CALM - user manual

Filip Skóra, Bartosz Woźnica

February 1, 2024

Contents

1 Introduction

This instruction will provide the basic information about how to use "ConservAtion Laws Model", called CALM, as a non programmer user. For more specific information please take a look to the "CALM - documentation" file, where you can find how to modify CALM source code to make simulation more suitable for your experiment. This user guide was written by Filip Skóra as an attachment for his engineering thesis conducted by Dr Engineer Małgorzata Janik. The user guide was subsequently updated by Bartosz Woźnica for CALM version 1.2 during the writing of his engineering thesis.

CALM was developed by Anna Zaborowska, Małgorzata Janik, Filip Skóra, Daniel Rodak and Bartosz Woźnica based on the TERMINATOR [1] event generator.

2 Installation

The installation will be carried out on Linux system.

2.1 Requirements

- CALM source archive – `https://github.com/wozniczu/CALM.git`
- Cmake – `https://cmake.org`
- C++ compiler (for example gcc)
- CERN Root Framework – `https://root.cern/install/`

Cmake, compiler and Root have to be installed on the machine and CALM source archive extracted to the installation directory.

2.2 CALM archive

The archive contains:

- Source code of CALM – "include" and "src" directories
- Particle database that are used by CALM in its runtime – "Shared/particles.data"
- Configuration files – event.ini, config.ini
- Calibration files – "distributions" directory
- Script that helps to launch CALM – "run_CALM.sh"

Details and utility of configuration files and script will be described later inside this manual.

2.3 Compilation

Type inside installation directory:

```
mkdir build
cd build
cmake ..
make
```

2.4 How to launch CALM

To launch CALM with default configuration just run "calm" file inside build directory. Alternatively, run "run_CALM.sh" to generate multiple events simultaneously (runs the CALM program on multiple cores).

3 CALM - working scheme

1. INITIALIZATION: Reading configuration from .ini files
2. Creating event
3. Generating multiplicity for event
4. Generating types for particles; event must obey conservation laws.
5. Generating direction of movement for each particle
6. Generating total energy of collision
7. Dividing total energy between all particles
8. Generating event; filtering the most unlikely events
9. Saving event to the file
10. POST ACTION: After doing steps 2-9 10000 times, CALM closes the file.

4 Configuration files

Configuration files gives access to some elements inside CALM.

4.1 events.ini

This file provides data which are used before the simulation starts.

Parameters:

- NumberOfEvents – number of events to generate
- EventFileType – event output file format (available: root, root&text or text)
- MultiplicityDistribution – distribution of primordial particles multiplicity (available: Poisson)
- Randomize – start each event with a new random seed taken from current time (1) or do a constant seed (0)
- ShareDir – directory with SHARE input files
- MacroDir – directory with ROOT macro files *.C
- EventDir – directory to write the events
- LogFile – log file
- MultiplicityMin – multiplicity range, minimum
- MultiplicityMax – multiplicity range, maximum
- GenbodEnergy – event energy
- EventType – what is to be generated (section 5: CALM options)
- NumberOfJets – number of jets generated

4.2 config.ini

This file provides data which CALM uses to simulate elements of each event.

Parameters:

- Nmean – average charged particle multiplicity per rapidity unit from 900 GeV
- RapidityInterval – rapidity interval
- XYZ – standard deviation of three dimensional gaussian distribution
- divideEn – coefficients for division of energy
- ${particleType}MultDistr – custom distribution function for ${particleType}
- ${particleType}_xMin – custom distribution function minimum range for ${particleType}
- ${particleType}_xMax – custom distribution function maximum range for ${particleType}
- singleEnergyDistr – custom distribution function for singleEnergy
- singleEnergyDistr_xMin – custom distribution function minimum range for singleEnergy
- singleEnergyDistr_xMax – custom distribution function maximum range for singleEnergy
- ${particleType}MultDistrPath – path to file containing custom distribution function for ${particleType}
- singleEnergyDistrPath – path to file containing custom distribution function for singleEnergy
- customMult – determines if custom distribution for particles will be used
- importMethod – determines if custom distributions will be imported from file

*${particleType} – pions, kaons, nucleons or lambdas

5 CALM options

CALM has six options of simulation. We can distinguish them into two types:

1. GENBOD – program uses CERN Root library Genbod which is used to make similar simulations
2. REGGAE [2] – REscatterig-after-Genbod GenerAtor of Events, more effective type of simulation, easier but much faster

Each of them has three types of influence on outgoing particles:

- Global – simple proton-proton collision (no influence)
- MinijetsGlobal – split particles into two jets and boosts them in opposite directions
- MinijetsLocal – same as above but here CALM will check conservation laws for each jet separately

6 The script

Script "run_CALM.sh" helps to launch CALM faster and more efficiently. It modifies events.ini file with values passed as a arguments and launches CALM in 4 background processes, which means that simulation will be running in 4 parallel threads. Events from each thread will be saved into separate directory inside "EventDir" directory passed in events.ini file.
Script takes two arguments:

1. $EventType – value from following: 0, 1, 2 ,3, 4, 5
2. $NumberOfEvents – each thread simulates that number of events divided by 4

For example:

```
./run_CALM.sh 3 40000
```

runs 4 CALMs each generating 10000 events for GlobalReggae option.

Tip: You can modify NumberOfCores variable, on the beggining of the script, to define how many background simulations will be run.

References

[1] Mikołaj Chojnacki, Adam Kisiel, Wojciech Florkowski, and Wojciech Broniowski, *Thermal heavy ion generator v.2*, Comput.Phys.Commun. 183 (2012) 746-773.

[2] Michal Meres, Ivan Melo, Boris Tomasik, Vladimir Balek, and Vladimir Cerny, *Generating heavy particles with energy and momentum conservation*, Comput. Phys. Commun. **182** (2011), 2561–2566.