative effectiveness (RE or η) of detectors. Contrary to LET, the observables will be independent on particle charge, when expressed as a function of Q . The quantity is dose-averaged (D), and thus not defined in vacuum or for neutral particles. Only particles with a kinetic energy of 100 keV/nucleon or more will be scored. **Experimental implementation. Q is not fully defined yet, and may change in future releases.**

$$\overline{Q}_D = \frac{\sum_i 1/\rho(z_i^2/E) D_i(E, Z)}{\sum_i D_i(E, Z)}. \quad (3.6)$$

where z is the projectile charge, ρ is the material density, and E is the energy divided by the integer nucleon number (i.e. not divided by the atomic mass). Q is per definition dimensionless.

ALANINE : Response-equivalent dose for the alanine detector. The medium in which this detector is applied must be alanine (ICRU ID = 105) in order to make sense. For each charged particle i :

$$D_{\text{al,photon}} = \sum_i \eta(Z_i, E_i) D_{i,\text{alanine,ion}}. \quad (3.7)$$

The relative effectiveness η is calculated using an implementation [?] of the Hansen and Olsen model [?, ?] which only takes pions, kaons, and ions with $Z < 7$ into account. For heavier ions ($Z > 6$) η is set to 1.0. For energies below 1 keV/nucleon η is fixed to 0.3 for all ions.

COUNTER : A counter which counts the number of particles crossing a PLANE estimator.

3.6.10 Uncertainties

SHIELD-HIT12A does not record the sum of squares of the results in order to minimize the memory footprint of the program at runtime. Therefore the output from the detectors will not include any statement of the precision of the calculation.

Instead the user is encouraged to run the process several times in order to derive a best estimate of the precision. For n independent runs (i.e. with different random seeds), but with equal number of primaries and each with the result x_i , the best estimate of precision s_n can be obtained by

$$s_n = \frac{\sum (x_i - X_n)^2}{(n-1)^{1/2}} \quad (3.8)$$

where X_n is the mean of all obtained x_i . Note that the underlying distributions may not be normal distributed, so the estimate can be misleading.

3.6.11 Examples - Scoring

Example 1: 1D and 2D maps

This example shows how to score a 1-D energy deposition of all particles in a cylindrical volume which is 20 cm long (from 35 cm to 55 cm). The output will be written into the file *foo*. In addition, a 2-D fluence map ($1 \times 20 \times 300$ bins) of carbon-12 ions ($Z = 6$ and $A = 12$) is generated and written to the file *bar*. Finally, the average energy deposited by all particles per primary particle in the zones 2 to 12 defined in *geo.dat* is outputted to *foobarlongfilename*.

```
*----0----><----1----><----2----><----3----><----4----><----5----><----6---->
CYL          0.0      0.0      35.0      20.0      7.0      55.0
              1        1        300        -1      ENERGY      foo
MSH          0.0      0.0      0.0      10.0      10.0      10.0
              1        20       300        25      FLUENCE      bar
              6        12
ZONE         2        12              -1      ENERGY      foobarlongfilename
```

Example 2: Angular histogram

Here is an example, how to record the angular histogram of particles crossing a plane. A X-Y plane is placed at $Z = 10.0$ cm. Differential angle scoring is invoked, scoring all from 0.0 to 5.0 mrad in 20 bins.

```
*----0----><----1----><----2----><----3----><----4----><----5----><----6---->
DPLANE       0.0      0.0      10.0      0.0      0.0      1.0
              -1      COUNTER NB_count4
              0.0      0.005      20      ANGLE
```

Example 3: DOSE and LET in CT-cubes, with particle tracks

This example shows how to score the Dose and Dose-averaged LET inside a CT-cube, which was loaded using the *VOX* card in *geo.dat*. A *TRACE* card is added, to visualize a few of the simulated transport events in the CT-cube.

```
*----0----><----1----><----2----><----3----><----4----><----5----><----6---->
VOXSCORE          -1      ENERGY      myvox01
*----0----><----1----><----2----><----3----><----4----><----5----><----6---->
VOXSCORE          -1      DOSE      myvox02
VOXSCORE          -1      DLET      myvox03
VOXSCORE          1      FLUENCE      myvox04
*----0----><----1----><----2----><----3----><----4----><----5----><----6---->
TRACE             1        1        0        0        0      trace
```

Note, that for a $512 \times 512 \times 172$ cube, this example allocates order of 4 GB of RAM for scoring. While this is normally not a problem in single-threaded mode, in parallel mode you may want to make sure that you have sufficient RAM available in order to prevent SHIELD-HIT12A to use swap space.

Chapter 4

Output files

SHIELD-HIT12A provides several output files, depending on the configuration in *geo.dat* and *detect.dat*.

Programs to convert the binary *.*bdo* files to human readable ASCII exist, where `pymchelper` is recommended to use [?].

See <https://pymchelper.readthedocs.io/>

Finally a run-specific logfile is created *shieldhit.log* which will cover all details of the run.

4.1 Auxiliary output files

These files are not intended for further processing. They contain auxiliary information for the user in the manner of man-readable ASCII coding. There are no scripts for handling these files in an parallelized environment.

for017 : This file is created and used by the geometry parser. If parsing of *geo.dat* failed you may find the file *for017* which may contain additional diagnostics. Else this file can safely be deleted.

shieldhit.log : Contains input and output data of the current run. It is recommended to keep this file, if you need your run to be documented. Also found here are the stopping powers used as well as dependencies range-energy, macroscopic cross sections and other useful information for all particles/fragments and all materials in the task. Also found here are multigroup neutron cross sections below 14.5 MeV for all chemical elements used within this run.

for028 : This is an optional file which can be requested by the `MAKELN` card in *beam.dat*. It contains a list of secondary neutrons below 14.5 MeV with all individual parameters: the birth point XYZ, direction of fly, kinetic energy. This file can be used as a neutron source for transportation with MCNP [?] like programs.

Chapter 5

Auxiliary programs

5.1 Helper scripts

5.2 Format conversion

5.2.1 *convertmc* - Convert *.bdo* files to ASCII or similar

convertmc converts the output of an estimator specified in *detect.dat* (see section 3.5). If run in a parallel environment, and multiple outputs exist of the same estimator, then *convertmc* will average over all files specified in the argument and estimate the error. *convertmc* is the most important script for postprocessing, and is not provided by the SHIELD-HIT12A installation, but is provided by the *pymchelper* package [?]. The user is referred to <https://github.com/DataMedSci/pymchelper> where this package is documented. For installation, make sure *pip* is installed (`sudo apt-get install pip` on Ubuntu/Debian systems) and run

```
$ pip install pymchelper
```

5.2.2 *shield2fluka* - Convert *geo.dat* to FLUKA format

shield2fluka converts the geometry file to a FLUKA formatted input file. Only the geometry file is converted, not the beam settings or scoring. The produced FLUKA formatted *output.inp* cannot be run in FLUKA, but it can be read by “SimpleGeo” which is a windows program for viewing geometries in 3D.

See <http://theis.web.cern.ch/theis/simplegeo/>. *shield2fluka* is about to be obsolete, and the user is instead referred to *pymchelper* <https://github.com/DataMedSci/pymchelper>, which is more complete than *shield2fluka*.

5.2.3 *fluka2shield* - Convert FLUKA *.inp* files to SHIELD-HIT12A input files

fluka2shield converts a FLUKA formatted input file to SHIELD-HIT12A input files. The resulting files are by no means complete, and must be manually edited by the user, however it may still serve as a help for converting more complex geometries.

fluka2shield is about to be obsolete, and the user is instead referred to *pymchelper* <https://github.com/DataMedSci/pymchelper>, which is more complete than *fluka2shield*.

5.2.4 *shield_dEdx* - Prepare an external stopping power table

shield_dEdx generates a properly formatted stopping power table which can be read by SHIELD-HIT12A. *shield_dEdx* requires the *libdedx* software package installed [?, ?], which can be downloaded for free at <https://github.com/APTG/libdedx>.

The *libdEdx* library features access to a range of alternative stopping power tables and a comprehensive list of target materials. An output file is generated according to the nomenclature found in table A.4. The generated ASCII file is human readable and contains a header with information on creation date, version number etc.

Example of usage:

```
$ ./processing/dEdx/shield_dEdx 1 223
Preparing stopping power table for PMMA using PSTAR + MSTAR.
Wrote Lucite.txt.
```


Chapter 6

Parallelization

SHIELD-HIT12A can run in a parallelized environment in several ways, ranging from single PC which features multiple cores, or massive computer clusters. SHIELD-HIT12A runs on CPUs only, currently no GPU code is implemented. In this chapter several ways of parallelizing SHIELD-HIT12A is explained. SHIELD-HIT12A will here run in “embarrassingly parallel” mode, where simple independent jobs are submitted to various cores.

Each independent job needs it’s own random number seed. The command line option `-N` can be used to add an offset in form of an integer value to the `RNDSEED` card specified in the *beam.dat* file. If `RNDSEED` was not specified in the *beam.dat* the offset is simply added to the default seed. In other words, assuming a project is setup in the *example/* directory, the command

```
$ shieldhit example/ -N17
```

will add 17 to the `RNDSEED` value specified in *example/beam.dat* and run SHIELD-HIT12A with this new seed.

All output results *example/*.bdo* will carry a four digit zero-padded string with the given offset. That is, following the aforementioned example, in the form of *example/*0017.bdo*. Beware that the offset specified by the `-N` option is limited to 9999 and negative values are not accepted.

Example scripts for parallelization are only shipped with the Linux distribution. These are *rtshield.sh* for Torque systems, *rbshield.sh* for Platform LSF based systems and *rcshield.py* for Condor based systems.

6.1 Torque

Torque is a widely used system for job submission across multiple nodes. Here a very simple tutorial is given for using multiple nodes on a PC running SHIELD-HIT12A processes. It is assumed that Torque and the PBS scheduler is running and correctly configured. Furthermore it is assumed that SHIELD-HIT12A was fully installed.

A simple PBS submission script *rtshield.sh* can look like this:

```
#!/bin/sh
#
#PBS -N SHIELDHIT_JOB
#PBS -M bassler@phys.au.dk
#
shieldhit $1 -N$PBS_ARRAYID
```

Simply create this file in the directory you wish to run. Step into the same directory and run:

```
$ qsub -V -t 0-9 -d . rtshield.sh
```

The `-V` forwards the local environment variables to the submission script. This way we do not need to specify the path of the `shieldhit` command, since it is referred to by the `PATH` environment variable.

The `-t` option specifies the nodes to be run, in this case from 0 to 9, meaning 10 instances in total. The output files will be suffixed accordingly, so they will not overwrite each other.

Finally, the `-d .` option specifies the working directory. `qsub` expects to find all the input files as well as the `rtshield.sh` script here.

You can check the status of your job submission with `qstat`. A good manual on Torque can be found on <http://www.adaptivecomputing.com/support/documentation-index/torque-resource-man>

After the run completes, each output file (which can be merged by the `convertmc` script) will be appended with an integer number corresponding to the instance number. Thus in the example above, the output files will be suffixed with integers ranging from 0000 to 0009.

6.2 LSF

LSF system is quite similar to torque. You will have to adapt the example script `rbshield.sh` to your local environment, and then do the submission by e.g:

```
$ bsub -q test -J jobname[1-2] < ./rbshield.sh
```

which will run two jobs in the `test` queue.

6.3 Condor

Another job submission and scheduling system is CONDOR¹. Here submission can be handled by the `rcshield.py` script.

```
$ rcshield.py -c -M100 examples/simple
```

will run the job found in the `examples/simple` directory 100 times. All results, debugging output is transferred back to the directory, and each result file is suffixed with a number in order to distinguish the results. `rcshield.py` accepts a range of options:

-h - print all options with a short description.

-f - should be followed by a comma separated (without space) file list, which will be transmitted along with the usual mandatory files. The usual input files which always will be transmitted are: `mat.dat geo.dat beam.dat Air.txt Water.txt detect.dat`. Additional files could be a ripple filter file `rifi.dat`, source or a files which overrides the default model parameters (see section 3.3.3).

-M - number of simulations to be performed. Random seed is automatically changed for each run.

-N - first run will have the number followed by this option. Useful if you want to add more statistics to a previous batch or runs. If the last completed run had number 99, you should set `-N100`.

¹<http://research.cs.wisc.edu/condor/>

-
- a** - after each batch of runs, *rcfluka.py* will wait 30 seconds, for any file transfers to complete. This grace time is necessary if you run any postprocessing scripts or the **-c** option. For small runs and quick testing, the 30 secs may be annoying, and can therefore be deactivated with the **-a** option.
 - n** - the jobs are submitted as nice jobs to the cluster, backfilling it if no other tasks are pending. Use this option if you are not in a hurry to get your data.
 - c** - clean up some of the condor files which were generated, which are only of interest if something goes wrong. Purpose of this option is to keep your run directory tidy.
 - C** - big clean: use this if you want to start from scratch in your directory. Nothing is submitted to the cluster, but the directory is emptied for any data generated by previous runs. DANGEROUS, files which may resemble generated data are deleted as well. Basically anything which matches *_rcshield**, **[0-9][0-9][0-9][0-9]*, *for[0-9][0-9][0-9]* and *fort.24* are removed.
 - t** - test script, but do not submit jobs to condor cluster.
 - S** - use standard universe instead of vanilla (see condor manual). Standard universe will only work if you have linked in the appropriate condor libraries for checkpointing into the SHIELD-HIT12A executable.
 - o** - followed by a string which will be added to the condor submit file.
 - i** - followed by a filename of a file which will be appended to the condor submit script.
 - m** - followed by an email address, which will be notified when all jobs are completed (or failed).
 - x** - single submission mode. This overrides **-M** and **-m** option. Simply one job is submitted, and *rcshield.py* immediately exits after submission. This option is useful if you just want the raw output from a single run, and dont want the *rcshield.py* script block the terminal while running. Entire control is within condor, which will transfer files upon job completion.

Appendix A

User's reference tables and lists

A.1 Command line options and arguments

SYNOPSIS

```
shieldhit [OPTIONS]... [WORKDIR]
```

DESCRIPTION

The `shieldhit` command takes exactly one optional argument, and that is the working directory. All input files are expected to be placed in the working directory, and all output files will be placed in the same directory. Additional options may be given, as described below.

OPTIONS

- h, --help Print all available options and exit.
- V, --version Print version and exit.
- v, --verbose Be verbose.
- n, --nstat=NUM Override NSTAT using NUM amount of particles instead. Useful for testing with short runs or dry runs, -n0.
- t, --time="HH:MM:SS" Set maximum total simulation time in HH:MM:SS (including initialization).
- N, --seedoffset=NUM Offset added to RNDSEED specified in *beam.dat*. Useful for parallelization, as described in chapter 6.
- b, --beamfile=**FILE** Load *FILE* instead of *WORKDIR/beam.dat*.
- g, --geofile=**FILE** Load *FILE* instead of *WORKDIR/geo.dat*.
- m, --matfile=**FILE** Load *FILE* instead of *WORKDIR/mat.dat*.
- d, --detectfile=**FILE** Load *FILE* instead of *WORKDIR/detect.dat*.

SIGNALS

SIGUSR1 (Linux only) Saves current results.

SIGINT Saves current results and exits.

A.2 Units

Table A.1 provides an overview of the standard units in SHIELD-HIT12A. They apply at any time unless explicitly stated otherwise.

Table A.1: Table of default units in SHIELD-HIT12A.

Value	Default unit
Distance; position	cm
Track length density; fluence	cm^{-2}
Volume	cm^3
Density	g/cm^3
Atomic density	atoms/cm^3
Projectile energy (ions)	$\text{MeV}/\text{nucleon}$
Projectile energy (non-ions)	MeV
Deposited energy	MeV
Dose	MeV/g
Stopping power; LET	MeV/cm
Mass stopping power	$\text{MeV cm}^2/\text{g}$
Mean excitation potential	eV

A.3 Particle codes

Table A.2: List of available particle identifiers **JPART** and the corresponding particle names.

JPART	Particle	JPART	Particle
-1	All particles	13	Electron
1	Neutron	14	Positron
2	Proton	15	Muon μ^-
3	Pion π^-	16	Muon μ^+
4	Pion π^+	17	e^- -Neutrino ν_e
5	Pion π^0	18	e^- -Anti-neutrino $\bar{\nu}_e$
6	Anti-neutron	19	μ^- -Neutrino ν_μ
7	Anti-proton	20	μ^- -Anti-neutrino $\bar{\nu}_\mu$
8	Kaon κ^-	21	Deuteron
9	Kaon κ^+	22	Triton
10	Kaon κ^0	23	He-3
11	Kaon κ^\sim	24	He-4
12	Gamma ray	25	Heavy ions

Note, the generalized particle **JPART**=-1 is only available within a scoring estimator, and not as a particle source. For a heavy ion **JPART**=25 the charge Z and atomic number A have usually to be defined additionally. As particle source, $Z > 2$ is expected.

A.4 Nuclear targets with neutron cross sections

Table A.3 shows a list of elements and isotopes for which neutron cross sections are included in SHIELD-HIT12A. Natural isotope mixtures are marked with a '*'.

Table A.3: List of nuclear target identifiers with available neutron cross sections NUCLID and the corresponding names used by SHIELD-HIT12A.

<i>Z</i>	NUCLID	Isotope	Reference
1	1	H-1 - Hydrogen	JEFF-3.1
1	101	H-2 - Deuterium	BNAB-81
1	102	H-3 - Tritium	BROND
2	104	He-3	BNAB-81
2	2	He-4	BNAB-81
3	105	Li-6	BNAB-81
3	3	Li-7	BNAB-81
4	4	Be-9	ENDFB7.1
5	106	B-10	BNAB-81
5	5	B-11	BNAB-81
6	6	C-*	BNAB-81
7	7	N-*	BNAB-81
8	8	O-*	BNAB-81
9	9	F-19	BROND-2
11	11	Na-23	BNAB-81
12	12	Mg-*	CENDL-2
13	13	Al-27	BNAB-81
14	14	Si-*	BNAB-81
15	15	P-31	BROND-2
16	16	S-*	JEFF-3
17	17	Cl-*	BROND-2
18	18	Ar-*	JENDL40
19	19	K-*	CENDL-2
20	20	Ca-*	BNAB-81
22	22	Ti-*	JEFF-3
23	23	V-51	CENDL-2
24	24	Cr-*	BNAB-81
25	25	Mn-55	BNAB-81
26	26	Fe-*	BNAB-81
27	27	Co-59	ANL/NDM
28	28	Ni-*	BNAB-81
29	29	Cu-*	BROND-2
30	30	Zn-*	BROND-2
31	31	Ga-*	JENDL
32	32	Ge-*	JEFF-3.1
33	33	As-75	JENDL
41	41	Nb-93	CENDL-2

Continued on Next Page...

Table A.3 – Continued

<i>Z</i>	NUCLID	Isotope	Reference
42	42	Mo-*	JENDL
47	47	Ag-*	JENDL
48	48	Cd-*	BNAB-81
50	50	Sn-*	CENDL-2
63	63	Eu-*	BNAB-81
64	64	Gd-*	BNAB-81
68	68	Er-*	BNAB-81
73	73	Ta-181	JEFF
74	74	W-*	CENDL31
75	75	Re-*	JEFF-3.1
79	79	Au-187	BROND-2
80	80	Hg-*	JEFF
82	82	Pb-*	BNAB-81
83	83	Bi-209	ENDFB7.1
90	90	Th-232	ENDFB-6
92	103	U-235	BNAB-81
92	92	U-238	BNAB-81
94	94	Pu-239	BNAB-81
94	107	Pu-240	BNAB-81

A.5 Material ICRU_ID

Table A.4 lists all material numbers as defined by ICRU which are known to SHIELD-HIT12A. The ICRU_ID number can optionally be specified in the material input file *mat.dat* which then will trigger SHIELD-HIT12A to load stopping power data from an external ASCII file (see section 3.2). The file containing the stopping power data *must* exactly be named as the according file name in the table below.

Table A.4: List of ICRU_ID and file names used by SHIELD-HIT12A.

ICRU_ID	File name	Material name and remarks
1	H.txt	HYDROGEN
2	He.txt	HELIUM
3	Li.txt	LITHIUM
4	Be.txt	BERYLLIUM
5	B.txt	BORON
6	C.txt	AMORPHOUS CARBON (density 2.0 g/cm ³)
906	Graphite.txt	GRAPHITE
7	N.txt	NITROGEN
8	O.txt	OXYGEN
(...)	(...)	(...) ¹
98	Cf.txt	CALIFORNIUM
99	A-150.txt	A-150 TISSUE-EQUIVALENT PLASTIC
100	Acetone.txt	ACETONE
101	Acetylene.txt	ACETYLENE
102	Adenine.txt	ADENINE
103	Adipose.txt	ADIPOSE TISSUE (ICRP)
104	Air.txt	AIR, DRY (NEAR SEA LEVEL)
105	Alanine.txt	ALANINE
106	Al2O3.txt	ALUMINUM OXIDE
107	Amber.txt	AMBER
108	Ammonia.txt	AMMONIA
109	Aniline.txt	ANILINE
110	Anthracene.txt	ANTHRACENE
111	B-100.txt	B100
112	Bakelite.txt	BAKELITE
113	BaF2.txt	BARIUM FLUORIDE
114	BaSO4.txt	BARIUM SULFATE
115	Benzene.txt	BENZENE
116	BeO.txt	BERYLLIUM OXIDE
117	BiGeO.txt	BISMUTH GERMANIUM OXIDE
118	BloodICRP.txt	BLOOD (ICRP)
119	BoneICRU.txt	BONE, COMPACT (ICRU)
120	BoneICRP.txt	BONE, CORTICAL (ICRP)
121	B4C.txt	BORON CARBIDE
122	BoronOxide.txt	BORON OXIDE
123	BrainICRP.txt	BRAIN (ICRP)
124	Butane.txt	BUTANE

Continued on Next Page...

¹All elements up to Californium ($Z = 98$) are supported following this naming scheme.

Table A.4 – Continued

ICRU_ID	File name	Material name and remarks
125	N-ButylAlcohol.txt	N-BUTYLALCOHOL
126	C-552.txt	C-552 AIR-EQUIVALENT PLASTIC
127	CdTe.txt	CADMIUM TELLURIDE
128	CdWO4.txt	CADMIUM TUNGSTATE
129	CaCO3.txt	CALCIUM CARBONATE
130	CaF2.txt	CALCIUM FLUORIDE
131	CaO.txt	CALCIUM OXIDE
132	CaSO4.txt	CALCIUM SULFATE
133	CaWO4.txt	CALCIUM TUNGSTATE
134	CO2.txt	CARBON DIOXIDE
135	CCl4.txt	CARBON TETRACHLORIDE
136	Cellophane.txt	CELLULOSE ACETATE, CELLOPHANE
137	CAB.txt	CELLULOSE ACETATE BUTYRATE
138	CeINitrate.txt	CELLULOSE NITRATE
139	CeSO4Dos.txt	CERIC SULFATE DOSIMETER SOLUTION
140	CsF.txt	CESIUM FLUORIDE
141	CsI.txt	CESIUM IODIDE
142	ClBenzene.txt	CHLOROBENZENE
143	CHCl3.txt	CHLOROFORM
144	Concrete.txt	CONCRETE PORTLAND
145	Cyclohexane.txt	CYCLOHEXANE
146	DClBenzene.txt	1,2-DICHLOROBENZENE
147	DClDEtyEth.txt	DICHLORODIETHYL ETHER
148	DClEthane.txt	DICHLOROETHANE
149	DEtyEther.txt	DIETHYLEETHER
150	DMF.txt	N,N-DIMETHYL FORMAMIDE
151	DMSO.txt	DIMETHYLSULFOXIDE
152	C2H6.txt	ETHANE
153	Ethanol.txt	ETHYL ALCOHOL
154	EthylCell.txt	ETHYL CELLULOSE
155	C2H4.txt	ETHYLENE
156	Eyelens.txt	EYELENS (ICRP)
157	Fe2O3.txt	FERRIC OXIDE
158	FeB.txt	FERRO BORIDE
159	FeO.txt	FERROUS OXIDE
160	FeSO4Dos.txt	FERROUS SULFATE DOSIMETER SOLUTION
161	Freon-12.txt	FREON-12
162	Freon-12B2.txt	FREON-12B2
163	Freon-13.txt	FREON-13
164	Freon-13B1.txt	FREON-13B1
165	Freon-13I1.txt	FREON-13I1
166	GOS.txt	GADOLINIUM OXYSULFIDE
167	GaAs.txt	GALLIUM ARSENIDE
168	GelPhot.txt	GEL IN PHOTOGRAPHIC EMULSION
169	GlassPyrex.txt	GLASS, PYREX
170	GlassLead.txt	GLASS, LEAD
171	GlassPlate.txt	GLASS, PLATE
172	Glucose.txt	GLUCOSE
173	Glutamine.txt	GLUTAMINE
174	Glycerol.txt	GLYCEROL

Continued on Next Page...

Table A.4 – Continued

ICRU_ID	File name	Material name and remarks
175	Guanine.txt	GUANINE
176	Gypsum.txt	GYPSUM / PLASTER OF PARIS
177	Heptane.txt	N-HEPTANE
178	Hexane.txt	N-HEXANE
179	Kapton.txt	KAPTON POLYIMIDE FILM
180	LaBrO.txt	LANTHANUM OXYBROMIDE
181	La2O2S.txt	LANTHANUM OXYSULFIDE
182	PbO.txt	LEADOXIDE
183	LiNH2.txt	LITHIUM AMIDE
184	Li2CO3.txt	LITHIUM CARBONATE
185	LiF.txt	LITHIUM FLUORIDE
186	LiH.txt	LITHIUM HYDRIDE
187	LiI.txt	LITHIUM IODIDE
188	Li2O.txt	LITHIUM OXIDE
189	Li2B4O7.txt	LITHIUM TETRABORATE
190	Lung.txt	LUNG (ICRP)
191	M3wax.txt	M3 WAX
192	MgCO3.txt	MAGNESIUM CARBONATE
193	MgF2.txt	MAGNESIUM FLUORIDE
194	MgO.txt	MAGNESIUM OXIDE
195	MgB4O7.txt	MAGNESIUM TETRABORATE
196	HgI2.txt	MERCURIC IODIDE
197	CH4.txt	METHANE
198	Methanol.txt	METHANOL
199	MixDwax.txt	MIX D WAX
200	MS20.txt	MS20 TISSUE SUBSTITUTE
201	MuscleICRP.txt	MUSCLE, SKELETAL (ICRP)
202	MuscleICRU.txt	MUSCLE, STRIATED (ICRU)
203	MuscleSuc.txt	MUSCLE EQUIVALENT LIQUID, WITH SUCROSE
204	MuscleNoSuc.txt	MUSCLE EQUIVALENT LIQUID, NO SUCROSE
205	Naphthalene.txt	NAPHTHALENE
206	NitroBenz.txt	NITROBENZENE
207	N2O.txt	NITROUS OXIDE
208	Elvamide.txt	NYLON, DU PONT ELVAMIDE 8062
209	Nylon.txt	NYLON, TYPE 6 AND 6/6
210	Nylon6-10.txt	NYLON, TYPE 6/10
211	Rilsan.txt	NYLON, TYPE 11 (RILSAN)
212	Octane.txt	OCTANE, LIQUID
213	Paraffin.txt	PARAFFINWAX
214	Pentane.txt	N-PENTANE
215	Emulsion.txt	PHOTOGRAPHIC EMULSION
216	Plasscin.txt	PLASTIC SCINTILLATOR (VINYLTOLENE BASED)
217	PuO2.txt	PLUTONIUM DIOXIDE
218	PAN.txt	POLYACRYLONITRILE
219	Polycarb.txt	POLYCARBONATE (MAKROLON, LEXAN)
220	Polyclst.txt	POLYCHLOROSTYRENE
221	Polyeth.txt	POLYETHYLENE
222	Mylar.txt	POLYETHYLENE TEREPHTHALATE (MYLAR)
223	Lucite.txt	POLYMETHYL METHACRALATE (LUCITE, PERSPEX, PMMA)
224	POM.txt	POLYOXYMETHYLENE

Continued on Next Page...

Table A.4 – Continued

ICRU_ID	File name	Material name and remarks
225	PP.txt	POLYPROPYLENE
226	PS.txt	POLYSTYRENE
227	Teflon.txt	POLYTETRAFLUOROETHYLENE (TEFLON)
228	PTFCE.txt	POLYTRIFLUOROCHLOROETHYLENE
229	PVAc.txt	POLYVINYL ACETATE
230	PVOH.txt	POLYVINYL ALCOHOL
231	PVB.txt	POLYVINYL BUTYRAL
232	PVC.txt	POLYVINYL CHLORIDE
233	Saran.txt	SARAN
234	PVDF.txt	POLYVINYLIDENE FLUORIDE
235	PVP.txt	POLYVINYLPYRROLIDONE
236	KI.txt	POTASSIUM IODIDE
237	K2O.txt	POTASSIUM OXIDE
238	Propane.txt	PROPANE
239	PropaneLiq.txt	PROPANE, LIQUID
240	NPropylOH.txt	N-PROPYL ALCOHOL
241	Pyridine.txt	PYRIDINE
242	RubberB.txt	RUBBER, BUTYL
243	Rubber.txt	RUBBER, NATURAL
244	Neoprene.txt	RUBBER, NEOPRENE
245	SiO2.txt	SILICON DIOXIDE
246	AgBr.txt	SILVER BROMIDE
247	AgCl.txt	SILVER CHLORIDE
248	AgHaEmul.txt	SILVER HALIDES IN PHOTOGRAPHIC EMULSION
249	AgI.txt	SILVER IODIDE
250	Skin.txt	SKIN (ICRP)
251	Na2CO3.txt	SODIUM CARBONATE
252	NaI.txt	SODIUM IODIDE
253	Na2O.txt	SODIUM MONOXIDE
254	NaNO3.txt	SODIUM NITRATE
255	Stilbene.txt	STILBENE
256	Sucrose.txt	SUCROSE
257	Terphenyl.txt	TERPHENYL
258	Testes.txt	TESTES (ICRP)
259	C2Cl4.txt	TETRACHLOROETHYLENE
260	TlCl.txt	THALLIUM CHLORIDE
261	TissueICRP.txt	TISSUE, SOFT (ICRP)
262	TissueICRU.txt	TISSUE, SOFT (ICRU, FOUR COMPONENT)
263	Tissmeth.txt	TISSUE-EQUIVALENT GAS (METHANE BASED)
264	Tissprop.txt	TISSUE-EQUIVALENT GAS (PROPANE BASED)
265	TiO2.txt	TITANIUM DIOXIDE
266	Toluene.txt	TOLUENE
267	Trichlor.txt	TRICHLOROETHYLENE
268	Triethphos.txt	TRIETHYL PHOSPHATE
269	WF6.txt	TUNGSTEN HEXAFLUORIDE
270	UC2.txt	URANIUM DICARBIDE
271	UC.txt	URANIUM MONOCARBIDE
272	UO.txt	URANIUM OXIDE
273	Urea.txt	UREA
274	Valine.txt	VALINE

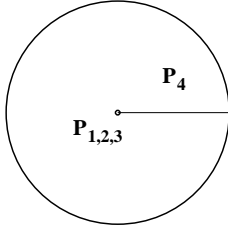
Continued on Next Page...

Table A.4 – Continued

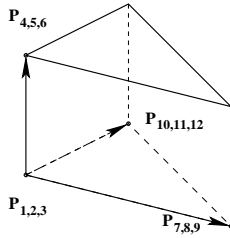
ICRU_ID	File name	Material name and remarks
275	Viton.txt	VITON FLUOROELASTOMER
276	Water.txt	WATER, LIQUID
277	H2Ovapor.txt	WATER VAPOR
278	Xylene.txt	XYLENE

A.6 Bodies

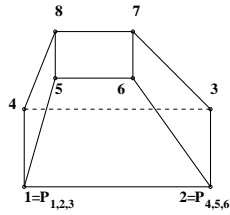
Each body is described with an identifier, that is a three-letter code preceded by two empty spaces. Another integer assigns a number to the body in the next 5 columns. Then 6 blocks each 10 columns wide specify the arguments P_1 to P_6 . If additional arguments are required, a second card is added holding the arguments P_7 to P_{12} , but leaving the first 10 columns empty. Even further cards follow the format of the second card. The arguments P_i can specify a point, a vector or a radius, depending on the body described. The bodies available in SHIELD-HIT12A are listed below:



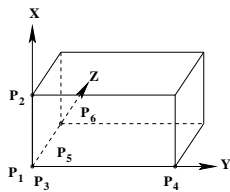
SPH : A sphere defined by the center coordinates (P_1, P_2, P_3) and the radius P_4 . The remaining arguments P_5 and P_6 are ignored.



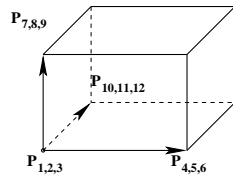
WED : A wedge spanned by three vectors. The origin is the point (P_1, P_2, P_3) from where the three vectors (P_4, P_5, P_6) , (P_7, P_8, P_9) and (P_{10}, P_{11}, P_{12}) start.



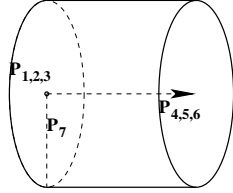
ARB : Arbitrary convex polyhedron. This body is described by 8 points, i.e. 24 arguments. The points must be specified in the order as indicated by the figure to the left: Point 1 is (P_1, P_2, P_3) , point 2 (P_4, P_5, P_6) ... point 8 (P_{22}, P_{23}, P_{24}) .



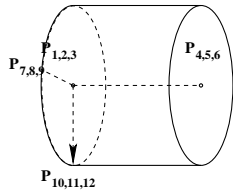
RPP : Rectangular parallelepiped. (P_1, P_2) marks minimum and maximum X coordinates. (P_3, P_4) is min and max values for Y, and (P_5, P_6) for Z.



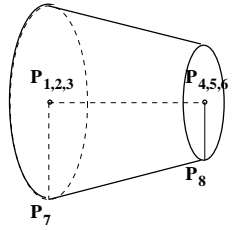
BOX : On the image there are given coordinates of selected corners of the box. A box is spanned by 3 orthogonal vectors. All vectors start in the corner point (P_1, P_2, P_3) . The vectors point to the corners (P_4, P_5, P_6) , (P_7, P_8, P_9) and (P_{10}, P_{11}, P_{12}) . What goes into the geo.dat line are the coordinate of the corner (P_1, P_2, P_3) and components of all the vectors, i.e. $(P_4 - P_1, P_5 - P_2, P_6 - P_3)$ etc...



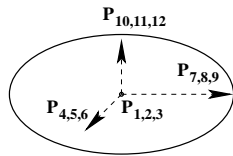
RCC : Right circular cylinder. One end of the cylinder is described by a circle with the center at (P_1, P_2, P_3) . The arguments (P_4, P_5, P_6) hold a vector measured from the center to the opposite end of the cylinder. P_7 marks the radius of the cylinder.



REC : Right elliptical cylinder. Similar to RCC, but instead of a radius, two additional vectors (P_7, P_8, P_9) and (P_{10}, P_{11}, P_{12}) describe the minor and major axis. All three vectors must be orthogonal to each other.



TRC : A cone spanned by two circles. First circle is centered at point (P_1, P_2, P_3) and has radius P_7 . The other end of the cone is described by the circle centered at the end of a vector (P_4, P_5, P_6) starting from the center of the first circle. The second circle has a radius of P_8 . The radius in P_7 must be larger than the radius in P_8 .



ELL : Ellipsoid. Centered at point (P_1, P_2, P_3) , three orthogonal vectors (P_4, P_5, P_6) , (P_7, P_8, P_9) and (P_{10}, P_{11}, P_{12}) span the body. *This body is differently defined than in FLUKA.*

VOX : Prepares a voxelized structure in shape of a box. (P_1, P_2, P_3) marks the SHIELD-HIT12A universe $(0, 0, 0)$ position inside the CT-cube. The (P_1, P_2, P_3) coordinates are therefore in terms of the CT-cube universe, and all are in [cm]. P_4, P_5 are the couch and gantry angles, respectively.

A continuation card is needed for specifying the filename to be loaded. The filename must be formatted in VOXELPLAN format, used also by the treatment planning system TRiP98. SHIELD-HIT12A will read both the header and the Hounsfield unit data cube, having the '.hed' and '.ctx' suffixes, respectively. Therefore, no suffix must be specified when loading the cube. See also section 3.4.3.

MOV : Copies a body number P_1 to a new position translated by the vector (P_2, P_3, P_4) . The columns 6-10 in the MOV card must hold an integer specifying a new and unique body number,

just like a normal body specification.

A.7 Input and output files.

Table A.5 provides an overview of input (I) and output (O) files used by SHIELD-HIT12A at run time. They have either ASCII (A) or binary (B) format. Some files are optional.

Table A.5: List of input and output files used at run time.

Filename	I/O	ASCII/Bin	Description	Remarks
<i>beam.dat</i>	I	A	Seed, projectile, stats, etc.	
<i>detect.dat</i>	I	A	Auxiliary scoring	1)
<i>geo.dat</i>	I	A	Geometry description	
<i>mat.dat</i>	I	A	Target medium chemical composition	
<i>*.bdo</i>	O	B	Result of scoring	1), 2)
<i>shieldhit.log</i>	O	A	Logfile	
<i>for017</i>	O	A	GEMCA parser log	
<i>for028</i>	O	A	Secondary neutrons production < 14.5 MeV enabled with MAKLN card	
<i>parlev.dat</i>	I	A	External parameter file for PARLEV parameters	1), 3)
<i>rifi.dat</i>	I	A	Description of the ripple filter	1), 3)
<i>sobp.dat</i>	I	A	External particle source file	1), 3)

¹⁾ Optional.

²⁾ File name is not predefined but set by user in *detect.dat*.

³⁾ File name is not predefined but set by user in *beam.dat*.

A.8 PARLEV parameters

The PARLEV parameters listed in table A.6 can be used for tuning the nuclear models in SHIELD-HIT. Up to 40 numeric parameters are set in SHIELD-HIT12A which may affect the calculated results. The default values are carefully selected and benchmarked against experimental data, and should therefore not be changed. However, for research purposes they can be changed using an external input file as described in section 3.3.3. In this case, a message is displayed for all modified settings in the output file *for024*.

Values which are marked "for high energies" in the table below, are only used at high energies (well above 1 GeV/A) and will therefore not influence calculations in normal particle radiotherapy settings.

A short description of the individual PARLEV parameters is given below. Details on the individual nuclear fragmentation models are given in the references [?, ?, ?, ?, ?, ?, ?, ?, ?].

PARLEV(1) - PIDABS : The probability of absorption of pion in nucleus by the quasi-deuteron pair.

PARLEV(2) - TLIMIT : Time of simulation of the cascade in nucleus (does not affect the strong decays of the resonances).

PARLEV(3) - MMES : The time of formation of mesons in their rest frame: $\tau = 1/(5.06 \cdot \text{MMES})$ [fm/c].

PARLEV(4) - MBAR : The time of formation of baryons in their rest frame: $\tau = 1/(5.06 \cdot \text{MBAR})$ [fm/c].

PARLEV(5) - EPS1, EPS2 : Separation energy of a nucleon from the nucleus to the cascade stage of the nuclear reaction. The energy is the same for the target nucleus (EPS2) and the projectile (EPS1).

PARLEV(6) - VP : The depth of the potential well for pion inside the nucleus.

PARLEV(7) - C1, C2 : The parameter of diffuseness of the Saxon-Woods distribution of nuclear density $\rho_{\text{WS}}(r) = \rho_0/[1 + \exp(r - R)/c]$, $R = r_0 A^{1/3}$. The parameter is the same for the target nucleus (C2) and the projectile (C1).

PARLEV(8) - D1, D2 : The parameter that determines the maximal radius R_{max} of the distribution of nuclear density $\rho(r)$ from the condition $\rho(R_{\text{max}})/\rho(0) = D$. This parameter is used for both the Saxon-Woods distribution $\rho_{\text{WS}}(r)$ and for light nuclei: $\rho(r) = \rho_0 \exp(-\alpha r^2)$. The parameter is the same for the target nucleus (D2) and the projectile (D1).

PARLEV(9) - RON1, RON2 : The parameter r_0 , which determines the radius of the nucleus by the formula $R = r_0 A^{1/3}$. For light nuclei this parameter directly sets the radius of nucleus. The parameter is the same for the target nucleus (RON2) and the projectile (RON1). PARLEV(9) is applicable to all isotopes, except for those specified in PARLEV(10) – PARLEV(21).

PARLEV(10) – PARLEV(21) : Individual values of RON1, RON2 for light nuclei. SHIELD-HIT12A does not discriminate between RON1 and RON2. In some cases groups of isotopes are affected. For instance, PARLEV(20) affects all nitrogen isotopes with nucleon number of 10 or less. Any isotope which are not affected by PARLEV(10) – PARLEV(21) are treated with PARLEV(9).

- PARLEV(22) - PRECOM : This parameter determines the contribution of pre-equilibrium emission of the lightest nuclei during transition from the cascade stage of nuclear reactions toward the equilibrium de-excitation ($0 \leq \text{PRECOM} \leq 1$).
- PARLEV(23) - ICAN : This parameter determines the number of evaporation channels at equilibrium deexcitation. If ICAN = 0 then only 6 channels are evaporated (n,p,d,t,He³,He⁴+fission). This mode should be used when mainly dealing with Fermi-Breakup of lighter nuclei. If ICAN = 1 then there are 32 channels (+ fission). The latter should only be used when working with heavy excited nuclei.
- PARLEV(24) - IMA : Maximal number of fragments at multifragmentation of nuclei with excitation below 3 MeV/nucleon.
- PARLEV(25) - EPSILO : Inverse level density parameter for internal bulk excitation of fragments in multifragmentation.
- PARLEV(26) - FKACOL : Freeze-out volume for Coulomb interaction of fragments in multifragmentation.
- PARLEV(27) - RNCL : The parameter r_0 , which determines the radius of the excited nucleus by the formula $R = r_0 A^{1/3}$ at the stage of equilibrium de-excitation.
- PARLEV(28) - AMS : Density level parameter of the excited nucleus at evaporation.
- PARLEV(29) - AMFS : Density level parameter of the excited nucleus at fission. Changing this parameter has no effect, as it is overridden internally.
- PARLEV(30) : The border for the Fermi break-up on the number of nucleons: for heavier nuclei the multifragmentation/evaporation/fission model is used.
- PARLEV(31) - ILEVRA : The border for the multifragmentation on the excitation energy per nucleon: for more low excitations only the evaporation/fission model is used. (do not set higher than 2 MeV/nucleon).
- PARLEV(32) - ILEVRA : Products of Fermi break-up are produced only in the ground state (ILEVRA=1), or these products can be formed in an excited state (ILEVRA=2).
- PARLEV(33) - FKAP1 : Fermi break-up: freeze-out volume for translation motion of fragments.
- PARLEV(34) - FKAP2 : Fermi break-up: freeze-out volume for Coulomb interaction of fragments.
- PARLEV(35) -- PARLEV(38) : Reserved.
- PARLEV(39) - SIGION : The parameter for renormalization the total $\sigma_{\text{tot}}(E)$ and inelastic $\sigma_{\text{in}}(E)$ cross sections of nucleus-nucleus interactions in the cascade inside a macroscopic target.
- PARLEV(40) - MICROD : The parameter for renormalization the total $\sigma_{\text{tot}}(E)$ and inelastic $\sigma_{\text{in}}(E)$ cross sections of hadron-nucleus interactions in the cascade inside a macroscopic target.

Table A.6: List of PARLEV parameters with their default values in SHIELD-HIT12A.

PARLEV#	Description	Default	Units	Remarks
1	AGT: PIDABS	0.01	probability	
2	AGT: TLIMIT	25.0	fm / c	¹⁾
3	AGT: MMES	0.2	GeV	¹⁾
4	AGT: MBAR	999.999	GeV	¹⁾
5	AGT: EPS1,EPS2	0.007	GeV	
6	AGT: VPI	0.025	GeV	
7	AGT: C1,C2	0.545	fm	Wood-Saxon param.
8	AGT: D1,D2	0.05	(N/A)	
9	AGT: R0N1,R0N2	1.097	fm	
10	AGT: R0N H(2,1)	2.704	fm	
11	AGT: R0N H(3,1)	2.55	fm	
12	AGT: R0N He(A,2)	2.55	fm	$A \neq 4$
13	AGT: R0N He(4,2)	2.157	fm	
14	AGT: R0N Li(6,3)	3.317	fm	
15	AGT: R0N Li(A,3)	3.11	fm	$A \neq 6$
16	AGT: R0N Be(A,4)	3.25	fm	all Be isotopes
17	AGT: R0N B(10,5)	3.16	fm	
18	AGT: R0N B(A,5)	2.48	fm	$A < 10$
19	AGT: R0N C(A,6)	2.48	fm	$A \leq 10$
20	AGT: R0N N(A,7)	2.48	fm	$A \leq 10$
21	AGT: R0N O(A,8)	2.48	fm	$A \leq 10$
22	PRECO: PRECOM	1.0	(N/A)	
23	DEEX: ICAN	0.0	flag: 0 or 1	
24	DEEX: IMA	3.0	(N/A)	
25	DEEX: EPSIL0	16.0	MeV	
26	DEEX: FKACOL	2.0	(N/A)	
27	DEEX: RNCL	0.0	fm	
28	DEEX: AMS	0.125	MeV ⁻¹	level density
29	DEEX: AMFS	0.125	MeV ⁻¹	level density
30	DEEX: PARLEV(30)	16.0	Mass number	A_{\max} for Fermi density
31	DEEX: PARLEV(31)	0.002	GeV/nucleon	min of U_{mult}/A
32	DEEX: ILEVRA	2.0	flag: 1 or 2	ground or excited
33	DEEX: FKAP1-Coulomb	0.65	(N/A)	
34	DEEX: FKAP2-Volume	18.0	(N/A)	
35	(Reserved)	0.0	-	
36	(Reserved)	0.0	-	
37	(Reserved)	0.0	-	
38	(Reserved)	0.0	-	
39	SIGION	1.00	(N/A)	Renorm $\sigma_{AA}(E)$
40	MICROD	1.00	(N/A)	Renorm $\sigma_{hA}(E)$

¹⁾ For high energies1 fm = 10^{-13} cmc = $3 \cdot 10^{10}$ cm/sec

N/A : Non-applicable, dimensionless

Appendix B

Additional background information

B.1 Optional antiproton annihilation corrections APCORR

When negatively charged antimatter particles traverse matter come to rest they will get captured to an atom replacing an orbital electron and eventually annihilate on the atomic nucleus.

In a compound target the projectile (in SHIELD-HIT12A the possible negative antiparticles are $p\bar{i}^-$, K^- and \bar{p}) can get captured to each of the atoms in the compound. The decision which atom the projectile is captured to is based on the calculation of the capture probability for the individual atoms in the compound target. Currently the default calculation is based on the Fermi-Teller Z -law [?], which states that the capture probability is proportional to the Z -value of the target atom. But this law has shown not to give the correct description of experimental data (see e.g. [?]). Physically more correct theories have been developed, and a combination of the formulas found by Daniel [?] and by Ponomarev [?] and extended by Pawlevicz [?] (called the PPD-law here) has been implemented into SHIELD-HIT12A – for $p\bar{i}^-$, K^- and \bar{p} projectiles. The PPD-law is enabled by setting the flag **APCORR** to 1 in *beam.dat*.

The two calculation routines have been tested against experimental data for the antiproton depth dose curve in water. It was found that the current calculation routine using the Fermi-Teller Z -law agreed well with the experimental data, whereas the new implementation of the PPD-law makes the simulations overestimate the dose in the top of the Bragg peak, but this can be resolved by scaling the antiproton cross sections by a factor of 1.08. This scaling is enabled along with the PPD-law, when setting **APCORR** to 1 in *beam.dat*. [Article in preparation.]

Therefore nearly the same result should be seen when transporting antiprotons setting **APCORR** to either 0 or 1 (this is tested having water as target medium), but a better physically justified theory is enabled by setting **APCORR** to 1. This new implementation is for research and development purposes and are subject to change as more experimental data become available.

B.2 History

This is a short ChangeLog that describes the most noteworthy changes in the A-branch of SHIELD-HIT.

SHIELD-HIT12A

SHIELD-HIT12A is the first version of SHIELD-HIT that was ever released in public.

- All changes from SHIELD-HIT10 were merged into SHIELD-HIT12A; these were mostly minor bugfixes.
- *for022.dat*, *for023.dat* and *pasin.dat* were renamed into *mat.dat*, *beam.dat* and *geo.dat*
- Comments allowed in *geo.dat*.
- *mat.dat* is now in free format, making extension with new cards possible without breaking compatibility. Comments are now allowed, too.
- *beam.dat* is now in free format, making extension with new cards possible without breaking compatibility. Comments are now allowed, too.
- Flat circular and flat square beams are now possible.
- Transportation in any direction, specified with the BEAMDIR card.
- Implemented Gaussian or flat optional beam divergence and beam focus model using the BEAMDIV card.
- Updated inelastic antiproton cross sections. This is still experimental.
- Implementation of a new Vavilov straggling function, which is 5-6 times faster than the old version taken from GEANT3.21. This speeds up SHIELD-HIT12A by 30-40 % for a typical C-12 depth dose calculation.
- New Moliere scattering function.
- Removed code that was taken from GEANT3.21 thereby liberating SHIELD-HIT12A from GPL license and Copyright issues.
- No need for *atab.dat* and *tabnuc.dat* files as they are hardcoded now.
- Bug fixes, most prominently the missing fluence in vacuum issue (ticket #27).
- Script for generating stopping power data files from libdEdx.
- Scoring by zone is now provided by the new scoring system. Zone-scoring was also possible with the old scorer, however, incompatible with parallelization.
- New detector: alanine dosimeter response model is now included following [?].
- Implementation of neutron cross sections (below 14.5 MeV) for Argon. Update of neutron cross sections for Tungsten.
- Update of antiproton total and inelastic inflight annihilation cross section, following Sychev's data book.
- TRACE scorer added for displaying particle trajectories.
- Voxelized structures in the form of CT-cubes can be loaded.

SHIELD-HIT10A

SHIELD-HIT10A was never released as such, but was subject to continuous development. SHIELD-HIT10A started as a fork from the SHIELD-HIT08 code. Changes to SHIELD-HIT10 were still migrated to SHIELD-HIT10A though during the development process.

- This manual was initiated.
- The *shieldhit* executable accepts a directory as argument, and -n option provides an iterator for multiple runs in the same directory, which is useful for parallelization.
- Arbitrary mesh / cylindrical scoring, with several estimators.
- External beam files can be read, i.e. Spread out Bragg-peak data files derived from GSI raster scan files.
- Ripple filters can be defined and loaded.
- RANLUX random number generator replaced by RANSHI.
- Added script *shield_detect2ascii* which can average results from multiple parallelized runs.
- Baraschenkov's cross sections updated for heavy ion - heavy ion reactions.
- Fermi-Breakup: free coulomb energy and free volume parameters changed from 1.0 to 0.65 and 18.0 respectively (see PARLEV33 and 34).

Appendix C

The BDO Fileformat

C.1 Overview

BDO stands for "Binary Data Object", and is a file where there results is stored with plenty of descriptive meta-data. The binary format is organized with a magic filename followed by several *tokens*. The BDO files is ordered as follows:

1. Magic number (ASCII, 8-bytes)
2. Short version number (ASCII, 16-bytes)
3. General header (Binary, tokenized)
4. Geometry data (Binary, tokenized)
5. Estimator data (Binary, tokenized)
6. One or more pages. (Binary, tokenized)

Fileformat details:

- Each .bdo file starts with 8 bytes: a 6 byte ASCII magic number + a 2-byte ASCII string indicating big (**MM**) or little endian (**II**).
- Then follows a 16 byte ASCII string indicating the major and minor version number.
- The rest of the file is described with tokens. The format of a token is described in section C.2.
- A general header follows, containing general information about the simulation.
- A SHIELD-HIT12A simulation may consist of many estimators, however, every file will only contain one single estimator (as one single **Output** section.)
- The geometry section is started with the **SHBDO_GEO_TYPE** tag. The available geometry types are listed in section C.6.
- The estimator section is started with the **SHBDO_EST_FILENAME** tag.
- Every estimator may contain one or more pages. Each new page is strated with the **SHBDO_PAG_TYPE** tag. The available detector types are listed in section C.7.

- The page data is grouped, since every page will lead to tag duplications.
- Other than for the page data, there is no requirement how the tokens are ordered.

C.2 Token Description

The idea of having tokens is to be able to embed data with some minimal description. The actual data is stored in the payload. Tags then describe the data length, the type of the elements (integer, float or similar), and what kind of data is stored (such number of particles simulated, what detector was used, what geometry was used, etc.).

- A token consists of a tag and a payload.
- The tag is always fixed in size.
- The payload size varies, and the size is specified in the tag.

A tag holds,

1. tag-id: an 8-byte integer. These tag IDs are defined in *sh_bdo.h*, as SHBDO_*, see table C.4.
2. payload descriptor : an 8-byte ASCII string, describing the type of *a single element of the payload*. E.g. "<8f_"" for 8-byte floats. The nymphy-type nomenclature¹ is followed here, see table C.5.
3. payload length : a `size_t` type, holding the number of elements (*not bytes*) in the payload.

In summary ("pl" = "payload"):

Token = [tag-id (8-byte int)]	! Part of tag
[pl element type (8-byte char)]	! Part of tag
[pl length (size_t)]	! Part of tag
[pl (variable size)]	! Payload

- The payload has variable length but is always 8-byte padded.
- ASCII strings in the payload will always be null-byte terminated with at least one null byte.
- Integers are always 8-byte integers (long long).
- Floats are always 8-byte floats (double).
- Therefore, in case of integers and floats, the number of elements in a payload is always the `bytlength / 8`.

```
struct SHBDO_TAG {    /* 8 bytes */
    uint64_t tag;
    char pltype[8];
    size_t len;        /* number of elements in payload (NOT number of bytes) */
};
```

¹<https://docs.scipy.org/doc/numpy/reference/arrays.dtypes.html>

C.3 Compressed Format - .bdz

The .bdz format is a .bdo format which is compressed using the zlib² DEFLATE method. These files can readily be unzipped, e.g. using Python:

```
import zlib
with open('foobar.bdz') as f:
    s = f.read()
bdo = zlib.decompress(s)
```

C.4 Tag IDs

This is listed in *sh_bdo.h*.

```
/*
   Hex values are used for better recognition in binary files, should they be inspected by humans.
*/
/* Group 0x0000 - 0x00FF : Miscellaneous info */
SHBDO_SHVERSION = 0x0000, /* [char*] version of SHIELD-HIT12A */
SHBDO_SHBUILDDATE, /* [char*] date of build */
SHBDO_FILEDATE, /* [char*] bdo file creation date, RFC 2822 compliant */
SHBDO_USER, /* [char*] optional login name */
SHBDO_HOST, /* [char*] optional host where this file was created */
SHBDO_FORMAT, /* [int] optional ID describing which flavour the format is, to help .bdo parsers */

/* Group 0xCB00 - 0xCBFF : Beam configuration */
SHBDO_JPART0 = 0xCB00, /* [int] primary particle ID, in SH12A JPART terminology (32768 = INVALID) */
SHBDO_APROO, /* [int] number of nucleons A of the projectile - only written if nucleons > 0 */
SHBDO_ZPROO, /* [int] charge Z of the projectile, may also be negative (32768 = INVALID) */
SHBDO_BEAMX, /* [float] start position of the beam - X coordinate */
SHBDO_BEAMY, /* [float] start position of the beam - Y coordinate */
SHBDO_BEAMZ, /* [float] start position of the beam - Z coordinate */
SHBDO_SIGMAX, /* [float] lateral extension of the beam in X direction */
SHBDO_SIGMAY, /* [float] lateral extension of the beam in Y direction */
SHBDO_TMAXO, /* [float] initial projectile energy (unit depends on projectile type) */
SHBDO_SIGMATO, /* [float] energy spread of the primary particle */
SHBDO_BEAMTHETA, /* [float] polar angle */
SHBDO_BEAMPHI, /* [float] azimuth angle */
SHBDO_BEAMDIVX, /* [float] beam divergence - X coordinate */
SHBDO_BEAMDIVY, /* [float] beam divergence - Y coordinate */
SHBDO_BEAMDIVK, /* [float] beam divergence - focus */
SHBDO_TMAXOMEV, /* [double] initial projectile energy, always in [MeV] */
SHBDO_TMAXOAMU, /* [double] initial projectile energy in [MeV/amu] - only written if mass > 1e-6 u */
SHBDO_TMAXONUC, /* [double] initial projectile energy in [MeV/nuc] - only written if nucleons > 0 */

/* Group 0xCC00 - 0xCCFF : Configuration */
SHBDO_DELE = 0xCC00,
SHBDO_DEMIN,
SHBDO_ITYPST,
SHBDO_ITYPMS,
SHBDO_OLN,
SHBDO_INUCRE,
SHBDO_IEMTRANS,
SHBDO_IEXTSPEC,
SHBDO_INTRFAST,
SHBDO_INTRSLOW,
SHBDO_APZLSCL,
SHBDO_IOFFSET,
SHBDO_IRIFIMC,
SHBDO_IRIFITRANS,
SHBDO_IRIFIZONE,
SHBDO_EXT_NPROJ, /* requested number of projectiles (not the actual number of simulated ones) */
SHBDO_EXT_PTVDOSE,
SHBDO_IXFIRS,

/* # Group 0xCE00 - 0xCEFF : CT specific tags */
```

²<https://www.zlib.net/>

```

SHBDO_CT_ANG = 0xCE00, /* holds two doubles with the couch and gantry angle */
SHBDO_CT_ICNT, /* holds three */
SHBDO_CT_LEN, /* holds three */

/* ----- NEW STUFF HERE ----- */
/* Each estimator is a separate file, so there can only be one single geometry per estimator,
   which simplifies things */

/* Group 0xEE00 - 0xEEFF : Geometry, as in gEOMETRY */
SHBDO_GEO_TYPE = 0xEE00, /* geometry type ID, see SH_SGEO_* in sh_scoredef.h */
SHBDO_GEO_NAME, /* User-given name of this geometry */
SHBDO_GEO_P, /* start values, e.g xmin, ymin, zmin */
SHBDO_GEO_Q, /* stop values, e.g xmax, ymax, zmax */
SHBDO_GEO_N, /* number of bins */
SHBDO_GEO_ROT, /* [future] rotation of geometry */
SHBDO_GEO_VOL, /* volume size in cm3 ... WARNING: may be a list in some future */
SHBDO_GEO_ZONES, /* single GEMCA zone, or list of zones */
SHBDO_GEO_NEQGRID, /* Array of non-equidistant z-grid. Tag only used if set. */
SHBDO_GEO_UNITS, /* [Future]: ASCII string of ;-separated units along each dimension. */
SHBDO_GEO_UNITIDS, /* Unit IDs according to sh_units.h, one unit along each axis. */

/* Group 0xEE00 - 0xEEFF : Estimator, */
SHBDO_EST_FILENAME = 0xEE00, /* number of detectors / pages for this estimator */
SHBDO_EST_COUNT, /* Unique number for this estimator, if several files were saved, starting at 0 */
SHBDO_EST_NPAGES, /* number of detectors / pages for this estimator */
SHBDO_EST_RESCALE_NSTAT, /* estimator "per particle" rescaling, absent or set to 1 if no rescaling */
/* note, that written data will *not* be multiplied with this value, it is up */
/* to the BDO reader to multiply with this, if SHBDO_PAGE_NORMALIZE was set */

/* Detector: deprecated detector stuff */
/* Group 0xDD00 - 0xDD2F : reserved for old detect.f fortran Detector / page specific tags. see sh_detect.h */

/* Page: meta-data */
/* Group 0xDD30 - 0xDDFF : page specific tags. */
SHBDO_PAG_TYPE = 0xDD30, /* detector_type, start of a new page block. */
SHBDO_PAG_COUNT, /* Number of this detector, starting at 0 (total is in SHBDO_EST_NPAGES) */
SHBDO_PAG_NORMALIZE, /* Flags for page->postproc on how to postprocess the data in SHBDO_PAG_DATA

Given:
- the data in page->data as x_j
- for j instances of this simulation (starting at j = 0)
- which was done with I_j number of particles.

The resulting data will be termed X and has the units given by SHBDO_PAG_DATA_UNIT

0: X = x_0 for GEOMAP type scorers
1: X = sum_j x_j COUNT, ...
2: X = (sum_j x_j) / (sum_j I_j) NORMCOUNT, ...
3: X = (sum_j x_j * I_j) / (sum_j I_j) LET, ...
4: X = [x_0, x_1, ... x_j] MCPL
(Defines are in score/sh_scoredef.h)
*/
SHBDO_PAG_RESCALE, /* if set and != 1.0 the data set was multiplied with this factor */
SHBDO_PAG_OFFSET, /* if set and != 0.0 the data set was offset with this value */
SHBDO_PAG_MEDIUM_TRANSP, /* [future] ASCII-string for detector medium set in geo.dat */
SHBDO_PAG_MEDIUM_SCORE, /* [future] ASCII-string for detector medium set in detect.dat scoring */
SHBDO_PAG_UNITIDS, /* Unit IDs according to sh_units.h. Set for detector and the two diff. bins */

/* page data */
SHBDO_PAG_DATA = 0xDDBB, /* data block, identical to SHBDO_DET_DATA. This will terminate a page. */
SHBDO_PAG_DATA_UNIT, /* ASCII string unit, including any differentials, *after* postprocessing */

/* Page differential data */
SHBDO_PAG_DIF_SET = 0xDDDD, /* flags if 1 or 2 differential binning was set. 1 for set, -1 for set as log10. */
SHBDO_PAG_DIF_TYPE, /* array holding 1 or 2 number of bins, for type of differential */
SHBDO_PAG_DIF_START, /* array holding 1 or 2 lower bounds, for 1-D or 2-D respectively */
SHBDO_PAG_DIF_STOP, /* array holding 1 or 2 upper bounds, for 1-D or 2-D respectively */
SHBDO_PAG_DIF_SIZE, /* array holding 1 or 2 number of bins, for 1-D or 2-D respectively */
SHBDO_PAG_DIF_UNITS, /* ASCII string of ;-separated units along each dimension. %s;%s;%s where latter
two %s are the differential units, and the first is the data in
non-differential form. */

/* Settings data attached to page */

```

```

SHBDO_PAG_SETTINGS_NAME = 0xDDE0, /* space delimited list of attached settings names */

/* Filter data attached to page */
SHBDO_PAG_FILTER_NAME = 0xDDF0, /* name of filter containing one or more rules */
SHBDO_PAG_FILTER_NRULES, /* number of filter rules applied */
SHBDO_PAG_FILTER_EMIN, /* lower energy threshold, emin */
SHBDO_PAG_FILTER_EMAX, /* upper energy threshold, emin */

/* Group 0xAA00 - 0xAFFF : Runtime variables */
SHBDO_RT_NSTAT = 0xAA00, /* number of actually simulated particles */
SHBDO_RT_TIME, /* optional total runtime in seconds */
SHBDO_RT_TIMESIM, /* optional simulation time in seconds, excluding initialization.
Future: exclude finalization from this as well. */

/* Group 0xFFCC - 0xFFFF : Diagnostics, may be ignored by readers. */
SHBDO_COMMENT = 0xFFCC, /* 0xFFCC-comment */
SHBDO_DEBUG, /* 0xFFCD-debug */
SHBDO_ERROR /* 0xFFCE-error */

```

C.5 Payload Element Type Description

```

#define SHBDO_PL_TYPE_NONE "V" /* raw data (void) */
#define SHBDO_PL_TYPE_CHAR "S" /* unsigned char */
#define SHBDO_PL_TYPE_LLUIINT "u8" /* long long unsigned int */
#define SHBDO_PL_TYPE_LLSINT "i8" /* long long signed int */
#define SHBDO_PL_TYPE_DOUBLE "f8" /* 8-byte double float */

```

C.6 Geometry and Detector Identifiers

The SHBDO_GEO_TYPE tag has a geometry identifier in its payload. This identifier is an integer number, which is mapped as shown in the table below. This map is listed in *sh_scoredef.h*.

```

/* geometry types */
#define SH_SGEO_NONE 0
#define SH_SGEO_MESH 1 /* cartesian mesh */
#define SH_SGEO_CYL 2 /* cylindrical mesh */
#define SH_SGEO_ZONE 3 /* GEMCA zones */
#define SH_SGEO_PLANE 4 /* single infinite plane */
#define SH_SGEO_VOXEL 5 /* voxelized structure */
#define SH_SGEO_DDDBSPC 6 /* DO NOT USE - DDB or SPC grid (always cylindrical) */
#define SH_SGEO_ALL 7 /* Geometry for scoring everything in the universe */
#define SH_SGEO_SPHSURF 8 /* scoring crossing a sphere surface */
#define SH_SGEO_CYLSURF 9 /* scoring crossing a cylinder surface */

```

C.7 Detector Identifiers

The SHBDO_DET_TYPE tag has a detector identifier in its payload. This identifier is an integer number, which is mapped as shown in the table below. This map is listed in *sh_scoredef.h*.

```

/* detector types */
#define SH_SDET_NONE 0 /* No detector */
#define SH_SDET_ENERGY 1 /* average energy lost in voxel, not E which is Ekin */
#define SH_SDET_FLUENCE 2 /* normal fluence, calculated by Chilton's definition, eqv. to ICRU fluence. */
#define SH_SDET_CROSSFLU 3 /* TODO: planar fluence */
#define SH_SDET_LETFLU 4 /* LET * fluence */

#define SH_SDET_DOSE 5 /* dose in MeV/g */
#define SH_SDET_DLET 6 /* dose-averaged LET, Cortez paper algorithm "C" */
#define SH_SDET_TLET 7 /* track-averaged LET, Cortez paper algorithm "C" */
#define SH_SDET_AVGENERGY 8 /* track-averaged kinetic energy */
#define SH_SDET_AVGBETA 9 /* track-averaged Beta */

#define SH_SDET_SPC 10 /* INVALID - DO NOT USE */

```

```

#define SH_SDET_MATERIAL 11 /* TODO */
#define SH_SDET_DDD 12 /* as DOSE, but with different ID for pymchelper to generate DDD files */
#define SH_SDET_ALANINE 13 /* Alanine scorer, as published by Bassler 2008 */
#define SH_SDET_NORMCOUNT 14 /* simple counter */

#define SH_SDET_PETCOUNT 15 /* TODO: pet-isotope generation */
#define SH_SDET_DLETG 16 /* dose-averaged LET in the "bad" way, Cortez paper algorithm A */
#define SH_SDET_TLETG 17 /* track-averaged LET in the "bad" way, Cortez paper algorithm A */
#define SH_SDET_NZONE 18 /* zone number */
#define SH_SDET_NMEDIUM 19 /* medium number */

#define SH_SDET_RHO 20 /* density of medium */
#define SH_SDET_Q 21 /* Q not averaged, e.g. for differential scorers */
#define SH_SDET_FLUCHAR 22 /* these are redundant, as you can filter as well, kept for compability */
#define SH_SDET_FLUNEUT 23 /* --- " " --- */
#define SH_SDET_1MEVNEQ 24 /* 1 MeV neutron equivalent fluence, used for space/radiation damage assessment */

#define SH_SDET_ANGLE 25 /* angle of particle relative to z axis */
#define SH_SDET_TRACE 26 /* trace scorer */
#define SH_SDET_EKIN 27 /* kinetic energy */
#define SH_SDET_ENUC 28 /* -- " -- per nucleon */
#define SH_SDET_EAMU 29 /* -- " -- per amu */

#define SH_SDET_A 30 /* particle nucleon number, if applicable */
#define SH_SDET_AMASS 31 /* particle mass in terms of MeV/c**2 */
#define SH_SDET_AMU 32 /* particle atomic mass in terms of u */
#define SH_SDET_GEN 33 /* particle generation */
#define SH_SDET_ID 34 /* particle ID, see sh_particle.h */

#define SH_SDET_DEDX 35 /* electronic stopping power - only for differential scorer */
#define SH_SDET_MDEDX 36 /* electronic MASS-stopping power - only for differential scorer */
#define SH_SDET_TL 37 /* track-length, also for differential */
#define SH_SDET_NKERMA 38 /* Neutron Kerma */
#define SH_SDET_DOSEGY 39 /* Just like DOSE, but will output in Gy */

#define SH_SDET_DOSEEQV 40 /* Dose-equivalent, ICRP 103 operational quantity. */
#define SH_SDET_EQVDOSE 41 /* Equivalent dose, ICRP 103 protection quantity. */
#define SH_SDET_USER1 42 /* User-defined detector 1. */
#define SH_SDET_USER2 43 /* User-defined detector 2. */
#define SH_SDET_NEQVDOSE 44 /* Neutron equivalent dose, ICRP 103 protection quantity. */

#define SH_SDET_Z2BETA2 45 /* z**2/beta**2 used if not averaged (e.g. for differential scorers) */
#define SH_SDET_DZ2BETA2 46 /* Dose-averaged z**2/beta**2 */
#define SH_SDET_TZ2BETA2 47 /* Track-averaged z**2/beta**2 */
#define SH_SDET_DQ 48 /* Dose-averaged Q */
#define SH_SDET_TQ 49 /* Track-averaged Q */

#define SH_SDET_Z 50 /* Charge (not averaged, e.g. for differential scorers) */
#define SH_SDET_ZEFF 51 /* Effective-charge (not averaged, e.g. for differential scorers) */
#define SH_SDET_ZEFF2BETA2 52 /* z_eff**2/beta**2 used if not averaged (e.g. for differential scorers) */
#define SH_SDET_DZEFF2BETA2 53 /* Dose-averaged z_eff**2/beta**2 */
#define SH_SDET_TZEFF2BETA2 54 /* Track-averaged z_eff**2/beta**2 */

#define SH_SDET_COUNT 55 /* simple counter, not normalized to per primary particle */
#define SH_SDET_NORMCOUNTPOINT 56 /* score_point counter, normalized per primary particle */
#define SH_SDET_COUNTPOINT 57 /* score_point counter, not normalized per primary particle */
#define SH_SDET_MOCA_YF 58 /* frequency-averaged lineal energy from Moca */
#define SH_SDET_MOCA_YD 59 /* dose-averaged lineal energy from Moca */

#define SH_SDET_QEFF 60 /* Effective Q, Q_eff, aka zeff2beta2 */
#define SH_SDET_DQEFF 61 /* Dose-averaged Q_eff */
#define SH_SDET_TQEFF 62 /* Track-averaged Q_eff */

#define SH_SDET_INVALID 32767

```

C.8 Unit IDs

```

#define SH_SCORING_UNIT_UNKNOWN -2 /* e.g. for user defined scoring */
#define SH_SCORING_UNIT_INVALID -1 /* N/A (not applicable), or undefined */

#define SH_SCORING_UNIT_NONE 0 /* dimensionless */
#define SH_SCORING_UNIT_AU 1 /* arbitrary units */

```



```

#define SH_SCORING_UNIT_PCT      2      /* percent (%) */
#define SH_SCORING_UNIT_PMIL    3      /* promill (‰) */
#define SH_SCORING_UNIT_RELATIVE 4      /* dimensionless relative fraction */

#define SH_SCORING_UNIT_CM      10     /* cm */
#define SH_SCORING_UNIT_CM2     11     /* cm^2 */
#define SH_SCORING_UNIT_CM3     12     /* cm^3 */
#define SH_SCORING_UNIT_PCM     13     /* cm^-1 */
#define SH_SCORING_UNIT_PCM2    14     /* cm^-2 */
#define SH_SCORING_UNIT_PCM3    15     /* cm^-3 */

#define SH_SCORING_UNIT_M       16     /* m */
#define SH_SCORING_UNIT_M2      17     /* m^2 */
#define SH_SCORING_UNIT_M3      18     /* m^3 */
#define SH_SCORING_UNIT_PM      19     /* m^-1 */
#define SH_SCORING_UNIT_PM2     20     /* m^-2 */
#define SH_SCORING_UNIT_PM3     21     /* m^-3 */

#define SH_SCORING_UNIT_GPCM3   22     /* density g / cm^3 */
#define SH_SCORING_UNIT_KGPM3   23     /* density kg / m^3 */

#define SH_SCORING_UNIT_KEVPUM  30     /* keV / um */
#define SH_SCORING_UNIT_MEVPCM  31     /* MeV / cm */
#define SH_SCORING_UNIT_MEVCM2PG 32     /* MeV * cm^2 / g */

#define SH_SCORING_UNIT_MEVPG   40     /* MeV / g */
#define SH_SCORING_UNIT_GY      41     /* Gy (J/kg) */
#define SH_SCORING_UNIT_GYRBE   42     /* Gy * RBE */
#define SH_SCORING_UNIT_GYRE    43     /* Gy * RE (quenching) */
#define SH_SCORING_UNIT_SV      44     /* Sv (J/kg) */
#define SH_SCORING_UNIT_DOSERAD 45     /* Rad ( = 1 cGy) */
#define SH_SCORING_UNIT_DOSEREM 46     /* Rem ( = 1 cSv) */

#define SH_SCORING_UNIT_DEGREES 50     /* degrees */
#define SH_SCORING_UNIT_RADIANS 51     /* radians */
#define SH_SCORING_UNIT_SR      52     /* steradian sr = rad^2*/

#define SH_SCORING_UNIT_COUNT   60     /* number # */

#define SH_SCORING_UNIT_MEV      70     /* MeV */
#define SH_SCORING_UNIT_MEVPMNUC 71     /* MeV / nucleon */
#define SH_SCORING_UNIT_MEVPAMU 72     /* MeV / amu */

#define SH_SCORING_UNIT_NUCN     80     /* number of nucleons */
#define SH_SCORING_UNIT_MEVPC2   81     /* particle mass MeV / c^2 */
#define SH_SCORING_UNIT_U        82     /* particle mass in terms of u */

#define SH_SCORING_UNIT_MATID    90     /* material ID */
#define SH_SCORING_UNIT_NZONE    91     /* zone number */

```

C.9 Fileformat IDs

```

#define SH_FILE_FORMAT_ID_UNDEFINED -1
#define SH_FILE_FORMAT_ID_BIN2010 0      /* Old binary format from 2010, the first version David wrote back then */
#define SH_FILE_FORMAT_ID_BDO2016 1      /* year 2016 .bdo file format, now with proper tags, used from SH 0.6.0 */
#define SH_FILE_FORMAT_ID_BDO2019 2      /* year 2019 .bdo style, introduced June 2019 */
#define SH_FILE_FORMAT_ID_ASCII   3      /* raw text format */
#define SH_FILE_FORMAT_ID_CSV     4      /* comma separated file format */
#define SH_FILE_FORMAT_ID_CSV_TRACE 5     /* special CSV for TRACE which will be saved for every step */

/* Whether the file is to be zipped or not, is handled irrespectly of the internal format notation. */

#define SH_FILE_FORMAT_ID_DEFAULT SH_FILE_FORMAT_ID_BDO2019

```


Index

- 1 MeV equivalent neutron fluence, 59
- alanine
 - detectors, 60
- AMASS, 14
- APCORR, 18
- average beta
 - detectors, 60
- average energy
 - detectors, 59
- beam modulator, 23
- beam.dat, 8, **17**
 - example, 26
 - examples, 26
- BEAMDIR, 18
- BEAMDIV, 18
- BEAMPOS, 19
- BEAMSAD, 19
- BEAMSIGMA, 19
- BMODMC, 19
- BMODTRANS, 19
- bodies, **27**
- convertmc, 5, 9, **65**
- counter
 - detectors, 60
- DELTAΕ, 19
- DEMIN, 19
- detect.dat, 8
 - example, 61
- detectors, 59
 - 1MEVNEQ, 59
 - alanine, 60
 - average beta, 60
 - average energy, 59
 - charged particle fluence, 59
 - counter, 60
 - crossing fluence, 59
 - dose, 59
 - dose-averaged LET, 59
 - energy, 59
 - fluence, 59
 - LET-fluence, 59
 - neutral particle fluence, 59
 - Q, 60
 - track-averaged LET, 59
- differential
 - example, 61
- dose-averaged LET
 - detectors, 59
- EMTRANS, 19
- END
 - mat.dat, 14
- examples
 - beam.dat, 26
 - detect.dat, 61
 - differential, 61
 - geo.dat, 24, 30, 31, 33
 - mat.dat, 15
 - scoring, 61
 - voxscore, 61
- EXTSPEC, 20
- fluence
 - charged particle, 59
 - neutral particle, 59
- for*, 63
- geo.dat, 6, **27**
 - example, 24
 - examples, 30, 31, 33
- HIPROJ, 20
- ICRU
 - mat.dat, 14
- IVALUE, 14
- JPART0, 20
- LET

- dose-averaged, 59
- track-averaged, 59

Linux

- installation, 5

- LOADDEDX, 14

- LOENT, 3

- MAKELN, 20

- mat.dat, 7, **12**

- example, 15

- media, **30**

- MEDIUM, 14

- MSCAT, 20

- NEUTRFAST, 20

- NEUTRLCUT, 20

- NIEL, 59

- NSTAT, 20

- NUCLID, 14

- NUCRE, 21

- output files, 63

- parameter

- load external, 23

- pymchelper, 5

Q

- detectors, 60

- RHO, 14

- RiFi, 4

- ripple filter, 4

- RNDSEED, 21

- scoring, 3

- cylindrical, 53

- examples, 61

- geomap, 51

- voxscore, 58

- zone, 52

- shield_dEdx, **66**

- source

- load external, 21

- STATE, 14

- stopping power

- load external, 12

- stopping power, 66

- STRAGG, 21

- TCUT0, 21

- TLE, 4

- TMAX0, 21

- TRACE, 57

- Track Length Estimation, 4

- track-averaged LET

- detectors, 59

- uncertainties, 60

- USEBMOD, 21, 23

- USECBEAM, 21

- USEPARLEV, 21, 23

- VOX, **28**

- voxel

- materials, 13

- scoring, 58

- voxel body, **28**

- VOXMED, 14

- voxscore

- example, 61

- Windows

- installation, 5

- zones, **29**

- geomap, 51

- scoring, 52