The MARS Code System User's Guide
Version 15(2010)

Nikolai V. Mokhov and Catherine C. James
Fermi National Accelerator Laboratory
P.O. Box 500, Batavia, Illinois 60510
September 14, 2011

This is a draft manual for the MARS15 code released in Summer 2011. Section 4 on the input files is up-to-date. All other sections still need further update. This manual, along with recent principal papers listed below, describe new and upgraded features of the MARS15 code, and all can be downloaded from the MARS web site http://www-ap.fnal.gov/MARS/

- N. V. Mokhov, "Recent Mars15 developments: nuclide inventory, DPA and gas production", Fermilab-Conf-10/518-APC (2010) [1].
- P. Aarnio, "Decay and transmutation of nuclides", CMS-NOTE-1998/086, CERN (1998) [2].
- A. Isotalo, "Modifications to DeTra", Technical Report September 18, 2008 [3].
- N. V. Mokhov, I. L. Rakhno and S. I. Striganov, "Simulation and verification of DPA in materials", in *Applications of High Intensity Proton Accelerators, World Scientific Proc. pp. 128-131*, 2010 [4].
- I. L. Rakhno, "Modeling heavy ion ionization energy loss at low and intermediate energies", Fermilab-FN-0835-APC, 2009 [5].
- N. V. Mokhov, S.I. Striganov, "MARS15 Overview", Fermilab-Conf-07/008-AD (2007) [6].
- N. V. Mokhov, K. K. Gudima, S. G. Mashnik et al, "Physics Models in the MARS15 Code for Accelerator and Space Applications", Fermilab-Conf-04/269-AD (2004) [7].
- S. G. Mashnik, K. K. Gudima, A. J. Sierk, M. I. Baznat, N. V. Mokhov, "CEM03.01 User Manual", LANL LA-UR-05-7321 (2005) [8], http://www-rsicc.ornl.gov/codes/psr/psr5/psr-532.html
- N. V. Mokhov, E. I. Rakhno, I. L. Rakhno, "Residual Activation of Thin Accelerator Components", Fermilab-FN-0788-AD (2006) [9].
- I. L. Rakhno, N. V. Mokhov, S. I. Striganov, "Modeling Heavy Ion Ionization Loss in the MARS15 Code", Fermilab-Conf-05/019-AD (2005) [10].
- N. V. Mokhov, K. K. Gudima, C. C. James et al, "Recent Enhancements to the MARS15 Code", Fermilab-Conf-04/053 (2004) [11].

- N. V. Mokhov, K. K. Gudima, S. G. Mashnik et al, “Towards a Heavy-Ion Transport Capability in the MARS15 Code”, Fermilab-Conf-04/052 (2004) [12].
- M. A. Kostin and N. V. Mokhov, “Parallelizing the MARS15 Code with MPI for Shielding Applications”, Fermilab-Conf-04/054 (2004) [13].
- M. A. Kostin, O. E. Krivosheev, N. V. Mokhov, I. S. Tropin, “An Improved MAD-MARS Beam Line Builder: Users Guide”, Fermilab-FN-738-rev (2004) [14].
- S. I. Striganov, “On the theory and simulation of multiple Coulomb scattering of heavy charged particles”, Fermilab-Conf-04/056 (2004) [15].

The main contributors to this version of the code are N.V. Mokhov, P. Aarnio, Y.I. Eidelman, K.K. Gudima, C.C. James, M.A. Kostin, O.E. Krivosheev, S.G. Mashnik, M.E. Monville, I.L. Rakhno, A.J. Sierk, and S.I. Striganov.

Abstract

This paper is a user's guide to the current version of the MARS15 Monte Carlo code. MARS allows inclusive and, in many cases, exclusive simulations of three-dimensional hadronic and electromagnetic cascades and modeling of heavy ion, muon and low energy neutron-photon transport in accelerator, detector, spacecraft and shielding components. Particles can have energies from a fraction of an electronvolt up to about 100 TeV. The code has undergone substantial improvements since the last documented version MARS13(95) and since previous releases of MARS14; all these as well as other specific features of the MARS code system are explained in detail. Descriptions of general input and output with commentary and recommendations are given. Examples are given for running the program with distributed sources, complex compounds, arbitrary geometries, and magnetic fields. A built-in graphical-user interface allows the visualization and detailed analysis of the geometry description. Enhanced options are described for the MARS code being coupled with: the DPMJET hadron event generator, the MCNP code for low-energy neutron transport, the ANSYS system for thermal and stress analysis, physics analysis and graphics packages, the MAD accelerator lattice description via a universal MAD-MARS beam line builder, and the STRUCT program for tracking particles in accelerator lattices with beam loss recording. Use of the code in a multistage mode, coupled with event generators (DPMJET), with the STRUCT program for tracking particles in accelerator lattices with beam loss recording, and with physics analysis and graphics packages is demonstrated with typical input and output examples.

Contents

1 Introduction **7**

2 Physics Models **9**

2.1 Basic Monte Carlo Method . . . 9

2.2 Nuclear Cross Sections . . . 11

2.2.1 Hadron-nucleon cross sections. . . . 11

2.2.2 *Add on heavy-ion cross sections here!!!* . . . 11

2.2.3 Hadron-nucleus cross sections. . . . 11

2.2.4 Photon-nucleus cross sections. . . . 11

2.3 Hadron Production . . . 12

2.3.1 Cascade-exciton model code . . . 12

2.3.2 Inclusive hadron production from 5 GeV to 100 TeV. . . . 12

2.3.3 Deuteron-nucleus collisions. . . . 13

2.3.4 Quark-Gluon String Model code LAQGSM2010. . . . 14

2.3.5 Exclusive hadron production from 5 GeV to 100 TeV. . . . 14

2.4 Elastic Scattering . . . 15

2.5 Muon Production . . . 15

2.6 Electromagnetic Interactions of Heavy Particles . . . 16

2.6.1 Electromagnetic interactions of muons, charged hadrons and heavy ions . . . 16

2.6.2 Multiple Coulomb scattering . . . 16

2.7 Electromagnetic Showers . . . 16

2.8 Synchrotron Radiation *(text to be added)* . . . 16

2.9 Stopped Hadrons and Muons . . . 16

2.9.1 Pions. . . . 17

2.9.2 Muons. . . . 17

2.9.3 Antiprotons. . . . 18

2.10 Neutrino Interactions . . . 18

2.11 Low Energy Neutrons . . . 18

3 Using the MARS Package **20**

3.1 Describing a Geometry by Zones . . . 20

3.1.1 Geometric Zone Volumes . . . 22

3.1.2 Geometries from other Codes . . . 22

3.2 Particles used in MARS . . . 22

3.3 Tracking . . . 23

3.4 Materials . . . 25

3.5 Tabulation & Results . . . 29

3.6 Histogram Options 29
3.7 Visualization *needs further update* 29
3.8 Statistics Guidelines *needs further update* 29
3.9 Variance Reduction and Biasing *needs further update* 29
3.10 Interfaces *needs further update* 30
3.11 Getting Started 30

4 The `MARS.INP` and `GEOM.INP` Input Files **34**
4.1 Structure of the `MARS.INP` Input Deck 35
4.2 List of Data Cards in `MARS.INP` 36
4.2.1 `IND` Switches for Default/non-Default Options 36
4.2.2 Run Control – `CTRL, RZVL, NEVT, SEED, SMIN, VARS, UCTR` 38
4.2.3 Beam & Energy Thresholds – `IPIB, BEAM, INIT, ENRG` 42
4.2.4 Materials and Material-Dependent Settings – `NMAT, MATR, MTDN, MTCH,MTNE, MTGA, MTEL, MTSM, M`
4.2.5 Standard Geometry Zone Definition – `ZMIN, NLNG, ZSEC, NLTR,RSEC, NAZM, AZIM, IRFL` 49
4.2.6 Importance Sampling – `IMPT, IMPZ, IMPR` 51
4.2.7 Histograms and Tabulation – `NOBL, RZOB, NSUR, RZTS, TOFF, NHBK,HBKE, NCLD, IMNC, NDET,`
4.2.8 Physics Control – `ALMX, BIAS, ICEM, MCSD, MUON, NUFR, PHOT, RZMN` 55
4.2.9 Termination – `STOP` 59
4.3 Examples Using `MARS.INP` Parameters 59
4.3.1 Z-Sandwich Geometry 59
4.3.2 R-Sandwich Geometry 61
4.3.3 Thin Window 62
4.4 Extended Geometry `GEOM.INP` Input File 64
4.4.1 Example of Extended Geometry 69

5 User Subroutines **72**
5.1 Main Program MARSMAIN.F 73
5.2 Subroutine MARS1510 73
5.2.1 Stand-alone event generator 74
5.3 Subroutine MIXTUR - Custom Materials 74
5.4 Subroutine BEG1 - Primary Beam Description 75
5.5 Subroutines REG1 - non-Standard Zone Description 76
5.6 Subroutine VFAN - Specifying Zone Volumes 83
5.7 Subroutine REG3 - Standard Zone Re-numbering 85
5.8 Subroutines FIELD & SUFI - Description of Magnetic & Electric Fields 85
5.9 Subroutine RFCAVT - Description of Accelerating Components 86
5.10 Fictitious Scattering (ALIGN, SAGIT) 86
5.11 Edge Scattering (EDGEUS) 87

5.12 Subroutine LEAK - Creating custom source terms . . . 88
5.13 Subroutines MHSETU, MFILL, HISTODB - User Histograms . . . 89
5.14 Subroutine WRTSUR - Custom Output . . . 91
5.15 Subroutine TAGPR - Particle Origin Tagging . . . 91
5.16 Subroutine TAGGING - Energy Deposition Tagging . . . 93
5.17 Subroutine KILLPTCL - Selective Particle Destruction . . . 93
5.18 Block Data BLPROCESS - Biasing Control . . . 93
5.19 Block Data HISTDUMP - Histogram Bin and DUMP Region Specification . . . 94
5.20 MAD-MARS Interface User Routines . . . 94
5.20.1 Subroutine MARS2BML . . . 94
5.20.2 Subroutine BML2MARS . . . 96
5.20.3 Subroutine BLINIT . . . 97
5.20.4 Subroutine BLGEOINIT . . . 99
5.20.5 Subroutine TUNNELGEO . . . 101
5.21 User Subroutine Examples . . . 101
5.21.1 Simple Model of Beam on a Target . . . 101
5.21.2 Beam Dump . . . 102

6 MCNP Mode 104
6.1 Setting up MARS.INP for use with MCNP . . . 104
6.1.1 Mode 1 – MCNP with Material description only . . . 105
6.1.2 Mode 2 – MCNP with both Material and Geometry description . . . 106
6.2 MCNP Installation . . . 109
6.2.1 Building and running MARS-MCNP . . . 110

7 FLUKA Geometry Mode 111

8 LAQGSM and DPMJET Modes 112
8.1 LAQGSM . . . 112
8.2 DPMJET . . . 112
8.3 Event Generator Output . . . 112

9 MAD-MARS Beam Line Builder 113

10 Histogram Input and Output Files 114
10.1 Built-in Volume Histograms . . . 114
10.2 Built-in Surface Histograms . . . 115
10.3 Other Built-in Histograms . . . 116
10.4 XYZ-Boxes or Arbitrary Mesh Tallies . . . 116
10.5 User-Defined Histograms and Ntuples . . . 120

11 Output of the Simulation **121**
11.1 Data Table Output Files . . . 121
11.1.1 The VOLMC.NON File . . . 122
11.1.2 The MARS.OUT File . . . 122
11.1.3 The MTUPLE File . . . 132
11.1.4 The MTUPLE.NON File . . . 132
11.1.5 The MTUPLE.EXG File . . . 133
11.1.6 The MTUPLE.MCNP File . . . 133
11.2 Other Data Output Files . . . 133
11.2.1 The TRACK.PLOT file . . . 133
11.2.2 The VERTEX.PLOT file . . . 133

12 Graphical-User Interface **134**
12.1 Main Features . . . 134
12.2 New Advanced Features . . . 135
12.3 Installation and usage . . . 136
12.4 Interface features . . . 137

13 Additional Code Topics **148**
13.1 Rules-Of-Thumb . . . 148
13.2 Biasing and Other Control of Physics Processes . . . 148
13.3 Supplementary Routines . . . 149
13.4 Multiprocessing . . . 149
13.5 External Packages Required . . . 149
13.6 Interfaces to Other Programs . . . 149
13.6.1 STRUCT . . . 149
13.6.2 ANSYS . . . 149

14 Additional Information **150**
14.1 MARS Web Page and World-Wide Support . . . 150
14.2 Benchmarking of the Simulation . . . 150
14.3 Future Developments . . . 150

15 Acknowledgements **151**

List of Tables

1 Neutron Energy Groups . . . 19
2 Particle Types . . . 23
3 Built-in Elements . . . 26
4 Built-in Compounds . . . 26
5 Source code organization . . . 33
6 IND(15) . . . 39
7 Extended Geometry shape types and parameters . . . 65
8 List of User Subroutines . . . 73
10 Histogram ID numbering . . . 114
11 Surfaces . . . 116
12 Histogram types in XYZ-histograming. DLT is currently not activated. . . . 117
13 Default GUI rotation settings . . . 136

List of Figures

1 EM shower diagrams, exclusive vs. inclusive-weighted . . . 10
2 MARS inclusive hA vertex diagram . . . 10
3 MARS pion cross sections . . . 12
4 MARS neutron cross sections . . . 12
5 CEM cross sections . . . 13
6 CEM cross sections . . . 13
7 Model-Data comparisons . . . 14
8 Model-Data comparisons . . . 14
9 Model-Data comparisons . . . 15
10 pi-K yield from Deuteron-nucleus collision . . . 15
11 Muon capture . . . 17
12 Absorbed Dose . . . 17
13 Sketch of boundary localization . . . 24
14 z-sandwich example, plan view . . . 60
15 z-sandwich example, flux map . . . 60
16 r-sandwich example, plan view . . . 62
17 r-sandwich example, cross-section view . . . 62
18 r-sandwich example, with flux map . . . 63
19 Thin Window example, plan view . . . 64
20 Thin Window example, cross-section view . . . 64
21 `GEOM.INP` shape over-lapping . . . 68
22 `GEOM.INP` example . . . 71
23 Sketch of a simple beam dump . . . 77
24 Diagram of FLUKA geometry as interpreted by MARS . . . 111

1 Introduction

The MARS code system is a set of Monte Carlo programs which simulate the passage of particles through matter. All manner of particle interactions for hadrons, leptons, photons and heavy ions are present. Interaction and production cross-section modeling has been developed to match data, wherever such data is available, over a wide range of energies. Three-dimensional hadronic and electromagnetic cascades are modeled, as is the transport of particles, including heavy ions, through matter. The simulated particles are tracked through a user described geometry. The users geometry model can be very simple to very complex, incorporating compound materials of any shape and arbitrary magnetic and electric fields. MARS tabulates the particles which pass through various portions of the geometry, and produces results on particle fluxes, spectra, energy deposition, material activation, DPA, and many other quantities. The results are presented in tables, histograms, and other specialized formats. MARS allows the user extensive control over the simulated physics processes, over the tabulation criteria, and over runtime optimizations, via a variety of options and switches implemented from an input file and via customizable user subroutines.

The basic model for the original MARS program introduced in 1974 [16, 17], came from Feynman's ideas concerning an inclusive approach to multiparticle reactions [18] and from the use of certain weighting techniques. At each interaction vertex, a particle cascade tree is constructed using only a fixed number of representative particles (the precise number and type depending on the specifics of the interaction), and each particle carries a statistical weight which is equal, in the simplest case, to the partial mean multiplicity of the particular event. Energy and momentum are conserved *on the average* over a number of collisions. A similar scheme is the basis of the program CASIM introduced in 1975 [19, 20]. Other codes have also adopted this approach: the EGS simulation of high energy electromagnetic showers [21], and hadronic showers simulated by the FLUKA program [22].

The details of this inclusive scheme are described in several references [23, 24, 25, 26]. The practical reasons for using this approach are:

- the CPU time per incident particle grows logarithmically with incident energy, compared to the linear increase in CPU time when performing a full exclusive simulation. This allows for easier simulation of multi-TeV particle cascades;
- in many applications one considers effects due to the simultaneous interactions of a huge number of particles, so to describe the cascades it is sufficient to obtain the first moment of the distribution function using the inclusive cross-sections, in the same manner as with Boltzman's equation;
- experimental data on inclusive interaction spectra are more readily available than for exclusive ones;
- the use of statistical adjustments to the assigned weights allows for the enhanced production of selected particles within a phase-space region of interest, especially for rarely produced particles.

The main disadvantage of this approach is the impossibility of directly studying a single interaction, or examining fluctuations from cascade to cascade, or of studying production correlations. Such studies require the simulation of the full, exclusive, interaction. So, MARS offers the alternative of using exclusive, also called analogous, simulations of hadronic and electromagnetic cascades, but using these alternatives will result in greatly increased CPU-time per generated event.

The MARS code has been developed over the past 3 decades, and is now widely used in the high-energy physics, accelerator, and radiation shielding communities. The code is under continuous development and benchmarked against new data as it becomes available, to maintain the simulation's reliability and predictive power. The mathematical foundation and the physical model of the MARS system, and benchmarks from numerous past applications, are described in many references [24, 26, 27, 28, 29]. Besides the original code

version [17], the milestones in code development are: MARS3 [30], MARS4 [31], MARS6 [32], MARS8 [33], MARS9 [34], MARS10 [35, 36], MARS12 [29, 37], MARS93 [38], the hydrodynamical MARS/MESA/SPHINX package [39] and a version for parallel processing, MARS12 [40].

The purpose of this document is to be a user's guide for the current MARS, Version 15(2010). Portions of the previous manual are carried over, with updated sections describing the main improvements and options. Section 2 gives an overview of the inclusive method and the physics models implemented in MARS. The main updates from the past few years in the physics models are also described in several references [41, 42, 43, 44, 45, 46, 47]. Section 3 outlines the basics of how a geometry is described, along with an overview of how MARS tracks particles, and then tabulates and presents the simulation results.

Further details on using the program are contained in subsequent Sections. Section 4 describes the syntax of the MARS input files `MARS.INP` and `GEOM.INP`, and how these are used to control physics processes and describe simpler geometries. The User Subroutines, utilized to describe more complex geometries and create custom output, are covered in Section 5. Section 12 explains how to use the Graphical User Interface, a visual tool to examine a geometry and to display results. Section 10 describes the input and output histogram files. Section 11 describes the output files produced by MARS, with details of the tables and values in the default output files, and the contents of various special output files. Other Sections describe special running modes.

Recent updates include extended histogram capabilities, more built-in materials, corrected handling of longer-lived hadrons, and improved computational performance. In addition to direct energy deposition calculations, a new set of fluence-to-dose conversion factors for all particles, including neutrinos, has been built into the code. MARS uses portions of the MCNP4C code for default neutron transport below 14 MeV, and has the ability to fully utilize the MCNP code where more detailed simulation of low-energy neutrons is needed. The MARS system also includes links to the ANSYS code for thermal and stress analyses and to the STRUCT code [48] for multi-turn particle tracking in large synchrotrons and collider rings.

The code is available for the Unix and Linux operating systems, and is distributed by the developers from Fermilab. Section 3.11 gives instructions on obtaining the code, and basics on building an executable. For recent publications on the current MARS15 version see the official MARS Web site at http://www-ap.fnal.gov/MARS/.

2 Physics Models

MARS simulates a variety of processes, taking into account all interactions of hadrons, leptons, photons and heavy ions during their passage through matter. The modeled energy range is large: up to 100 TeV for all particles, and down to 100 keV for muons, charged hadrons and heavy ions, down to 1 keV for electrons and photons, and to 0.00215 eV for neutrons. All processes introduced in older versions of the code and described in prior versions of this manual are, for the most part, still present. Processes updated for the current version include: a new nuclear cross section library, a model for soft pion production, a cascade-exciton model, a quark-gluon string model, a dual parton model, deuteron-nucleus and neutrino-nucleus interaction models, a detailed description of negative hadron and muon absorption, and a unified treatment of muon, charged hadron and heavy ion electromagnetic interactions with matter.

This Section summarizes the simulated physics interactions, and gives brief descriptions of the Monte Carlo techniques and the modeling algorithms used to implement them. References to sources and to more detailed descriptions are given for the interested reader. All algorithms are benchmarked against available experimental data, and some of these comparisons are shown here.

Physics interactions are classified into two types - discrete and continuous. A discrete interaction would be when a particle collides with a nucleus - a variety of things can happen at this particular vertex, as given by cross-section probabilities. A continuous interaction would be Coulomb scattering - the particle experiences the effect everywhere along it's path through a material. In general, MARS first examines the likelihood of a discrete interaction occurring, and then applies continuous interactions to the list of surviving or newly produced particles.

The methods for modeling most continuous interactions, like Coulomb scattering or a magnetic field, are fairly standard, and their discussion below is brief. Of more interest to users of a simulation program like MARS is how the discrete interactions are handled, and how well the simulation compares to data. The basic *inclusive* modeling method used for most discrete interactions is briefly discussed here. Sub-sections fill in the details - sources of algorithms and applicable energy ranges for specific interaction types.

The *inclusive* approach to discrete interactions, also called *biased* interaction method, is the MARS default. *Exclusive* production models, also called *analogous* or *unbiased*, are also available in MARS. In certain situations, MARS applies an exclusive approach by default, as that is the only means to accurately reproduce data; such cases are noted where applicable in the physics model descriptions. But in most cases, the use of exclusive interactions in MARS must be explicitly activated by the user. For example, Section 5.2.1 describes how to use the DPMJET generator in a stand-alone mode. The user can also set parameters to generate exclusive EM interactions, but continue using the default inclusive method for hadronic interactions, or visa-versa; these parameters are controlled by the iput cards `BIAS`, `ICEM` and `PHOT` and discussed in Section 4.

The list of particle types employed by MARS is given in Table 2, located in Section 3.2.

2.1 Basic Monte Carlo Method

As stated in the Introduction, Feynman's inclusive approach [18] is used for multiparticle reactions. At an interaction vertex, a particle cascade tree is constructed using a fixed number of representative particles, and each particle carries a statistical weight which is equal, in the simplest case, to the partial mean multiplicity for the particular interaction. Energy and momentum are conserved *on the average* over a number of collisions, but not precisely conserved at any single vertex. An example of this simplest case is shown in Figure 1 for an electromagnetic shower, showing a diagram of a full shower, the exclusive approach, compared to the inclusively modeled version. While the exclusive picture looks more satisfying as an event,

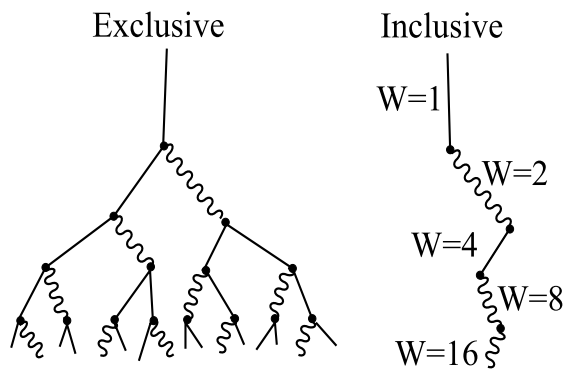


Figure 1: The left diagram is a full EM shower, each particle with a weight =1. The right diagram is the inclusive equivalent, with a single representative particle exiting each vertex, with the given weight.

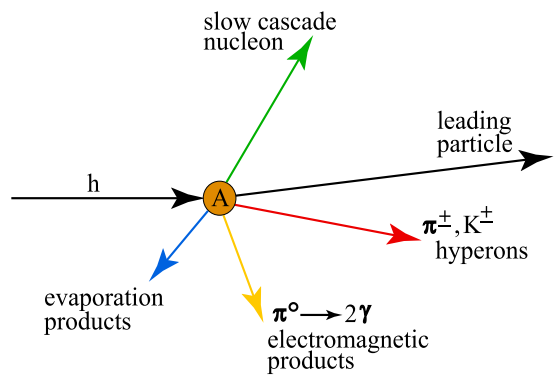


Figure 2: An hA vertex as modeled by the MARS - 5 particles exit the vertex, representing the labeled particle classes.

the inclusive approach has real advantages for a simulation - there are fewer particles to handle. Over many events, the results from the inclusive approach for the bulk quantities of interest, such as particle flux, energy deposition, etc., match data quite well.

A hadronic shower can be represented the same way in the inclusive approach, with a single particle, usually a pion, occasionally something else, exiting the vertex with an appropriate weight. But over time, simulation software developers found that more accurate results were obtained when a certain number of particle classes were included on every vertex. For MARS, a hadron-nucleus vertex is modeled by the inclusive method as shown in Figure 2. Five particle classes are always represented: the leading particle, shower hadrons, electromagnetic products, evaporation products from the nucleus, and the nucleon. The precise particle properties for each of these classes - particle types, weights, X_F, p_t, etc. - are given by a variety of cross-section and production models discussed in sub-sections below. Which models are used depends upon the incoming particle type and it's energy. The interaction type, and the circumstances for use, are given for each model.

There are a couple of important statistical techniques when using the inclusive approach, having to do with the weights applied to representative particles. The statistical accuracy of the results of the simulation - for example the error on the particle flux through a certain portion of the geometry - depends upon the weights assigned to all the simulated particles which were tracked through that portion of the geometry. For very large and very small weights, the divergence of these errors becomes a problem. The solution within MARS is to employ a *weight window*: particles with weights within the window are left as is, but particles with weights above and below the window have special handling applied.

For particles with a weight above the weight window value, *surface splitting* is applied. At an arbitrary surface some distance from the actual interaction vertex, a generated particle with weight W is split into n particles each with a new weight $W' = \frac{1}{n}W$. For example, an initial π with $W = 12$ would be split into four π's each with $W' = 3$. The splitting increases the number of tracked particles and therefore increases the simulation cpu-time, but the results are statistically more stable. For specialized studies, the user can control the location of splitting surfaces and the particle energy at which they are applied - see Section 4.2.6.

Surface splitting is sometimes applied even for particles within the weight window, in order to accumulate statistics in specific regions of interest. This is done, for example, in simulations of thick shielding.

For rarely produced particles, those with a small weight below the weight window, the method of *Russian*

Roulette is applied. When such a particle is generated, it is simply dropped most of the time, but once in a while it is kept, and it's weight is adjusted upward to compensate. For example, a p impacts a target, and the cross-section probability for producing a $\overline{p}$ is, say, $\sim 10^{-10}$. Instead of keeping a particle with such a small weight, it is simply dropped 99% of the time. In 1% of the events which produce it, however, the $\overline{p}$ is kept, and given a larger weight to make up for all the events in which it was generated, but dropped. This method gives more statistically stable contributions to the simulation results from rarely produced particle types. For cases where the user is interested in interactions which involve rarely produced particles, their generation and interactions can be forced - see Section 4.2.6.

2.2 Nuclear Cross Sections

There are several models used for these discrete interactions, depending on the type of incoming particle, and the energy range of the incoming particle.

2.2.1 Hadron-nucleon cross sections.

Updates now cover elastic and inelastic σ_{hN} interactions for hadrons with kinetic energies from 1 MeV to 100 TeV. Total cross sections, σ_{tot}, for p, n, π^+ and π^-,with energies from 1 MeV to 10 GeV, are calculated by the Cascade-Exciton Model (CEM) [49] algorithm [50] used by the code package CEM95 [51]. This package has recently been updated. Cross-sections for K^+, K^- and $\overline{p}$ are derived from PDG data compilations [52], and PDG parameterizations are used for all particles with energies between 10 GeV and 100 TeV. Elastic cross sections, σ_{el}, for p, n, π^+ and π^- from 10 MeV to 10 GeV, are calculated from CEM algoritms, while elastic cross sections for K^+, K^- and $\overline{p}$ are interpolated from data by PDG [52]. Parameterizations from PDG are used for elastic cross-sections for all these particles with energies between 10 and 200 GeV. For all particle types with energies between 200 GeV and 100 TeV, the optical theorem with 'universal slope' [53] is applied. As a sample comparison between data and MARS, Figure 3 shows comparisons for σ_{tot} and σ_{el} for $\pi^- - p$ collisions.

2.2.2 *Add on heavy-ion cross sections here!!!*

2.2.3 Hadron-nucleus cross sections.

Total, inelastic and elastic σ_{hA} cross sections from 1 MeV to 5 GeV are described using new compilations and improved interpolation algorithms [54, 55]. At higher energies (5 GeV<E<100 TeV), σ_{tot}, σ_{in}, σ_{prod} and σ_{el} are calculated in the framework of the Glauber multiple scattering theory with the above σ_{hN} as an input. The nucleon density distribution in nuclei is represented as the symmetrized Fermi function with the parameters of [56] for medium and heavy nuclei ($Z > 10$) and the ones of [57] for $Z < 10$. An example is shown in Fig. 4 for neutron-nucleus σ_{tot} as calculated with this algorithm (solid line) and with the improved algorithm [55] (dashed line).

2.2.4 Photon-nucleus cross sections.

Data compilation and interpolation algorithm for $\sigma_{\gamma N}$ with phenomenological A-dependence for $\sigma_{\gamma A}$ are as described in [58].

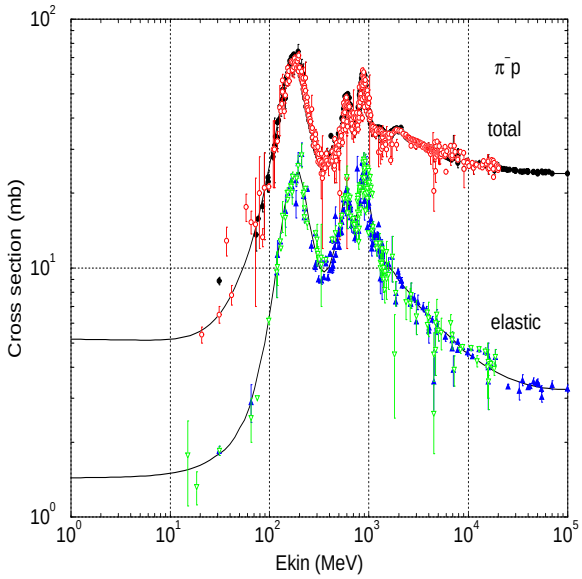


Figure 3: MARS cross sections in comparison with experimental data: σ_{tot} and σ_{el} for $\pi^- p$ collisions as a function of pion kinetic energy

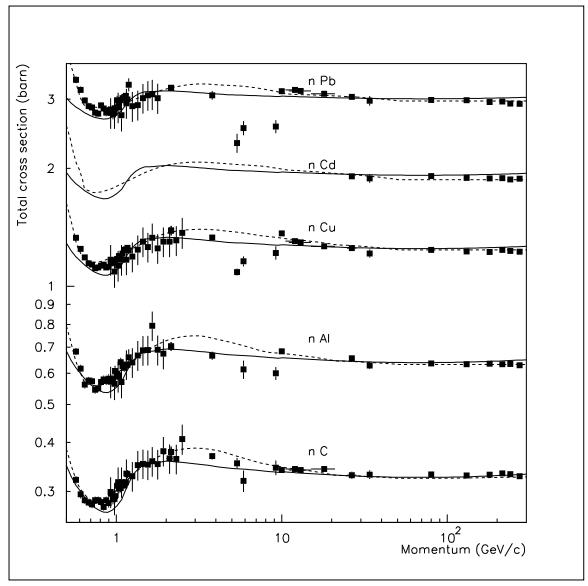


Figure 4: MARS cross sections in comparison with experimental data: σ_{tot} for neutrons *vs* beam momentum.

2.3 Hadron Production

This section needs more detail, enough that the casual reader gets a better idea of what the code does - a short version of sections 2 and 3 from Conf-04-053, appropriately updated, would work.

2.3.1 Cascade-exciton model code

A version of the Cascade-Exciton Model of nuclear reactions [49] as realized in the code CEM2003 [51] and containing also several recent refinements [50] is implemented as default for 1 MeV $<$ E $<$ 5 GeV. The 1994 *International Code Comparison for Intermediate Energy Nuclear Data* has shown that CEM95 adequately describes nuclear reactions at intermediate energies and has one of the best predictive powers for double differential cross sections of secondary particles as compared to other available models. Besides that, it adds to MARS reliable π^--capture description (with a few modifications, e.g., radiative capture $\pi^- p \rightarrow n\gamma$), better description of photon induced reactions in the intermediate energy range and of radionuclide production. When installed within MARS, the CEM code was converted to double precision along with some other necessary modifications, but is otherwise identical to the CEM2003 version. Several examples of the CEM predictions compared with experimental data and results of several other models are given in Figs. 5 and 6. One can see that on the whole, the code reproduces quite well not only spectra of secondary nucleons but also excitation functions for the spallation yields, a much more *difficult* characteristic of nuclear reactions to be predicted by any theory, and is consistent with other well-known models [59].

2.3.2 Inclusive hadron production from 5 GeV to 100 TeV.

An inclusive approach is the default in this energy range. The hadron production model uses a combination of phenomenological models, parameterizations and integration algorithms, covering a hadron kinetic energy range of 1 MeV$<$E$<$100 TeV as described in Refs. [42, 60, 61]. An improved phenomenological model has recently been developed and introduced into MARS as the default to describe pion production in high-

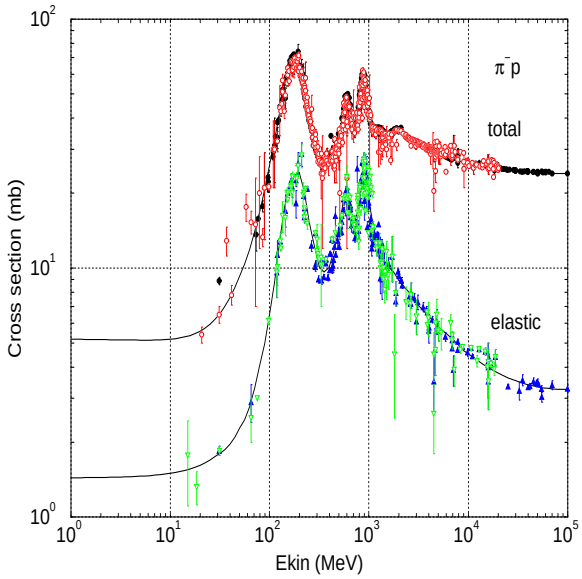


Figure 5: missing figure from CEM section

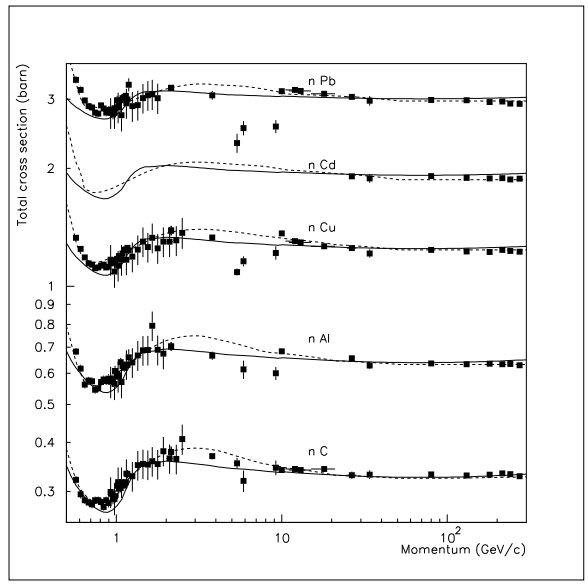


Figure 6: missing figure from CEM section

energy proton-nucleus interactions [62]. Special attention is paid to high-p$_t$ events, π^0 and kaon production, and low-momentum pions (p<2 GeV/c) from intermediate incident proton momenta (5< p_0 <30 GeV/c). The following form is used for the double differential cross section of the $pA \to \pi^{\pm}X$ reaction:

$$\frac{d^2\sigma^{pA\to\pi^{\pm}X}}{dpd\Omega} = R^{pA\to\pi^{\pm}X}(A, p_0, p, p_{\perp})\frac{d^2\sigma^{pp\to\pi^{\pm}X}}{dpd\Omega} \tag{1}$$

where p and $p_{\perp}$ are total and transverse momenta of $\pi^{\pm}$, and A is the atomic mass of the target nucleus. The function $R^{pA\to\pi^{\pm}X}$, measurable with much higher precision than the absolute yields, is almost independent of $p_{\perp}$ and its dependence on p_0 and p is much weaker than for the differential cross-section itself. Rather sophisticated algorithms have been developed to treat this function for pion production on nuclei in the forward (x_F >0) and backward (x_F <0) hemispheres separately. It is demonstrated in [62] that model predictions are in a good agreement with data in the entire kinematic region. Typical examples of comparison with data are shown in Figs. 7 and 8. Calculations with the MARS13(98) code of the pion double differential spectra from a thick lead target at p_0=8 GeV/c agree reasonably well with data [63] in the *difficult* momentum region 0.5< p <5 GeV/c, Fig. 9 whereas GEANT seems to have a problem. Information on the nuclides generated in nuclear collisions is now scored, or accumulated, and reported in the output results.

2.3.3 Deuteron-nucleus collisions.

Deuteron interactions have little in common with the general picture of the interaction between complex nuclei because of the deuteron's relatively large size and small binding energy. Therefore a special model has been developed [64] and implemented into MARS. Deuteron-nucleus interactions are classified as elastic, dissociation, stripping, and full inelastic. In elastic interactions the deuteron emerges intact in the final state while the nucleus may be unchanged (coherent elastic) or have lost one nucleon (incoherent). Coherent elastic uses Glauber's treatment with some adjustments of the parameters to fit experiment. Incoherent elastic scattering assumes a differential cross section to be twice that of the proton – using the prescription of [65] – and the nuclear parameters as for the coherent case. This is then multiplied by a deuteron – and a nucleon form factor as well as a Pauli suppression factor which hinders low momentum transfers. Exchange of a long range virtual photon may result in Coulomb dissociation whereby the deuteron splits into a proton

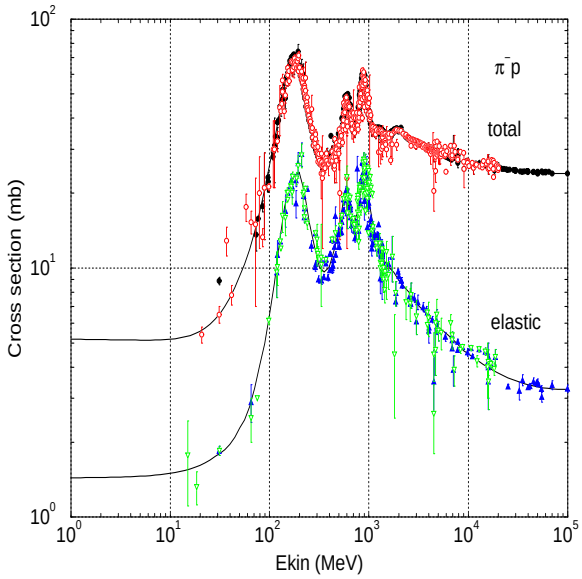


Figure 7: missing figure from hadron production section

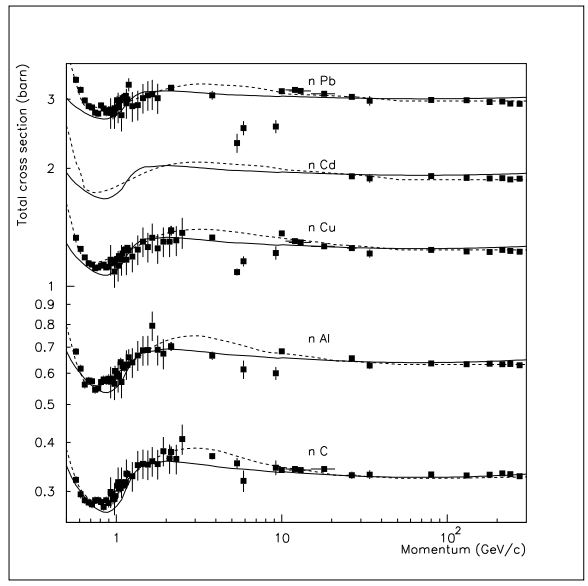


Figure 8: missing figure from hadron production section

and neutron. This is calculated using a Weiszacker-Williams approach for virtual photon emission. Dissociation may also result from (nuclear) elastic processes at relatively high momentum transfers. In stripping one nucleon undergoes an inelastic nuclear event while its partner continues without interaction. The total stripping probability is calculated based on the projected n–p separation as predicted by the deuteron wave function [66] and geometrical arguments. Deuterons dissociate as in [67] with full relativistic kinematics. Interaction with the nucleus by one of the partners proceeds within the standard MARS scheme. In full inelastic events both nucleons interact with the nucleus. The stripping routine provides the angular deflection and momentum of each nucleon after which both are allowed to interact as other MARS nucleons. As an example, a calculated π, K-meson yield out of a 3-cm radius gallium target 36-cm long in a 7.5-cm radius solenoid (B=20 T) is presented in Fig. 10 for proton and deuteron beams of equal momentum per nucleon.

2.3.4 Quark-Gluon String Model code LAQGSM2010.

Exclusive particle event generator for hadron and heavy-ion nuclear interactions from 1 MeV to 1000 GeV. Comes here !!!

2.3.5 Exclusive hadron production from 5 GeV to 100 TeV.

The DPMJET-3.0 code [68, 69] is implemented into MARS to sample the initial hN, hA, AA and νA interaction for 5 GeV<E<100 TeV [42]. This provides – at least partially – the features of a full exclusive event generation with all known particles in a final state, and is available as a non-default option which the user must specify, see Section 8.2 for instructions. The DPMJET code has been proven to be consistent with collider and cosmic ray data in a multi-TeV energy region. The two-component Dual Parton Model is used with multiple soft chains and multiple minijets at each elementary interaction. Within this model the high energy projectile undergoes a multiple scattering process as formulated in the Glauber approach. Particle production is realized by the fragmentation of colorless parton-parton chains constructed from the quark content of the interacting hadrons. The code includes cascading of secondaries – suppressed by the formation time concept – within both target and projectile nucleus. The excitation energies of the remaining

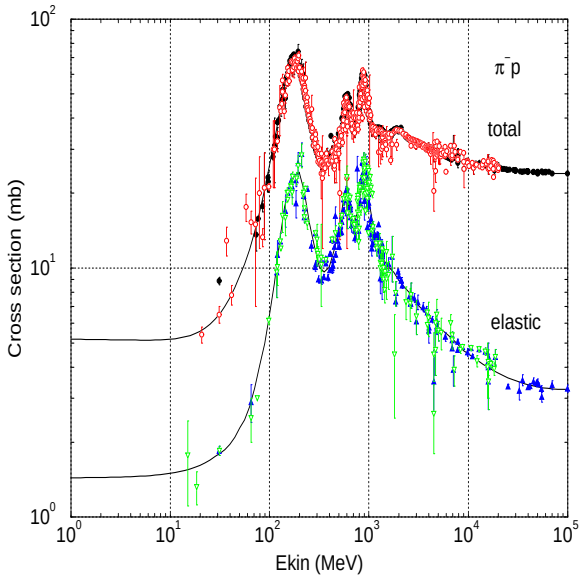


Figure 9: missing figure from hadron production section

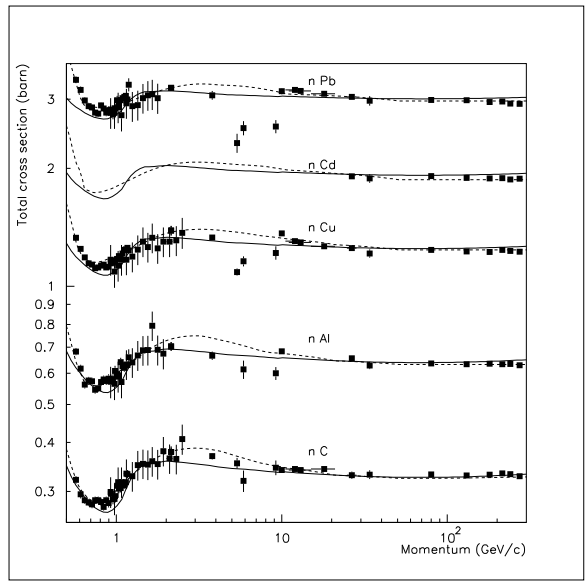


Figure 10: missing figure from deuteron-nucleus collision section

target – and projectile nuclei are calculated and simulation of subsequent nuclear evaporation is included in the model.

2.4 Elastic Scattering

Modern evaluated nuclear data as well as fitting formulae are used to simulate hadron-nucleus elastic scattering. For protons, nuclear Coulomb elastic scattering and their interference is taken into account. At E>5 GeV, a simple analytical description used in the code for both coherent and incoherent components of $d\sigma/dt$ is quite consistent with experiment.

2.5 Muon Production

Simulation algorithms for $\pi^{\pm}\rightarrow\mu^{\pm}\nu(\overline{\nu})$ and $K^{\pm}\rightarrow\mu^{\pm}\nu(\overline{\nu})$ decays and for prompt muon production (single muons in charmed meson decays, $\mu^{+}\mu^{-}$ pairs in vector muon decays, and the dimuon continuum) with forced generation of weighted muons have been improved. Prompt muons produced in electromagnetic showers are described in detail.

- Bethe-Heitler pairs $\gamma Z\rightarrow Z\mu^{+}\mu^{-}$ are produced at $E_{\gamma}\geq 0.25$ GeV at a rate of $(m_e/m_{\mu})^2$ times that for $e^{+}e^{-}$ with the appropriate statistical weights and a complete simulation of electromagnetic showers. It was shown [70] that this approach produces remarkable results that agree with those based on the numerical integration in the Tsai formalism [71].
- muon production with forced decays of mesons and short-lived resonances [24, 26, 72, 73];
- very efficient algorithms for muon interactions (ionization, bremsstrahlung, direct $e^{+}e^{-}$ pair, and deep inelastic) and transport [74] well advanced compared to previous versions [35, 26, 72, 73];
- optional forced $\mu\rightarrow e\nu\overline{\nu}$ decays and synchrotron radiation generation [75];

- At $E \geq 45\,\mathrm{GeV}$ direct positron annihilation $e^+e^- \rightarrow \mu^+\mu^-$ is simulated according to [76] with $\sigma = 86.8/s$ nb, where s is in GeV^2, and with $(1 + cos^2\theta)$ as the angular distribution.
- For $\mu \rightarrow e\nu\overline{\nu}$ decays, the vector momenta of the emitted electrons and neutrinos are sampled according to the differential decay probability of the *Vector-Axial* model of four-fermion interactions,[76].

2.6 Electromagnetic Interactions of Heavy Particles

2.6.1 Electromagnetic interactions of muons, charged hadrons and heavy ions

in arbitrary composite materials are simulated down to several tens of keV. Radiative processes and atomic excitation and ionization with energy transfer ϵ greater than a cutoff ϵ_c are considered as discrete events involving production of δ-electrons, e^+e^--pairs, and bremsstrahlung photons which are followed explicitly [77]. Energy losses with $\epsilon < \epsilon_c$ (so-called restricted losses) are considered as continuous. Their distribution is described by Vavilov function – with redefined parameters – which approaches a Gaussian with decreasing ϵ_c. Independent of energy, material or thickness traversed, the quality of the Gaussian approximation is governed by the average number of events (κ_n) one chooses to evaluate individually and becomes acceptable for most purposes when κ_n >10. Bremsstrahlung and direct e^+e^- production differential cross-sections used in the code are as given in Ref. [78].

2.6.2 Multiple Coulomb scattering

is modeled from the Moliere distribution with nuclear form-factors included [79]. A very careful treatment is done in MARS of processes near and below the Coulomb barrier in hadron, muon and heavy ion transport (ionization absorption vs nuclear interaction vs decay) as is further described in Ref. [42]. The scattering is applied as a continuous process while the particle passes through material, rather than being applied at discrete locations.

2.7 Electromagnetic Showers

New modules for simulating electromagnetic showers based on current knowledge of physics of electromagnetic interactions were recently developed and have been implemented into the code [80]. The main focus is given to electron and photon interactions in arbitrary composite solid, liquid and gaseous materials at low energies (1 keV to a few MeV). The entire shower, and such processes as emission of synchrotron photons, photohadron production, $\gamma Z \rightarrow \mu^+\mu^-$ and $e^+e^- \rightarrow \mu^+\mu^-$, can be treated – in the spirit of the MARS framework – either analogously or inclusively with corresponding statistical weights. The choice of method is left for the user to decide, via the input settings.

following statement needs more detail Generation and transport of de-excitation photons is improved. Undesirable fluctuations in the inclusive description of electromagnetic showers are reduced.

2.8 Synchrotron Radiation *(text to be added)*

2.9 Stopped Hadrons and Muons

A very careful treatment is done in MARS of processes near and below the Coulomb barrier in hadron and muon transport (ionization absorption vs nuclear interaction vs decay).

2.9.1 Pions.

A stopping π^+ decays into μ^+ of 4.1 MeV plus a neutrino while a π^- attaches to a nucleus (via the modified Fermi-Teller law). While cascading down the atomic energy levels, the pion is captured from a high orbit thus emitting only a few low energy photons which are neglected here. The hadronic interaction of the stopped π^- is treated using the Cascade-Exciton Model [51] with a few modifications. When hydrogen is the target it is assumed there is a 60% probability to for charge exchange ($\pi^- p \rightarrow \pi^0 n$) whereupon the π^0 decays into two photons of 68.9 MeV each and the neutron acquires a small (0.4 MeV) kinetic energy. The remaining 40% of stopped π^- in hydrogen interact via radiative capture: $\pi^- p \rightarrow n\gamma$. Here the photon acquires 129.4 MeV and the neutron 8.9 MeV kinetic energy. Other nuclides have a much smaller probability for radiative capture (1–2% which is taken into account in competition with CEM95). The photon energy is chosen from an empirical fit to experiment while the remainder is deposited as excitation energy.

2.9.2 Muons.

A stopping μ^+ always decays into $e\nu\overline{\nu}$ while a μ^- attaches itself to a nucleus. When a μ^- stops in a compound or mixture one first decides to which nucleus the μ^- attaches (modified Fermi-Teller law). Following attachment the muon may still decay as decided by comparing capture and decay lifetimes of which the latter is favored for light nuclei (Z$\leq$11). A captured μ^- then cascades down to the ground state of the muonic atom emitting photons along with some Auger electrons, all of which is simulated using approximate fits to the atomic energy levels. In hydrogen muon capture always produces a 5.1 MeV neutron via inverse β-decay. In complex nuclei the giant dipole resonance plays a role and results in an 'evaporation'-type neutron spectrum with one or more resonances superimposed. This is illustrated in Fig. 11 which shows the neutron spectrum resulting from μ^- capture on oxygen. In addition smaller numbers of evaporation-type charged particles and photons may be emitted. Calculated with the above algorithms longitudinal dose distributions in a slab tissue-equivalent phantom are shown in Fig. 12 at the axis of 150 MeV proton and 75 MeV pion, muon and neutron beams striking the phantom.

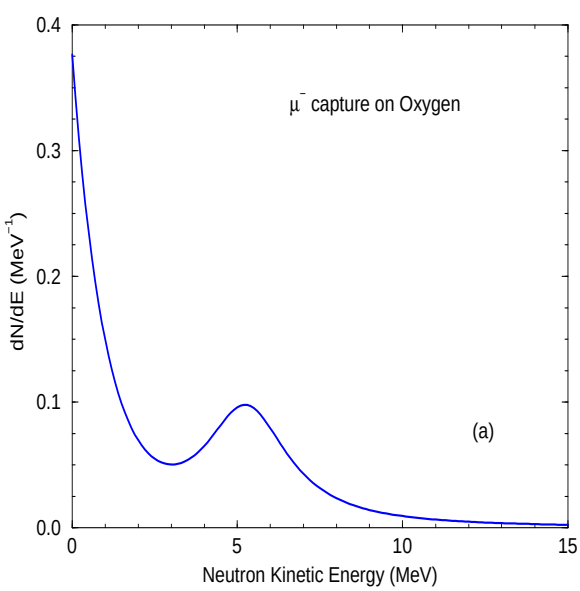


Figure 11: Neutron spectrum generated in a μ^- capture on oxygen atom

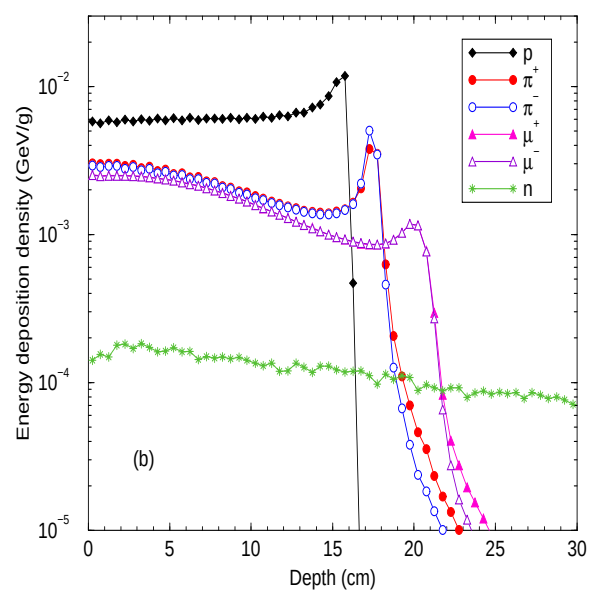


Figure 12: Axial absorbed dose in a tissue-equivalent phantom for 150 MeV (p) and 75 MeV ($\pi^\pm$ $\mu^\pm$ n) beams with 1$\times$1 cm transverse dimension.

2.9.3 Antiprotons.

Stopped $\overline{p}$ attach to nuclei in the same way as π^- or μ^-. Annihilation at rest is assumed to produce only pions, neglecting some of the rarer modes involving strange particles. Charges of produced pions are slightly skewed towards π^- in view of the 'brought in' negative charge. Pion momenta are chosen from an inclusive distribution loosely based on experiment. The energy weighted distribution is normalized to twice the nucleon mass which predicts a multiplicity of 4.3 – close to observation. In a complex nucleus the annihilation is treated as though it occurs on free nucleon except that each pion produced by the annihilation process is given a 50% probability to interact within the nucleus. This shortcut attempts to include – at least qualitatively – participation by the constituents nucleons.

For antiprotons in flight the annihilation cross section results in a larger cross section for $\overline{p}A$ vis-a-vis pA, especially for light nuclei at lower energies. Total cross sections for both $\overline{p}A$ and pA are estimated on the basis of simple geometrical considerations and $\overline{p}p$, $\overline{p}n$ and pp,pn cross sections. The ratio $\sigma_{\overline{p}A}/\sigma_{pA}$ is then applied to the more accurate σ_{pA} used in MARS. Annihilation in flight uses the same inclusive pion distribution as at rest in the $\overline{p}$-nucleon rest frame after which the pions are Lorentz transformed back to the lab. Above about 0.1 GeV/c a small $\overline{p}p \rightarrow \overline{n}n$ component is included. For both mechanisms *nuclear* target effects are again approximated by allowing emerging particles to interact in the same nucleus or escape each with one half probability. There is also added a third component in which the $\overline{p}$ or $\overline{n}$ interact only quasi-elastically with the nucleons. These are simulated using conventional MARS algorithms exactly as for protons except that the fastest nucleon emerging (leading particle) from the collision is identified as its antiparticle.

2.10 Neutrino Interactions

A special weighted neutrino interaction generator has been developed [42, 81] and incorporated into MARS. This model permits the selection of the energy and angle of each particle (ν, e, μ and hadrons) emanating from a simulated interaction. These particles, and the showers initiated by them, are then further processed in the code in the usual way. Four types of neutrinos are distinguished throughout ($\nu_\mu, \overline{\nu}_\mu, \nu_e, \overline{\nu}_e$) and the model identifies all possible types of neutrino interactions with nuclei. The corresponding formulas for these processes as well as results of Monte Carlo simulations for muon colliders and storage rings are described in [81].

2.11 Low Energy Neutrons

The default low energy ($E \leq 0.0145 GeV$) neutron model in MARS is sufficient for most applications. It uses a 28-group cross-section library, which in turn is derived from data on a set of 14 materials [BNAB]. The cross-sections are extrapolated to cover other materials not in the initial set of 14. Alternatively, one can use, so-called the MCNP mode in MARS, where neutron interactions at $E \leq 0.0145 GeV$ are described using the ENDFB cross-section libraries [82]. In this regime, the code models all physics processes from 0.001 eV to 0.0145 GeV such as $n - \gamma$ reactions, creation of recoil protons, heavier recoils, thermal neutron capture on ^6Li and ^{10}B; corresponding effects in hydrogenous, borated and lithium-loaded materials [83] etc.

In some cases more precise modeling of low energy neutrons is necessary, particularly when predicting residual dose rates, and when predicting prompt dose rates in labyrinth type geometries. Residual rates are sensitive to the materials present, and prompt dose rates to the $n - \gamma$ reactions, which occur when slow neutrons are stopped and captured. In these cases, MARS can be used coupled with the full MCNP code, in a combined MARS-MCNP mode. In this mode, MARS generates the initial neutron but sends it to the full version of MCNP for tracking and interactions, to the point where the neutron is captured or to it's cutoff

Table 1: Neutron energy group numbers `NG` and lower energy boundaries `EG` (MeV) in the group at $E \leq$14.5 MeV in the 28-group representation.

`NG=`	1	2	3	4	5	6	7
`EG=`	14.	10.5	6.5	4.0	2.5	1.4	0.8
`NG=`	8	9	10	11	12	13	14
`EG=`	0.4	0.2	0.1	4.65E-2	2.15E-2	1.E-2	4.65E-3
`NG=`	15	16	17	18	19	20	21
`EG=`	2.15E-3	1.E-3	4.65E-4	2.15E-4	1.E-4	4.65E-5	2.15E-5
`NG=`	22	23	24	25	26	27	28
`EG=`	1.E-5	4.65E-6	2.15E-6	1.E-6	4.65E-7	2.15E-7	2.15E-9

energy. At that point the particle information is handed back to MARS for compilation of tallies and results. The full MCNP code contains very detailed material handling, taking into account the nuclear physics details that occur in interactions with different nuclei, and this affects residual rate calculations. MCNP also includes the $n - \gamma$ reactions critical to predicting prompt dose calculations in labyrinth type geometries, where slow neutrons are dominant.

Using MARS with the full MCNP code requires that the user obtain and install the MCNP code library. The code is obtained from RSICC [84], the Radiation Safety Information Computational Center, at the Oak Ridge National Laboratory, on the web at www-rsicc.ornl.gov, or from the NEA Databank in Europe [85]. The code package is proprietary and a license must be purchased. For further details on using MARS in the MARS-MCNP mode, see Section 6.

3 Using the MARS Package

MARS uses it's physics models to generate particles and track them through a described geometry, applying appropriate physics interactions, and tabulating the particle flux, energy deposition, and other parameters in the various portions of the geometry. This Section describes how the MARS package functions, by giving overviews of : MARS geometric zones; the particle types used in the simulation and how MARS tracks these particles through a modeled geometry; how materials are defined; how results are tabulated and what histogram options are available; variance reduction techniques; and interfaces to other codes. Each overview directs the reader to further details in other Sections.

The units in MARS are: energy in GeV, dimensions in cm, angle ϕ in $degrees$, and magnetic fields in $Tesla$. The reference system is a cartesian coordinate system, with the z axis longitudinal, and the positive direction is from left to right. The z axis is usually the center of symmetry, and as a rule the primary particles strike along this axis in the positive direction. The positive direction of the x axis of the global system is up and the y axis is toward the viewer, completing a right-handed system.

3.1 Describing a Geometry by Zones

The geometry description, or model, is seen by MARS as a contiguous array of volumes, called zones. The user has complete control over the size, shape, content and location of these zones. As particles are tracked through the model, MARS queries specific subroutines to determine which zone the particle is in. Each zone is uniquely numbered, and the zone number corresponds to an array index used by many variables in the program. For example, there is an array, indexed by zone number, which tells MARS what material is present in each zone, and the material in turn is used to define manrrey parameters which affect the physics interactions which can occur within that zone. As particles are generated and tracked, the *path length* of each particle type within each zone is accumulated and stored; the final *path length* divided by the zone volume results in the flux for each particle type through each zone. Particle flux, tabulated by zone number, is used in turn to calculate many other results from the program. In short, by controlling the size, location and material specifications for the zones, the user controls the level of detail in the results which MARS reports. The user can also tweak parameters which control some of the interactions which MARS simulates.

MARS classifies geometry zones as one of three types: Standard, Extended and Non-Standard. Standard Geometry zones are those whose borders are defined by quantities entered in the `MARS.INP` input deck. Standard zones always have a cylindrical symmetry, with borders defined only on the z, r and ϕ axes; the syntax is described in Section 4.2.5. Extended Geometry zones are those whose borders are defined by the contents of the `GEOM.INP` input deck; the syntax for this deck is described in Section 4.4. Extended Geometry zones are similar to the geometry description used by the Geant Monte Carlo, in that they are constructed from a set of contiguous or overlapping geometrical shapes such as boxes, spheres and cones. Non-Standard Geometry zones are those described by the user's software encoded description of a geometry (there is no restriction on the type of software – while the MARS user subroutines are written in `FORTRAN`, the user can write his own modules in other languages, and have these called from the MARS subroutines). There is no restriction on the shape of Non-Standard zones; they can have cylindrical or cartesian symmetry, or can be of an arbitrary shape too intricate to render using the Extended Geometry syntax. Further details on setting up non-Standard zones is in Section 5.5.

Geometry descriptions can be imported from a few external programs, and these are referred to as imported zones, and MARS will treat these areas as non-Standard zones. See below in Section 3.1.2.

All geometry options can co-exist in a setup description. Only extended geometry option provides exact crossing of particle tracks with surfaces that prevents small regions within a large volume from being skipped over. In other geometry options, boundary localization is based on iterative algorithm

and user needs to take care of appropriate region numbering, pilot steps and localization parameters. Zone overlapping in the extended geometry mode means in particular that in this mode a user doesn't need to care about zone numbering when one large zone is filled with small objects.

Any type of zone can hold any material which the user has declared to be "in use" (see Sections 3.4 and 4.2.4). But there can be only one material in each zone. On the other hand, a magnetic field can vary from point to point within a zone. The difference between the requirements for materials and fields within zones has to do with how the tracking is handled, discussed below in Section 3.3.

The entire volume of the user's model must be filled by any combination of Standard, Non-Standard, Extended, or imported geometric zones; there can be no "holes" left undefined and yet surrounded by defined spaces. If such holes exist, then MARS will take care of defining them for you – by assigning it's own zone number to them and, if no other material is defined for that location, filling the areas with the material called "black hole". The material functions as named: any simulated particle which enters the volume of the hole will disappear, be dropped from the tracking list, and not propagated into the surrounding zones. In the GUI visualization of the model, these undefined spaces show up as black filled. For most applications, if such holes exist, the simulation results will not be correct. There are, however, certain applications where the user might desire to kill off particles which cross into a certain zone; the user subroutine `LEAK`, Section 5.12 gives an example.

The outermost extent of the user's entire model is a Standard zone cylinder whose extent is given by the values in the **ZSEC** and **RSEC** cards in the `MARS.INP` input file. This fact implies that there must be at least one Standard zone - a sort of container zone. All other zones, Standard, Extended and non-Standard (including imported), must fit within the Standard zone container or have outer boundaries aligned with it. For example, if the users model is defined entirely using software-encoded non-Standard zones, he must still have at least one Standard zone, and this Standard zone must be sized to surround the maximum geometrical extent of the encoded non-Standard zones.

The total number of all zones, Standard, non-Standard, Extended, and imported, has a maximum value of 100,001, and this limitation is set by the maximum size of the arrays used to store and accumulate the results of the simulation. The number of Extended zones must be between 0 and 50,000, but there is no similar limit on the number of other zone types, except the upper limit on the sum of all the types. It is up to the user to keep track of the numbers of the zone types declared, and to be certain they remain within the stated limits.

A single model description can use all zone types simultaneously, and these can even overlap in space, but doing so will cause MARS to use it's own priority system to determine the final zone number for the overlapped area. The priority to zone assignment is based on the order in which MARS calls the different geometry modules. When MARS has a coordinate position, and needs to know what zone that coordinate is located in, it first calls the non-Standard geometry module, via user subroutine `REG1`. Imported geometries are grouped with the non-Standard zones. If no zone is defined for the coordinate position there, then next the Extended Geometry module is called. If no zone is assigned there, then the Standard geometry module is called. If no zone is assigned to a coordinate position after all these modules are called, then the location becomes a "hole" in the geometry; however, because there is at least one "container" Standard zone, it typically serves as a background to all other zones, and errors in the non-Standard and Extended geometry descriptions can show up as being assigned to the "container" Standard zone rather than as a "black hole".

An interactive GUI interface is included within MARS, which allows the user to view his geometry, and check items such as zone boundaries, the material content of zones, and the zone index number assignments. An overview of the visual interface is given in Section 3.7; further details on using the interface are given in Section 12.

3.1.1 Geometric Zone Volumes

The boundaries of zones describe a geometry model to MARS, and give the program the information it needs to track particles and apply physics interactions within the model. But the boundaries alone are not sufficient to calculate results. In order to do that, MARS also needs to have the volumes of all the defined zones. As particles are generated and tracked, the *path length* of each particle type within each zone is accumulated and stored; the final *path length* divided by the zone volume results in the flux for each particle type through each zone. The particle flux is then used in turn to calculate many of the results reported by the various output files. A defined zone which has a zero volume will not only have no results calculated, but also the zone number will not appear in any of the output results lists.

So, the user must be certain that zone volumes are provided to the program. In the case of Standard zones, MARS is able to calculate *most* of the volumes automatically, since these zones have a simple cylindrical symmetry. However, a "container" Standard zone in which other types of zones are embedded will be "broken", i.e. it is no longer a cylinder, but a cylinder minus the shapes of the other zones inside of it, and in this case the volume for such a Standard zone will have to be provided to the program. In the case of non-Standard and Extended Geometry zones, the user must always provide the volumes. The volume data is provided to MARS via user subroutine `VFAN`, which enters zone volumes into the appropriate arrays. Calculating zone volumes "by hand" is an option but can be tedious for complex shapes. MARS has a "volume MC" running mode, where it "throws" coordinate positions to the geometry subroutines, and probes the boundaries of zones; as it does this, it forms a volume measurement of the zones. The resulting volume estimates are listed in the `VOLMC.NON` output file. The user then cuts and pastes the volume values into the `VFAN` subroutine. Section 4.2.2 gives the `MARS.INP` settings for running in "volume MC" mode, Section11.1.1 describes the contents of the `VOLMC.NON` file, and Section 5.6 describes the `VFAN` routine and the process of obtaining zone volumes in more detail.

3.1.2 Geometries from other Codes

The MARS program can accept geometry descriptions created by a few other programs. At present, these programs are MCNP, FLUKA, and the MAD format. The MAD format is used by various accelerator design programs as a unified method to describe typical beamline elements. For MARS to use a MAD format file, the file must first be pre-processed through a stand-alone parsing program called the MAD-MARS **Beamline Builder**; it's use is described in Section 9. FLUKA geometry description files must also be pre-processed; in this case the parsing program is a script which is distributed with the MARS code and libraries, and its use is described in Section 7. Instructions for using MCNP geometry descriptions is in Section 6.1.2. For each of these, the processed geometry file from the external code is imported into MARS by appending the file contents onto the end of the main MARS input file, `MARS.INP`, and then certain flags are set on the **INDX** card so that MARS knows to examine this file past the **STOP** card. These geometry descriptions function in MARS just as the native geometry descriptions do – they describe an array of geometric zones, and these zones are treated just as any other defined MARS zones, including the volume requirement described above. In general, the zones created by these imported geometries are treated as non-Standard zones, and as such must follow the same numbering conventions and limitations on the total number of zones; the physics results for these zones will be listed as for all non-Standard zones.

3.2 Particles used in MARS

The list of particles which the MARS simulation produces, interacts and tracks through a model is given in Table 2. Heavy ions are also transported in the code. The mesons listed, π^+, π^-, K^+, K^-, muons, and the longer-lived particles such as K^0, Λ and other hyperons, are allowed to decay in flight according to their

lifetimes, if they survive interactions in matter between their production and decay points. Their main decay modes are included in the simulation. Short-lived particle types generated at an interaction point, such as π^0, ρ, ω, D, and J/Ψ, are forced to decay immediately, using their most probable branching fractions; these decays produce daughters which are listed in Table 2.

Each particle in Table 2 has an index associated with it. This index is used to refer to the particle type in various places in the MARS code. For example: the primary particle type can be specified from the input `MARS.INP` file using this index; the user histogram subroutines can utilize the index to accumulate data on a particular particle type. In the MARS common blocks and the user subroutines, the index value is generally assigned to variable `JJ`.

Table 2: Transported particle types and their corresponding index value.

1	2	3	4	5	6	7	8	9	10
p	n	π^+	π^-	K^+	K^-	μ^+	μ^-	γ	e^-
11	12	13	14	15	16	17	18	19	20
e^+	$\overline{p}$	π^0	d	t	He_3	He_4	ν_μ	$\overline{\nu_\mu}$	ν_e
21	22	23	24	25	26	27	28	29	30
$\overline{\nu_e}$	K_L^0	K_S^0	K^0	$\overline{K^0}$	Λ	$\overline{\Lambda}$	Σ^+	Σ^0	Σ^-
31	32	33	34	35	36	37	38	39	40
$\overline{n}$	Ξ^0	Ξ^-	Ω^-	$\overline{\Sigma^-}$	$\overline{\Sigma^0}$	$\overline{\Sigma^+}$	$\overline{\Xi^0}$	$\overline{\Xi^+}$	$\overline{\Omega^+}$
Heavy ions with $A > 4$ have the following indexes: $ID = 1000 * Z1 + A1 - Z1$									

3.3 Tracking

The methods used to simulate the passage of particles through matter are a large part of any physics Monte Carlo system and MARS is no exception. The path any particle takes through matter has both discrete and continuous components. An example of a discrete process is nuclear collisions; an example of a continuous processes is the trajectory of a particle through a magnetic field. In the MARS simulation, a particle's trajectory is approximated by a series of connected line segments. The possibility of a discrete interaction occurring, or the effect of a magnetic field, is evaluated at the ends of a segment; the length of each segment and the magnitude of the change in direction from segment to segment are what control the overall accuracy of the tracking simulation.

Therefore, the first thing MARS does when tracking a particle is determine the size of the next line segment. There are four main parts to this determination. First, MARS calculates the mean distance to the next discrete process, D_{disc}, where the set of these discrete processes consists of nuclear inelastic and elastic interactions, particle decay, and energy loss due to knock-on electrons. This distance depends upon the particle type and energy, and on the material the particle is passing through. Second, MARS calculates the step size D_{cont} allowed by the algorithm which models the main continuous process, the effect of magnetic fields. The step size for tracking through magnetic fields varies with the particle energy, as it is determined by limits on the bend angle; the MARS defaults for these limits are adequate for most situations, and can be modified by the user using the `ALMX` card in the input deck, described in Section 4.2.8. Third, MARS sets the overall interaction distance D_{int} by choosing the minimum of D_{disc} and D_{cont}.

The fourth part of the line segment size determination compares D_{int} to the step size allowed by the geometry description, D_G. The geometry portion of the tracking steps through the users model using a "pilot" step size, and after each such step, MARS queries the geometry description routines to determine what material (or zone number) is present at the end point of the step. If the material (or zone number) has not changed from the previous step then D_G is set to the pilot step size. If however the material (or zone number) has changed, then there are two possibilities. In the extended geometry mode (see Section 4.4), the code takes as the next step the distance to the nearest surface boundary by solving corresponding equations. In all other geometry modes, the code jumps back to near the start of the pilot step, checks to see that the initial material is located there, jumps forward again, with a reduced step size, checks to see if the material has changed, and so back and forth in smaller and smaller steps until the smallest allowed step size bridges across the boundary between the two materials. Figure 13 shows a simple schematic of this boundary localization procedure. D_G is reset at each iteration to a shortened step length which approaches or crosses the material boundary, and is compared to D_{int}. The boundary localization process halts when D_G is smaller than D_{int}, or when D_G has reached an allowed minimum size. If this allowed minimum step crosses the boundary into a new material, then D_{int} must be redefined, and the tracking sequence resets back to part one described above. The final tracking line segment length is the minimum of D_{int} and D_G. The current algorithm [15] for multiple Coulomb scattering correlated with the ionization energy loss provides accurate modeling on this segment independent of its size.

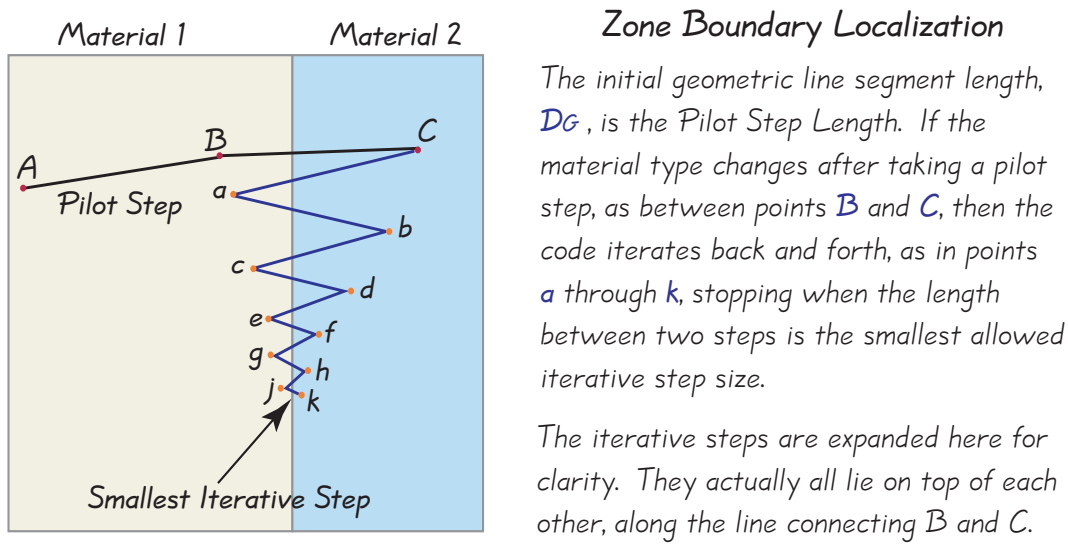


Figure 13: A schematic of the boundary localization procedure.

The accuracy of the geometric boundary localization process is controlled by the user, using the input deck as described in Sections 4.2.2 and 4.2.4. The pilot and minimum step sizes can be defined separately for different materials, using the *material dependent* cards. *Global* pilot and minimum step sizes are otherwise applied. Guidelines for setting the values of these step sizes to attain the desired accuracy are also discussed in Sections 4.2.2 and 4.2.4, and the example in Section 4.3.3 discusses in detail the correct application of the step size controls.

Another user control over the tracking is the particle threshold energy. As particles pass through matter they lose energy. The user can specify an energy below which particles will be dropped from the tracking; the main advantage of this feature is to reduce the computing time spent tracking particles which will have little or no effect on a particular result. The specification is made via the input deck. Just as for boundary localization, there are *global* and *material dependent* thresholds. The *global* energy thresholds are applied

in all materials, although there are several of them, so that each class of particle types can have a separate threshold applied, as described in Section 4.2.3. The *material dependent* thresholds are applied only where those materials are present, and there can separate thresholds for each class of particle type in each material, as described in Section 4.2.4.

Details on the development of MARS particle tracking methods can be found in references [26, 86, 87].

3.4 Materials

The user declares what materials are present in his simulation via the input file `MARS.INP`. The syntax is described in Section 4.2.4 and utilizes a 2-4 character string as an identifier. A maximum of 100 materials can be used within any simulation. An index, 1:100, is assigned to each declared material according to the order in which they are listed by the user in the `MARS.INP` file, and the user must use this index when assigning a specific material to a geometric zone. There can be no more than one material within a single zone. A material can be declared multiple times, so that different step-size or energy thresholds can be applied to the same material used in different zones; an example of this is given in Section 4.3.3.

There are two special materials within the MARS code. The first one, $blackhole$ is not included in the total number of materials and not declared in the `MARS.INP` file. The program assigns a negative material index and a zone number ≤ 0 to $blackholes$, which are either all material outside of the simulated volume (zone number$= 0$) or any region inside marked with zone number$= -1, -2, -3$,... . Any particle entering a $blackhole$ zone is immediately killed: it will be dropped from the tracking list and it's parameters will not be accumulated to any built-in results or histograms. The negative zone number is also, however, a trigger to calling the user subroutine `LEAK`, which can then be used for special scoring or for forming a source term. For example, a $blackhole$ zone is created in a particular quadrant of a geometry, so that every particle entering it can be recorded onto a special set of histograms defined by the user. Or, past a certain $Z-$plane a $blackhole$ zone is defined, and any hadron crossing this plane gets recorded to an output file, which is then in turn used as the primary beam input for a separate run of MARS which contains the geometry downstream of the one used in the first run; breaking the model up into two or more such stages can save CPU-time by dropping uninteresting particles at strategic points. Details and examples are given in Section 5.12 which describes the `LEAK` subroutine. **The easiest way to convert any material in `MARS.INP` to a $blackhole$ is to define its density on the `MTDN` card to be $\geq 10^{20}$.**

The second special material is $vacuum$. MARS uses material index `IM`$= 0$ for the global vacuum, where global means that the global values for tracking step lengths and energy thresholds are applied. If the user wishes to control these values for areas of vacuum, then the material name `VAC` can be used in the **MATR** input card list. This material gets assigned an index `IM`> 0 just as for other materials, and then the user can assign different step parameters or cutoff energies to different vacuum regions.

The first 100 elements in the periodic table, and many common compounds used in particle and nuclear physics, are built-in to the MARS code. These are listed in Tables 3 and 4. The user can also define his own compounds, or mixtures, as described in Section 5.3. All compounds are defined through the weight or atomic fractions of the individual elements they consist of. At program startup, MARS calculates the cross-sections and other physics interaction parameters for the list of compounds declared for use in the simulation. The precise effect of individual elements in compounds is taken into account for all the electromagnetic and nuclear elastic and inelastic processes modeled by MARS; material averaging, by using an effective overall Z and A, is no longer performed, as was done in MARS-13-95 and earlier.

It is recommended that the user replaces air and other gases with vacuum if a thickness of gaseous regions is less than about 10-30 cm. A CPU-performance and even transport accuracy would be substantially better with such a replacement with no effect on the oveall results. Of course, it makes

Table 3: Built-in elements. The first 100 elements in the periodic table are built-in, and listed in order of atomic number in the table below. Their MARS abbreviation is the same as their standard abbreviation.

H	HE	LI	BE	B	C	N	O	F	NE
NA	MG	AL	SI	P	S	CL	AR	K	CA
SC	TI	V	CR	MN	FE	CO	NI	CU	ZN
GA	GE	AS	SE	BR	KR	RB	SR	Y	ZR
NB	MO	TC	RU	RH	PD	AG	CD	IN	SN
SB	TE	I	XE	CS	BA	LA	CE	PR	ND
PM	SM	EU	GD	TB	DY	HO	ER	TM	YB
LU	HF	TA	W	RE	OS	IR	PT	AU	HG
TL	PB	BI	PO	AT	RN	FR	RA	AC	TH
PA	U	NP	PU	AM	CM	BK	CF	ES	FM

The following elements, used with their standard abbreviations, are treated by MARS as being in gaseous phase, with density at STP

H	HE	N	O	F	NE	CL	AR	KR	XE	RN

The same elements used with the following abbreviations are treated by MARS as being in liquid phase with the stated densities

LH	LHE	LN	LO	LF	LNE	LCL	LAR	LKR	LXE
0.0708	0.125	0.807	1.141	1.507	1.204	1.574	1.396	2.418	2.953

Carbon, element C has a default density of 2.265 g/cm^3. Carbon can also be declared as graphite by using GRPH, which has a default density of 1.7 g/cm^3

sense only if studying effects specifically in the gaseous regions is not the purpose of the run.

The calculated cross-section tables, as well as a summary of the composition and properties of all declared materials, is written to the output MARS.OUT file. If the user has any questions about the detailed properties of a built-in material, he can declare the material, and run MARS for a single event, and then examine the information in the MARS.OUT file. The default densities for most elements are those at standard room temperature and pressure.

The general chemical makeup of the built-in compounds is listed in Table 4, along with the resulting effective A and Z (note, effective values are not used by MARS, given just for orientation). The built-in default densities for each compound is also listed. The user can modify this density using the **MTDN** card in the MARS.INP file as described in Section 4.2.4.

Table 4: Built in compounds, with their default density values.

Abbr	$\rho(g/cm^3)$	Compound Name/Formula
VAC	0.	Vacuum
CH2	0.950	Polyethylene CH_2
BCH2	0.950	Borated Polyethylene
CH	1.032	Polystyrene
TEFL	2.200	Teflon, CF_2

continued on next page

Table 4: *table continued*

Abbr	$\rho(g/cm^3)$	Compound Name/Formula
NAI	3.670	Sodium Iodide crystal
ELEC	1.034	Electronics N-Si-Pb mixture
BGO	7.100	Bismuth germanate $(Bi_2O_3)_2(GeO_2)_3$
PBWO	8.280	$PbWO_4$
G10	1.700	NEMA G10 $60\%SiO_2 40\%$ epoxy
QUAR	2.640	Quartz, SiO_2
TISS	1.000	Tissue, $H_{40}C_4NO_{17}$
WATR	1.000	Water, H_2O
PARA	0.930	Parafin wax $CH_3(CH_2)_{23}CH_3$
AIR	0.0012	Air at $18°C$ and 58% humidity
SOIL	1.900	Soil, $H_{17}O_{27}Al_2Si_9$
CONC	2.350	Concrete + steel reinforcing, $O_2SiCaNaFeAl$
BART	3.200	Boron-Barite Concrete
BMCN	3.637	Borated Magnetite Concrete
STST	7.920	Stainless Steel 304
SCON	7.000	Super Conductor $90\%(60\%Cu + 40\%NbTi) + 10\%$Kapton
YOKE	7.870	Steel magnet yoke
COIL	5.108	Water-cooled copper coil
INSU	1.690	Insulation
CABL	2.896	Cables
CH4	0.4224	Methane CH_4
MYLR	1.390	$C_5H_4O_2$
MYAL	1.783	Aluminized Mylar
KAPT	1.420	Kapton $C_{22}H_{10}N_2O_5$
CRMC	3.970	Ceramic, Al_2O_3
TIAL	4.430	Ti-alloy, $0.9Ti + 0.06Al + 0.04V$ by weight
INCO	8.190	Inconel alloy 718
FECO	8.000	$0.64Fe + 0.36Co$ by weight
NBSN	6.802	Nb_3Sn SC coil
SCC	8.500	$0.24Nb_3Sn + 0.70CuSn + 0.06Ta$
GFRP	5.682	0.33STST$+0.21Cu + 0.1$SCC$+0.2He + 0.16$G10
BITR	8.146	Magnet bitter coil, $0.9Cu + 0.1$water
HOLW	5.50	Hollow conductor, 0.19water$+0.25MgO + 0.56Cu + 9\%void$
DLMT	2.85	Dolomite with 15% water by volume
GTIL	2.90	Glacial till with 30% water by volume
SHTC	2.24	Shotcrete
NYLN	1.13	Nylon, Type 6, $NH(CH_2)_5CO = H_{11}C_6NO$
NDFB	8.00	$NdFeB$ magnet, $0.06B + 0.82Fe + 0.12Nd$
LIH	0.82	LiH
MRBL	2.70	Marble $CaC)_3$
S316	7.92	Stainless Steel 316
S347	7.92	Stainless Steel 347
STCA	7.82	Carbon Steel
CAST	7.31	Cast Iron

continued on next page

Table 4: *table continued*

Abbr	$\rho(g/cm^3)$	Compound Name/Formula
NBS2	7.00	Nb_3Sn SC cable
ISOB	0.002493	Isobutane C_4H_{10} (gas)
OILM	0.769	Mineral Oil
ACRL	1.1388	Acrylic
SCI	0.7903	Scintillator
SCIG	1.3753	Gadalinium Loaded Scintillator
DEUT	0.1624	Deuterium (liquid)
TRIT	0.0709	Tritium (gas)
D2O	1.0177	Heavy Water
BEO	3.01	Beryllia Ceramic
CSI	4.51	CsI Scintillator
SMCO	8.2	$SmCo_5$ magnet
FERR	5.0	$SrO * 6Fe_2O_3$ Ferrite

3.5 Tabulation & Results

Results are accumulated, or tallied, zone by zone in the modeled geometry. The zones can be volumes defined by the user's encoded geometry, or volumes and surfaces where histograms have been defined. A description of standard mars output information and how to use the results is given in Section 11.1.2.

Geometry and phase-space tagging options, used intensively in studies with MARS (see, e. g., [88, 89, 90]), have been further improved. These allow a very efficient way to study a source term.

Calculated are three-dimensional distributions of star density, total and partial particle fluences above several thresholds, total and partial energy deposition densities, temperature rise, dose equivalent, residual dose rates, displacements per atom (DPA), one- and two-dimensional mass and charge nuclide distributions. All of these are accumulated and presented with corresponding statistical errors as part of the standard MARS outputs in the files MARS.OUT, MTUPLE, MTUPLE.NON, MTUPLE.EXG, MTUPLE.MCNP. Energy spectra in the defined zones, tagged distributions, and some integral values are also produced

3.6 Histogram Options

The I/O sequence as well as the histograming for surface and volume detectors is substantially improved and extended.

MARS native and HBOOK-based histograming and analyses have been extended to include scoring of surface current, flux, dose equivalent, particle spectra and time distributions.

See XYZ-histograming section.

3.7 Visualization *needs further update*

See GUI Section. The interactive GUI interface is invoked by setting a flag in the main `MARS.INP` input deck. The interface is very useful to visually check the geometry of the model, before actually running the executable to generate events. The flag must be reset to run MARS in the event generating mode, as this mode and the interactive GUI mode cannot be run simultaneously. Details on using the GUI interface are given in Section 12.

3.8 Statistics Guidelines *needs further update*

Some of the Rules of Thumb should just be moved here.

3.9 Variance Reduction and Biasing *needs further update*

Algorithms for splitting and Russian roulette at hA vertices (statistical weights) and in particle transport (particle trajectories). Phase-space and particle type biasing, exponential conversion of path length. Mathematical expectation: for 'deep penetration' problems in complex highly non-uniform geometries, algorithms for scoring probabilities rather than real particle crossings or interactions now take into account all possible processes for both stable and unstable particles and charged as well as neutral hadrons [26].

The user can now choose between sampling and forcing π-, K- and μ-decays. Algorithms for splitting and Russian roulette at hA vertices and in particle transport are also further improved. For 'deep penetration' problems in complex highly non-uniform geometries, algorithms for scoring probabilities, rather than real particle crossings or interactions, take into account all possible processes for both stable and unstable

particles and charged as well as neutral hadrons. Use of accelerating field (RF-cavities) is now optional in the code.

3.10 Interfaces *needs further update*

Interfaces to the ANSYS code for thermal and stress analyses and to the STRUCT code for multi-turn particle tracking in accelerators have also been improved.

fusion with the STRUCT code [48] for multi-turn particle tracking in the accelerator lattice represented by an arbitrary combination of magnetic elements and transfer matricies allowing unified approach to beam loss and radiation effects studies at modern accelerators [91, 86, 92, 93, 94, 95];

3.11 Getting Started

This section presents an overview of the code and outlines how to create and run an executable. Details on the structure and content of the input deck are given in Section 4. Details on the usage of each of the user subroutines are found in Section 5. Details on the structure and content of the input and output histogram files are given in Section 10. Descriptions of the various output files are given in Section 11.

MARS is primarily supported to run on SunOS and Linux operating systems, with secondary support SGI-IRIX, IBM-AIX, DEC-ALPHA and HP-UX. The MARS code system consists of a few hundred `FORTRAN77` subroutines and a few CERN service *C*-routines, organized by functionality into several files. The source code file structure is listed in Table 5. The computing efficiencies of many of the MARS software algorithms have been improved and optimized over time. The code runs completely in the IEEE-754 double precision model, [96], except for the CERN HBOOK package. MARS utilizes a machine independent universal random number generator [97].

The user subroutines, in file `m1510.f`, are the collection of routines which can be customized for a particular simulation. While these routines are `FORTRAN`, the user can of course write his routines in other languages, and have them called from the appropriate locations within the `m1510.f` routines. Input decks are also made available to the user for customization; the input decks consist of format-free, plain text data cards, which are read by the CERN FFREAD package. The output of MARS consists of several files, a few of which are always created and others which are created only when requested by the user via the input decks; many are text based tables, some are HBOOK files, and some are a specialized format for use as input in multi-step jobs.

The code is installed at many accelerator laboratories and universities. The code must be installed by the author, on request, and is not available for self-installation via anonymous ftp. For the latest information, a list of installation locations, code status, platform availability, comments and other related questions visit the official MARS Web site http://www-ap.fnal.gov/MARS or contact the author at `mokhov@fnal.gov`.

In general, the MARS code system is installed into a directory structure as shown below:

```
dat/          linux/    sun/        GEOM.INP
GNUmakefile   MARS.INP  XYZHIS.INP  m1510.f
marsmain.f    xsdir
```

The user has access to the source code only of the user subroutines in file m1510.f; all other source code is built into libraries. `marsmain.f` is a short file which holds the `FORTRAN` Program statement, the names of the main input and output files, and the call to the `mars1510` master code:

```
      PROGRAM MARSMAIN
C..................................................
C     MARS15 CODE
C-----
C     LAST CHANGE: 01-FEB-2007 BY NVM
C-----
      IMPLICIT NONE

      INCLUDE 'fnames.inc'
C..................................................

      INPNAM        = 'MARS.INP'
      OUTNAM        = 'MARS.OUT'
      HBKNAM        = 'MARS.HBOOK'
      XYZTABNAM     = 'XYZHIS.TAB'
      DUMPNAM       = 'DUMP'
      MTUPLENAM     = 'MTUPLE'
      MTUPLEEXGNAM  = 'MTUPLE.EXG'
      MTUPLEMCNPNAM = 'MTUPLE.MCNP'
      MTUPLENONNAM  = 'MTUPLE.NON'

      CALL MARS1507

      STOP
      END
```

The user is free to change the input/output file names in `marsmain.f`. The user copies `marsmain.f`, `m1510.f`, `GNUmakefile`, and the input decks `MARS.INP`, `GEOM.INP`, and `XYZHIS.INP` along with the MCNP file `xsdir` to his/her own working area. The user customizes the input decks and the user subroutines as needed; the user is free to create additional subroutines called from the provided user subroutines. The `GNUmakefile` can be edited by the user to include the list of all the user's local subroutine files, so they will be compiled and linked to the MARS libraries to obtain an executable.

The executable is made by running the `GNUmakefile` script. This manual assumes the user is familiar with the syntax for compiling and linking source code using `gmake` or `make` on unix or linux systems. The `GNUmakefile` script in the installation directory uses compiler options appropriate for the code and for the operating system and compiler version on the machine where the code is installed. The `GNUmakefile` contains pointers to all the required external libraries, including those for graphics and for CERNLIB.

The directory sample above is located on a hypothetical cluster which contains various types of computer platforms. The directories named by platform type, linux/ sun/ irix/ hold the MARS object libraries compiled for that platform within a `lib/` subdirectory, and also hold a subdirectory of include files, `include/`, which contain MARS common blocks and variable declarations. On a single node or on a cluster of identical machines, there will be just one platform directory listed instead of the multiple platforms shown in this example.

Some geometric models which have a simple cylindrical structure can be described completely via the main `MARS.INP` input deck, using MARS Standard zones; an example of such a geometry is given in Section 4.3. The sample `MARS.INP` file contained in the MARS installation directory is fairly simple and lists the items which most users will typically customize. It also lists, on the 2nd line of the file, the full path, on the user's local system, to the `dat/` files in the installation directory, which the MARS executable will need to load at run time. Other geometric models can be adequately described using groups of boxes, spheres and other standard shapes, and these models can be created using MARS Extended Geometry zones via the `GEOM.INP` deck; an example of such a geometry is given in Section ref/ingeom. The sample `GEOM.INP` and `MARS.INP.exgeom` files are set up to illustrate the use of Extended zones (one must put `INDX 3=T`

to MARS.INP to execute the example.

Both these simple geometric models, using only the MARS.INP file, or the MARS.INP and GEOM.INP files together, do not require customization of the user subroutines. In these cases, however, the user must still compile a local copy of the user subroutines to obtain an executable, even if the routines have not been modified from their initial dummy state.

For some problems involving the production and interactions of low energy neutrons, the user might consider using MARS coupled with the MCNP code library. See the discussion in Section 2.11 for the relevant conditions. Details on running in MARS-MCNP mode are in Section 6.

The MARS executable can be run interactively, in the background, or submitted to a batch system. By default, the executable and the input MARS.INP and GEOM.INP files must all reside in the same directory. For batch running, when the executable and input files might not be located in the same directory, the paths to the input files can be modified via customization of the user routine MARS1504, see Section 5.2 for more information. The running time per event can vary widely, depending on the complexity of the user's model. The user is advised to try short trial runs of 100 to 1000 events, to make an estimate of the cpu-time required. The more events which are generated, the better the statistical error on the results will be. In general, any zone of interest should have a relative error of no more than 20% for the results in that zone to have a physical validity. More discussion on this subject is in Section 13.1.

The interactive GUI interface is invoked by setting a flag in the main MARS.INP input deck. The interface is very useful to visually check the geometry of the model, before actually running the executable to generate events. The flag must be reset to run MARS in the event generating mode, as this mode and the interactive GUI mode cannot be run simultaneously. Details on using the GUI interface are given in Section 12.

The default output files are MARS.OUT, MTUPLE, MTUPLE.EXG, MTUPLE.MCNP and MTUPLE.NON; if histograms were requested there will also be a mars.hbook file. MARS.OUT and MTUPLE list the results accumulated in MARS Standard zones, with some information on Non-Standard zones also contained within MARS.OUT. MTUPLE.EXG lists results accumulated in Extended Geometry zones. MTUPLE.NON lists results accumulated in Non-Standard zones. The format of these files is defined within non-user code, and so cannot be changed by the user. Details on the contents of these files, and other specialized output files, is found in Section 11.

Table 5: Source code organization.

File name	Content
marsmain.f	PROGRAM file
m1510.f	User routines
m15bldt.f, m15bnab.f	BLOCK DATA modules
m15cstuff.c, m15gui.c, m15mc.c	*C* service routines
m15in1.f, m15in2.f, m15out.f	I/O and initialization routines
m15mareg.f	the main event processing steering routines
m15tr.f, m15trems.f	hadronic, electromagnetic and heavy ion transport
m15trneu.f, m15trneu-mcnp.f	neutron transport routines
m15field.f	magnetic field and synchrotron routines
m15dedx.f	DE/DX routines
m15region.f, m15exg.f	zone identification routines
m15treem, m15eve.f, m15evepi.f, m15evtgen.f	event generator routines
m15elast.f	elastic collision routines
m15xsec.f	cross-section routines
m15ph.f	additional physics simulation subroutines
m15cem.f	CEM(2007) model routines
m15neutrino.f	neutrino transport and interaction routines
m15deutron.f	deuteron interaction routines
m15rad.f, m15omega.f	radiation interaction routines
m15hist1.f, m15hist2.f	histogram booking and entry routines
m15util1.f, m15util2.f	utility routines, including FFREAD
m15tuple.f, m15srcterm.f	output routines
m15treem-laqgsm.f,laqgsm1-6.f	LAQGSM routines

4 The MARS.INP and GEOM.INP Input Files

This section describes the main input files, sometimes referred to as *decks*, used by MARS. These are the *native* input files, as opposed to user-created input files which might supplement the user's own subroutines. The files described in this Section are the mandatory MARS Input file MARS.INP, and the optional Extended Geometry input file GEOM.INP. The MARS.INP file is used to define the main operating parameters for running the simulation – there are many switches and selections that can be set. The number and size of the Standard Zones are defined within the MARS.INP file. The file might also contain an optional section for importing geometry descriptions from other programs; Section 3.1.2 contains a brief description of these options and refers to the manual sections which contain further details for each. The optional GEOM.INP file is used to define Extended Geometry Zones.

Note that all the various geometry descriptions which MARS can utilize – Standard, Non-Standard, Extended, MCNP, imported – can be used simultaneously, and can co-exist and overlap in the same setup, in a single problem. However, it is the users job to number the defined zones appropriately and uniquely, and to be certain that the correct volumes for each zone have been defined. For more information see the introductory Section 3.1 and references therein.

The main MARS.INP deck controls many features in MARS, such as: the primary beam parameters, materials, energy thresholds, termination conditions, hA-vertices parameters, scoring parameters, and regions where the standard histograms are accumulated. Nearly all these features have default values built into the code; if the default values are appropriate for the user's model, then the control lines for those features do not need to be present in the MARS.INP file. Some problems cannot be described solely by information in MARS.INP, such as complex composite materials different of the built-in compounds, complex geometries, atypical primary beam distributions, or magnetic fields. In these cases, the appropriate user subroutines are customized to describe the problem's complexity, and the use of these is discussed Section 5. However, even with extensive code customization, the MARS.INP file parameters are a key method of specifying basic information. Section 13.2 discusses some of the 'switches' in the MARS.INP file which turn on or off various physics process, and the conditions under which one might wish to modify these.

Sections 4.1 and 4.2 cover all aspects of the MARS.INP parameters and syntax. Section 4.4 describes the use of the GEOM.INP file. Section 4.3 holds examples showing the use of some the more common MARS.INP parameters, and Section 4.4.1 has an example of Extended Geometry zones.

Other sections of this manual hold descriptions for other, optional, input files. Section 10.4 describes the use of the XYZHIS.INP file and the histograms it produces. Section 6 describes the inputs required when using the MARS-MCNP running mode. Section 9 holds instructions for generating input information using the MARS-MAD Beamline Builder. Section 7 describes how to process a FLUKA geometry file for use as input to MARS.

The more advanced user can create his own input files as required by a specific problem: special source terms (*e.g.* DPM.EVE for event generators, or BLOSS for beam loss distributions in a beam-line or accelerator lattice), magnetic field maps in detectors and accelerator elements (*e.g.* QUAD1.MAP), particle distributions produced by MARS in a previous job for multi-stage runs, and many various optics and lattice element files used by the MAD-MARS Beam-Line Builder. In many cases, the user will have to customize a user-subroutine to open and read these customized files, and then also ensure the data is placed into appropriate variables and made accessible to MARS.

4.1 Structure of the `MARS.INP` Input Deck

The `MARS.INP` file, sometimes called the *input deck*, is always required to run a MARS executable. At the very least, the `MARS.INP` file describes the geometry's Standard Zone(s) (there must be at least one!), and the list of materials used by the model. The input deck consists of what are called cards, following ancient `FORTRAN` conventions; a card is a line or series of lines in the `MARS.INP` file.

The name of the file is, by default, `MARS.INP`, but if the user wishes to change the file name, he must edit the corresponding line in the main program `marsmain.f`; see Section 5.1.

The units used by the input deck parameters, and indeed used throughout MARS are: energy in GeV, dimensions in cm, angle ϕ in $degrees$, and magnetic fields in $Tesla$. The reference system is a cartesian coordinate system, where the z-axis is longitudinal, and the positive direction is from *left to right*. The z-axis is usually the center of symmetry, and as a rule the primary particles strike along this axis in the positive direction. The positive direction of the x-axis of the coordinate system is *up* and the y-axis is *toward the viewer*, completing a right-handed system. Any other units used by parameters in the input deck will be specified in the list of input cards where appropriate.

The first two cards in the `MARS.INP` deck (lines in the file) are lines of text; the remaining are data cards. The order of the the data cards and their number is arbitrary. The only requirement is that the two text cards appear on the first two lines of the input deck, and that the **`STOP`** data card be present to terminate the sequence of data cards. Blank lines can be introduced anywhere to group similar cards together and organize the file. Any given line which starts with the character `C` will be ignored, just as in `FORTRAN`. In general any lines after the **`STOP`** card are ignored, unless the user has set switches for importing a geometry description from another program; in this case a block of lines is present after the **`STOP`** card – see Section 3.1.2 for more information.

The first line of the input deck is the Job Title card. It has a `FORTRAN` format of `A80`. The user can enter any text on this line, usually a short title to describe the problem, and perhaps the date, such as *Cu target in Area B with design 1-a shielding, March 15, 2004*. This line of text is read into character variable MTEXT in the code, and is written to the `MARS.OUT` output file.

The second line of the input deck is the Path card, and also has a `FORTRAN` format of `A80`. The card gives the full directory path to the location of the `dat/` directory in the MARS installation directory, and as such is case sensitive. Example: */home/mokhov/restricted/mars15/dat*. This line is read into character variable `direct` in the code, and is used in `OPEN` statements to read the files of cross-sections and other data which MARS requires. The sample input deck `MARS.INP` located in the MARS installation directory will have the correct full path for that system. If the path is incorrect, a message will be written to the screen file stating that the required data cannot be located.

The rest of the `MARS.INP` file consists of data cards. The CERN FFREAD package is used to read these data cards in the MARS routine `BEGINN`. The cards are format-free in the sense that they can appear in any order, and the qualifiers within each card can be listed in any order. However, there are structural conventions which must be followed for the cards to function properly.

The structure of all these cards is the same:

`KEYW` `a(1) a(2)...` N_{b1}`=b(1) b(2)...` N_{c4}`=c(4) c(5) ...`

`KEYW` is a keyword assigned to a group of variables `a(i)`, `b(j)` and `c(k)`. The variables are mapped onto a one dimensional array `A(i+j+k)`. Within array `A`, N_{b1} is the location where element `b(1)` is located; N_{c4} is where element `c(4)` is located. The numerical values listed after **`KEYW`** will be loaded sequentially into array `A`, until a "N=" is encountered. Once detected, the filling of array `A` will jump to the location

given by N, and continue filling sequentially from there.

For example, there is a data card with keyword **ENRG**, which controls the primary energy and energy threshold cutoffs, below which particles will not be tracked. **ENRG** is assigned to nine variables. All of these thresholds have default values. For this example, all of the defaults are satisfactory for a given model, except the photon energy cutoff, which the user would like to raise from the default 0.0002 GeV to 0.001 GeV. That variable, EMIGA, is the 6th of 9 listed. The user would enter

ENRG $6 = 0.001$

into the input deck. This directive will change the default value of the 6th variable to 0.001 GeV.

If the user next wished to also change the default value on the 4th variable (EMCHR, the cuttoff on charged hadrons and muons), then this card entry would become

ENRG $4 = 0.03 \quad 6 = 0.001$

A space must separate the two equality statements, but that is all that is required.

Finally, if the user decided to additionally change the default on the 5th variable (EMNEU, the cuttoff on neutrons), then the card becomes

ENRG $4 = 0.03 \quad 5 = 0.05 \quad 6 = 0.001$

which is equivalent to

ENRG $4 = 0.03 \quad 0.05 \quad 0.001$

For the previous example all the variables were one-dimensional arrays. The example would appear the same if the receiving variable was a 3-dimensional variable, only one has to think about the FORTRAN conventions in the order in which the elements of arrays are filled. An example of cards which are assigned multiple variables with dimensions > 1 is given in Section 4.3.

The variables associated with a card can be integer, real, logical (represented by T or F), and, in one case, character. In general, all the variables assigned to a card are of the same type, but there are a few cards of mixed integer and real variables. The numerical values for real variables must contain a decimal point, for example "700" is not correct for a real variable while "700." is. A message will be generated to the screen if a real number value is missing the decimal point.

The card is ignored and treated as a comment if it starts with **C** separated by a space from the rest of the card.

4.2 List of Data Cards in MARS.INP

The listing of Data Cards is organized in functional groups: cards used in the overall control of the program; cards used to describe the materials and material dependent parameters; cards used to describe the Standard Zone geometry; cards which control histograms and tabulation; and cards to control specific physics interactions. Cards are generally listed alphabetically within these functional groups.

4.2.1 IND Switches for Default/non-Default Options

INDX IND(16)

Logical variables which act as on-off switches to control various options, described by the list below. Default: 9*F 10=T 6*F.

`IND(1)=T` The `MARS.OUT` file will contain an extended format, such as tables of the time-dependent residual dose in all materials. See Section 11.1.2 for details.

`IND(1)=F` Standard format `MARS.OUT` file. See Section 11.1.2 for details.

`IND(2)=T` sets "$Z - sandwich$" geometry as a basis for the Standard Geometry zones. See the **`ZSEC`** and **`RSEC`** cards.

`IND(2)=F` sets "$R - sandwich$" geometry as a basis for the Standard Geometry zones. See the **`ZSEC`** and **`RSEC`** cards.

`IND(3)=T` turns on the Extended Geometry option, where a model is described using combinations of boxes, cylinders, cones, etc. See Section 4.4 for a description for using the `GEOM.INP` input deck.

`IND(3)=F` Extended Geometry option turned off.

`IND(4)=T` set this switch when magnetic or electric fields are present in the geometry description. This switch triggers calls to the user subroutine `FIELD`, where field maps or components must be defined.

`IND(4)=F` no magnetic or electric fields present.

`IND(5)=T` turns on MCNP mode, which uses the MCNP4C–ENDF/B-VI code system for neutron transport below 14.5 MeV, with corresponding low energy photon production.

`IND(5)=F` use the MARS default neutron transport, for neutrons in the 0.00215 eV to 14.5 MeV energy range, using the 28-group neutron cross-section library with no low-energy photon treatment.

`IND(6)=T` Switch on the mathematical expectation method to accumulate results for transported particles: probabilities are used to tabulate fluence, star density, particle spectra and the calculations for dose equivalent, rather than using the analog method to directly tabulate particle contributions. This option is useful for "deep penetration" problems, *i.e.* thick shielding; however, it can substantially increase the CPU-time, especially when neutron cutoff energy is below a few MeV. It is recommended to conduct a short test to evaluate an efficiency of this option for a given application. See Section 13.1 for more discussion. Note that results for direct energy deposition and related values are not affected by this flag.

`IND(6)=F` use the default analog method for tabulation of results of transported particles; appropriate for geometries which do not model thick shielding.

`IND(7)=T` switch on forced nuclear interaction ('point target mode'): the incident primary hadron or heavy ion interacts, with a certain probability, with a point-like target placed in the system at coordinates (x_0, y_0, z_0). The probability is specified by the `EFF` variable associated with the **`VARS`** data card. The material index for this target is defined by user in the `BEG1` subroutine. If not, it is determined automatically at the initial point x_0, y_0, z_0 (see **`INIT`** card). `IM=1` is used if this point belongs to vacuum. This switch also forces μ-decay when the primary is a μ (variable `IO` in the **`IPIB`** card $= 7$ or $= 8$), without ν-transport turned on. This switch is typically used for models of muon accelerators.

`IND(7)=F` no point-like target, and no forced μ-decay for μ primaries.

`IND(8)=T` switch on neutrino beam and transport, for both ν primary beam and ν from π and μ decays. This switch is typically used for models of muon accelerators, not for conventional neutrino beams.

`IND(8)=F` no neutrino beam and transport.

`IND(9)=T` initiates the use of special algorithms (as referenced in [37, 98]) to construct extremely accurately the particle trajectories prior to the first two inelastic nuclear interactions. The current default algorithm now provides the same level of accuracy, therefore this option is obsolete.

`IND(9)=F` switch off the above algorithms.

`IND(10)=T` Default. The flag activates muon transport with all possible interaction processes included (see Section 13.2 for details).

`IND(10)=F` turns off muon transport.

`IND(11)=T` adds the $\phi-$ dimension to the Standard (*r*-*z*) Geometry zones, giving an azimuthal structure to the accumulation of the results. The user must specify the number and size of the azimuthal zones via the **`NAZM`** and **`AZIM`** data cards.

`IND(11)=F` no azimuthal divisions in the Standard Geometry zones.

`IND(12)` reserved for the point-kernel option to calculate residual radioactivity with routine MARACT; not implemented in this version.

`IND(13)=T` switch on the use of the MAD-MARS beam line builder MMBLB interfaced with the MAD lattice description, and define histograms, using user subroutines `MARS2BML` and `BML2MARS`.

`IND(13)=F` MMBLB is not activated.

`IND(14)=T` User subroutines `ALIGN`, `SAGIT` and `RFCAVT` are called by the MARS geometry description software modules.

`IND(14)=F` The above subroutines are not called.

`IND(15)=T` Enables a "global parameter set for thick shielding". The set consists of various physics options, energy thresholds, and tabulation options which are collectively appropriate when modeling thick shielding. Use in conjunction with `IND(6)=T`. Setting this flag can provide a *substantial* CPU-time saving for models of thick extended shielding. The particular parameters and their settings are given in Table 6. As with all similar global settings, any of the individual parameters in this set can be re-set, globally or for specific materials, using the appropriate input deck parameter

`IND(15)=F` The above items in the "global parameter set for thick shielding" are set to standard MARS defaults

`IND(16)=T` MCNP style geometry description is used, via entries in the MCNP section of the `MARS.INP` file, and with `IND(5)=T`.

`IND(16)=F` MCNP geometry description is not used.

`IND(17)=T` Enables algorithms for accurate modeling of Displacement-per-Atom (DPA) distributions and histograms. These algorithms are valid only if the threshold energies of charged particles in matter, `EMCHR` and `EMIEL` and corresponding material-dependent values of the `MTCH` and `MTEL` cards, are below $EDPA_{th}$ = 0.005 GeV. That is why the threshold energies for charged particles are converted to $EDPA_{th}$ if they exceed $EDPA_{th}$ and `IND(17)=T`.

`IND(17)=F` The DPA algorithms are not activated, and DPA distributions and histograms are not calculated.

4.2.2 Run Control – `CTRL, RZVL, NEVT, SEED, SMIN, VARS, UCTR`

`CTRL` `IVIS IEVT IVOL IHIS IDTR`

Integer variables which control running modes: the normal Monte Carlo session, GUI visualization mode, "stand-alone" event generator mode, and volume calculation mode. Each of these is independent and they cannot be used concurrently. Details on using the GUI visualization interface are in Section 12, and details on using the event generator in stand-alone mode are in Section 5.2.

Table 6: Values of parameters set by using the `IND(15)` switch.

Process or Parameter	default setting	`IND(15)=T` setting
Charged hadron multiple Coulomb scattering	Always on	On for 1st 3 generations of cascade tree
Charged hadron nuclear elastic scattering at $E > 5$ GeV	Always on	On for 1st 3 generations of cascade tree
Knock-on electron production by hadrons, with modeling of induced electromagnetic showers	Always on	On for 1st 3 generations of cascade tree, continuous dE/dx after
Knock-on electron production by muons with modeling of induced electromagnetic showers	Always on	On for muons with $E/E0 > 0.2$, continuous dE/dx after
Direct e^+e^- pair production by muons with modeling of induced electromagnetic showers	Always on	On for muons with $E > 5$ GeV and $E/E0 > 0.2$, continuous dE/dx after
Global cutoff energies:		
Charged particles	0.2 MeV	the larger of 3 MeV and the user-defined value
Photons	0.2 MeV	the larger of 1 MeV and the user-defined value
Neutrons	0.1 MeV(*)	the larger of 0.1 MeV and the user-defined value
Value for boundary localization variable `STEPEM` (see **`SMIN`** card)	User-defined, typically 0.01cm	larger of 0.3 cm and user-defined
`NEVTYPE` evaporated particle types	17	6
`LAQGSM` mode	Optional	Always off
`KEMINCL` parameter for EMS control	10	0

(*) Used to be 10^{-12} GeV in previous versions

`IVIS` When `IVIS=0`, MARS will run in it's normal simulation mode. If `IVIS=1` then the MARS GUI interface is activated. No events will be generated, and the GUI `X`-window will pop up when the executable is run. The size of the GUI window is defined interactively in an dialog menu. Default: 0.

`IEVT` If `IEVT`$\geq$`1`, then regardless of the value of `IVIS`, MARS will run in it's stand-alone event generator mode. The primary particle energy and type are specified using the **`ENRG`** and **`IPIB`** cards, and these interact on nucleus type `IEVT=IM`, where `IM` is the index of the selected material as listed in the **`MATR`** card. The generator is run for `NSTOP` events as given in the **`NEVT`** card. Default: 0.

`IVOL` If `IVOL=1`, then regardless of the value of `IVIS` or of `IEVT`, MARS will run in volume calculation mode. It will generate coordinates within the values set by the **`RZVL`** card, and probe the boundaries of the zones defined within that space, tallying the volumes of those zones. The results go into an output file VOLMC.NON in the format ready for inclusion into the user routine `VFAN`. Statistical errors for the calculated volumes are also included to see if a number of events `NVTRIAL` is sufficient to get a good estimate of the zone volumes. This mode is useful to obtain the volumes of oddly shaped objects. As discussed in Sections 4.2.5 and 5.5, MARS must know the volumes of all zones to be able to calculate results such as particle flux. Default: 0.

`IHIS` When `IHIS=0`, a detailed 16-color palette is used for histogram presentation

in the GUI, optimized for color view of results calculated spaning up to 16 decades. If `IHIS=1`, then a 8-color palette is used for histogram presentation, convenient for figures to be included in a black and white paper. Default: 0.

`IDTR` If `IDTR=1`, then the DETRA code [2, 3] is called to solve decay and transmutation equations for all nuclides generated in the materials specified by the **`NCLD`** and **`IMNC`** cards. After such a MARS run is done, the corresponding output files are processed by running the MARS executable again with `IDTR=2`. Default: 0.

`RZVL` `R1 R2 Z1 Z2 NVTRIAL`

Variables which define a cylinder containing zones (non-standard, MCNP, extended or overlaped), whose volumes a user wants to have calculated. By default this cylinder is the mother Standard zone, but in general one should define a smaller cylinder around specific regions to get better statistics. If the overall geometry is large and contains many smaller non-standard zone objects at large distances from each other, it is recommended to run several short jobs defining the cylinders surrounding different zone groups. Statistical RMS errors in the volume calculation are printed in the `VOLMC.NON` file and can be reduced by increasing `NVTRIAL`. Typically it takes a few tens of seconds on a typical workstation to get the above errors $\leq 1\%$ with `NVTRIAL` of the order of a few millions.

`R1, R2, Z1, Z2` Real variables for minimal and maximal radii and z-coordinates of a cylinder which contains the non-standard regions whose volumes need to be calculated. Default: 0., `RMAX, ZMIN, ZMAX`, the size of the mother Standard zone .

`NVTRIAL` Integer or real variable equal to the number of events to run in this volume determination session. Default: 10^7.

`NEVT` `NSTOP NTIME NHIPR NITRMX IPRINM`

Integer variables which control the number of events to run and print, and control the printout of a given history and of nuclear cross-sections.

`NSTOP` the number of events to generate (equals incident particles thrown). Can be integer or real in the interval $1 \leq NSTOP \leq 2 \times 10^9$. Default: 200.

`NTIME` the number of times user routine `DUMP` will be called. Each call writes intermediate results to the `DUMP` file. Recommended: 10. Default: 0. Useful to keep track of the session in a standard Monte Carlo run as well as in the event generator ($IEVT \geq 1$) or volume Monte Carlo ($IVOL = 1$)sessions. If `NTIME=-1` then a file `TRACKFIRST` is generated which contains x, y, z, dcx, dcy coordinates and direction cosines of the first track, useful for beam-line studies.

`NHIPR` an event number which will be written in great detail to file `TRACK.PLOT` (if `NWEGH=18` on the **`TAPE`** card) and to file `fort.30`. If `NHIPR` is negative then the event number of the current event being generated is printed to the screen, starting with event number `N=abs(NHIPR)`. Default: 0.

`NITRMX` the maximum event number for which the `TRACK.PLOT` file is generated. Default: 1000.

`IPRINM` If `IPRINM=1` then detailed tables of hadron and photon nuclear cross-sections, as well as electromagnetic energy loss dE/dx, are printed to `MARS.OUT`, for each material in the model and as a function of energy. Default: 0.

SEED IJKLIN NTOTIN NTOT2N

Integer variables to initialize the random number generator, and to track how many random numbers are used.

- IJKLIN — The initial 64-bit seed. The integer entered is assumed to be in octal, therefore no digit should be greater than 7. Default:54217137; shift example:64217136.
- NTOTIN — Default: 0
- NTOT2N — Default: 0

SMIN STEPEM STEPH

Real variables specifying global boundary localization precision and pilot step lengths. The program tracks particles in line segment steps (see discussion in Section 3.3. The pilot step, STEPH is used initially if it is smaller than that defined from physics and magnetic field criteria; if within that step, the zone number changes, then the program will iterate back and forth, in smaller and smaller steps, to locate the boundary between the zones (the boundary is located exactly, without iterations, in the Extended Geometry mode). The minimum size of these iterative steps is given by STEPEM. The larger the step lengths, the faster the code will run, but accuracy is affected if the step lengths are so large that smaller size zones become "invisible". One can, however, specify fairly large (or small) step lengths here, and then apply smaller (or larger) step lengths to particular materials via variables assigned to the **MTSM** and **MTSH** cards. See the example in Section 4.3.3.

- STEPEM — The global parameter, in cm, for the accuracy of boundary localization in the particle transport algorithm. It is strongly recommended to define STEPEM to be no larger than $0.1 \times t_{min}$, where t_{min} is the smallest dimension of the smallest zone in the model. Default: 0.01
- STEPH — The global pilot step length, in cm. While making the value large has a modest effect on the time per event $\simeq \log(\mathtt{STEPH}/\mathtt{STEPEM})$. The recommendation is to set STEPH$= \min(\lambda, l)$, where λ is the mean inelastic length for hadrons and l is the length of the zones of interest, in the direction which most particles pass through those zones. Default: 10 cm.

VARS EFF DLEXP TEMPO AINT

Real variables which control miscellaneous aspects of the model.

- EFF — The point-like target efficiency, which is active only when IND(7)=T, $0 \leq$ Abs(EFF) ≤ 1. This provides a forced nuclear inelastic interaction of a primary beam, if EFF > 0. This provides a forced nuclear sampled inelastic or elastic interaction of a primary beam, if EFF < 0. Default: 0.
- DLEXP — An exponential transform factor, which allows an increase (DLEXP$\geq$1) or decrease (DLEXP$\leq$1) in the effective hadronic mean free path: $\lambda^{'} = \lambda \times DLEXP$. This feature is useful where the "deep penetration" problem applies, and for compact restricted models. Recommendation: 0.3$\leq$DLEXP$\leq$3. Default: 1.
- TEMPO — The initial temperature T_0, in $Kelvin$, of the model (all zones). The allowed range is $1.8 \leq T_0 \leq 1800$. Default: 300.
- AINT — The number of particles, N_0, in a given pulse of beam, which is used for two types of normalization. The format is 4=1.0e12. The units for the value are different for temperature rise results than for other results, so use caution when examining output results. For instantaneous temperature rise

due to beam, then N_0 is assumed to be the number of particles in a single instantaneous pulse. For all other results - contact residual dose, power density, radiation damage (DPA per year) - N_0 is assumed to be an average beam intensity in particles per second. Default: 10^{12}.

UCTR `ICTR(10)`

Integer variables for arbitrary control in user routines.

`ICTR` The array containing ten integer numbers (negative, positive or zero) which can be used in user routines for arbitrary control. Most useful in the following routines: `BEG1, FIELD, LEAK, VFAN, TAGPR, WRTSUR`. A COMMON block /BLICTR/ICTR(10) is inserted in these routines. Default: $10 * 0$.

4.2.3 Beam & Energy Thresholds – **IPIB, BEAM, INIT, ENRG**

IPIB `I0 IBEAM IZPRJ IAPRJ`

Integer variables to specify the incident particle type, the incident beam distribution and the charge and mass for incident heavy ions. If the `IBEAM` value is non-zero, then the parameters of the profile are defined using the **BEAM** card. Use these variables for simple, uniform beam distributions. More complex primary beam distributions must be modeled via the user subroutine `BEG1`.

`I0` The incident particle type; see Table 2 for the index values assigned to various particles. Default: 1, proton.

`IBEAM` The type of incident beam distribution, given by one of the four following values. Default: 0.
-1 – laterally and longitudinally infinitesimal beam.
0 – "a diffused track": the beam is distributed uniformly in a box $\pm 10^{-5}$ cm around variables `XINI, YINI, ZINI` in the **INIT** card, default.
1 – the beam is distributed uniformly in a rectangular area of half-size as specified by variables `SIXX SIYY` in the **BEAM** card.
2 – the beam has a simple Gaussian distribution with σ_x and σ_y specified by `SIXX SIYY` in the **BEAM** card.
3 – the beam has a Gaussian spatial distribution, as for `IBEAM=2`, and also has a Gaussian angular spread with $\sigma(\theta_x)$ and $\sigma(\theta_y)$, in radians, specified by variables `SITX SITY` in the **BEAM** card. This angular spread will always be diverging, however. Focused beam trajectories must be modeled via the user subroutine `BEG1`.

`IZPRJ` Integer electric charge of the projectile if it is heavier than ^4He. Default: 1.

`IAPRJ` Integer mass (baryon) number of the projectile if it is heavier than ^4He. Default: 1.

BEAM `SIXX SIYY SITX SITY DLBNCH`

Real variables used to specify the x, y, z beam shape around initial values from the **INIT** card. The first four variables are active only for non-zero values of `IBEAM` in the **IPIB** card. The default for each is 0.0.

`SIXX` For `IBEAM=1`, the half-size in x of a uniform retangular beam; for `IBEAM=2,3`, the σ_x of a Gaussian beam. Both in cm.

`SIYY` For `IBEAM=1`, the half-size in y of a uniform retangular beam; for `IBEAM=2,3`, the σ_y of a Gaussian beam. Both in cm.

`SITX` The gaussian angular spread $\sigma(\theta_x)$, in radians; used only when `IBEAM=3`.

`SITY` The gaussian angular spread $\sigma(\theta_y)$, in radians; used only when `IBEAM=3`.

`DLBNCH` The RMS bunch length σ_z of a Gaussian beam in cm (30 cm = 1 ns). If positive, it specifies a Gaussian longitudinal distribution around $ZINI$: "sample z-position and clock start around $t_0 = 0$". Be sure that $ZMIN < ZINI - 3 \times DLBNCH$. If $DLBNCH < 0$, then the code generates the initial particle time distribution from a Gaussian distribution with $< t >= 0$ and σ derived from $abs(DLBNCH)$: "sample clock start only". The z-coordinate of the initial particle is unchanged: $z = ZINI$. Corresponding time-of-flight distributions in all detectors are naturally affected in both cases. Note that all the MARS versions before July 27, 2005, had just a positive $DLBNCH$ as a beam RMS bunch length (cm) to generate a Gaussian time distribution with $< t >= 0$ with the unchanged initial z-coordinate. The current version prints on screen parameters of several first projectiles, so, one can always control the initial conditions.

INIT `XINI YINI ZINI DXIN DYIN DZIN WINIT`

Real variables used to specify the initial starting point and direction cosines of the incident beam. In particular, if the starting point of the users model is not zero, as given by variable `ZLEFT` in the **ZMIN** card, then the starting point of the primary should be adjusted appropriately. Use these variables for simple beam descriptions. More complex primary beam trajectories must be modeled via the user subroutine `BEG1`.

`XINI,YINI,ZINI` These are initial x_0, y_0, z_0 coordinates of the beam spot center. Default: $0.0, 0.0, 0.0$, in cm.

`DXIN,DYIN,DZIN` These are initial direction cosines of the beam centroid. Default: $0.0, 0.0, 1.0$

`WINIT` The initial weight of each incident particle, which the results are normalized to. Beware of interference between this value and `AINT` in the **VARS** card, which also affects normalization. An often-used example: put `WINIT`=d/λ_{in} when the point-like target efficiency `EFF`=1 on the `VARS` card; here d is the target thickness, and λ_{in} is an nuclear inelastic mean free path of the beam hadrons in the thin target material. Beware of interference between this value and `AINT` in the **VARS** card, which also affects normalization. Default: 1.0.

ENRG `E0 EM EPSTAM EMCHR EMNEU EMIGA EMIEL EMNU EMEVAP`

Real variables which specify the incident particle energy, and the thresholds in matter for subsequent generated particles. In most cases, generated particles will not be tracked once their energy is below the threshold, and their contribution to flux and energy distributions will no longer be counted; exceptions to this are noted below. These thresholds are global - applied to all zones; material/zone specific thresholds can be specified using variables assigned to the **MTCH**, **MTNE**, **MTGA**, and **MTEL** data cards. The minimal threshold energy is 10^{-12} GeV for neutrons, 10^{-6} GeV for charged hadrons, muons and heavy ions, and 10^{-4} GeV (currently) for electrons and photons. **Note that for accurate calculation of energy deposition the user must keep all the threshold energies below 0.0145 GeV! They can be arbitrary high if one studies fluxes, spectra, yields, star densities etc., but the code issues a warning on screen and - in most cases - disregards particles falling below a threshold energy if it is greater than 0.0145 GeV**.

`E0` The incident particle kinetic energy (kinetic energy per nucleon for incident heavy ions). It must be $\geq 10^{-12}$ GeV for neutrons, 10^{-6} GeV for charged

hadrons, muons and heavy ions, and 10^{-4} GeV (currently) for electrons and photons. Default: 100.0 GeV.

`EM` The hadron threshold energy. This parameter induces different behaviours for settings above or below the default. Default: 0.0145 GeV. For values less than the default, the threshold affects only flux and spectra tabulation, and not energy deposition. For example, π^0's decay and deposit their energy as γ's at their point of generation, and this process will be modeled even for π^0's below the `EM` setting. For values greater than the default, however, the threshold value becomes **global**, and is applied to all particles and not just hadrons. In this case, energy deposition will be affected, and all flux results biased, but the CPU-time will have decreased significantly. This feature is useful in models which study shielding configurations, where star density is the main result of interest. For such models, `EM` should be set to 0.03 GeV, which is the standard star production threshold.

`EPSTAM` The star production threshold kinetic energy. Default: 0.03 GeV, which is the standard definition of a star.

`EMCHR` The threshold energy applied collectively to muons, heavy ions and charged hadrons. The smallest possible value in the current version is 10^{-6} GeV. Default: 0.001 GeV.

`EMNEU` The threshold energy for neutrons. The smallest possible value is 10^{-12} GeV. Default: 10^{-4} GeV

`EMIGA` The threshold energy for γ. The smallest possible value is 10^{-4} GeV, in all materials except for the standard vacuum material (`'VAC'`), where the threshold can be as low as 10^{-6} GeV. Default: 2×10^{-4} GeV.

`EMIEL` The threshold energy for $e^{\pm}$. The smallest possible value is 10^{-4} GeV, in all materials except for the standard vacuum material (`'VAC'`), where the threshold can be as low as 10^{-6} GeV. Default: 2×10^{-4} GeV.

`EMNU` neutrino energy threshold. The smallest possible value is 0.01 GeV. Default: 0.01 GeV.

`EMEVAP` The minimal energy for pre-equilibrium and evaporation modeling, which is turned off if `EMEVAP`$\geq$0.2 GeV. Affects nucleons only. Raising this value can substantially reduce CPU-time in models with high-Z materials. For example, if one is mainly interested in π or μ production this value can be safely raised to. Default: 0 GeV.

4.2.4 Materials and Material-Dependent Settings – `NMAT, MATR, MTDN, MTCH, MTNE, MTGA, MTEL, MTSM, MTSH, MTQG, MTEG, BIDC, BIPR, BIBH, BIGV, BIAN, BIGA, BIEA, BIAP`

`NMAT` `NREMA`

Integer variable specifying the number of materials in the model. This value defines to the program the number of active array elements in the variables assigned to the **`MATR`**, **`MTDN`**, **`MTCH`**, **`MTNE`**, **`MTEM`**, **`MTSM`**, **`MTSH`**, **`MTQG`**, **`MTEG`**, **`BIDC`**, **`BIPR`**, **`BIBH`**, **`BIGV`**, **`BIAN`**, **`BIGA`**, **`BIEA`**, **`BIAP`** data cards. The special material, $blackhole$ is not included in the total number. Material $blackhole$ is tagged with a negative `IM` material index value; use of this material in conjunction with user subroutine `LEAK` is described in Section 5.12. The global vacuum, tagged by material index `IM`$= 0$, is also not included in the total number of materials. The global vacuum has global

step sizes and global energy thresholds applied. If the user wishes to control these parameters in certain vacuum regions, then the built-in material VAC should be declared in the **MATR** card list, and then the material-dependent cards can be utilized.

NREMA The total number of different materials used in the model. $1 \leq$NREMA≤ 100. Default: 1.

MATR AMA(100)

Character variables which specify the names of the NREMA materials in the model. Each of these materials has an index, IM$= 1 :$ NREMA, following the order in which they are listed on this card. The material-dependent density, energy thresholds, and step lengths in the **MTDN**, **MTCH**, **MTNE**, **MTEM**, **MTSM**, **MTSH** data cards use the same IM index. **Reminder: for accurate calculation of energy deposition the user must keep all the threshold energies below 0.0145 GeV!**

Materials can be single elements or complex compounds. Tables 3 and 4 list all the elements and built-in compounds used by MARS, with their identifying abbreviations. These abbreviations are the character strings to use in the **MATR** list.

The user can define custom compound mixtures with names MIX1, MIX2, MIX3, etc., up to MIX0 for the first 10, and continue using the names MIXA, MIXB, MIXC ... for subsequent mixtures. Each of these is assigned an IM index, as above, according to their order in the list of materials. The user defines each mixture, identified by the IM index, in user subroutine MIXTUR (see Section 5.3).

A particular material can be listed more than once, for cases where MARS variables applied to that material need to have different values in different regions of the geometry. For example, a model might have both steel shielding and thin steel vacuum pipe, both specified as material FE. For the steel shielding, a large STEPEM and STEPH could be specified, using the global step control parameters assigned to the **SMIN** card. The thin steel pipe could have a smaller step parameters applied, by using the **MTSM** and **MTSH** cards. The material FE would be listed twice so that two IM indexes are assigned. The shielding zone and the pipe zone would have a different material IM index assigned, so that the appropriate step length properties get applied by the code. See the example in Section 4.3.3.

AMA The material name, using the abbreviation exactly as shown in Tables 3 and 4: 'XXXX', including the single quotes, where XXXX is the 2 to 4 character abbreviation for the material. Default: 'FE'.

MTDN ROW(100)

Real variables giving the density, in g/cm^3, of the materials listed in the **MATR** card. All of the elements listed in Table 3 have their standard density value built in. All of the compounds listed in Table 4 have typical density values built in (usually from PDG). If the user wants to apply a different density value, for example if his graphite target has a density different from standard, then this card is used. User-defined mixtures are required to have their density specified by using this card. The default material is FE, and it's standard density is the default value, which means that for the default case, this card does not need to be listed in the input deck. If the density specifed $\geq 10^{20}\ g/cm^3$ then this material is treated as a *blackhole*. If the density specifed $\leq 10^{-20}\ g/cm^3$ then this material is treated as a *vacuum*. In both case a corresponding message will be written to the screen.

ROW The values for the material densities, indexed by IM, the order the materials are listed in the **MATR** card.

MTCH RLCTCH(100)

Real variables giving the charged hadron, heavy ion and muon energy threshold applied only to specific materials. They must be less than 0.0145 GeV if one wishes to calculate energy deposition correctly. Any material which does not have a corresponding entry here will have the global threshold applied as the default, with the value given by the `EMCHR` variable in the **`ENRG`** data card.

`RLCTCH` The values for the energy thresholds, indexed by `IM`, the order the materials are listed in the **`MATR`** card.

`MTNE` `RLCTNE(100)`

Real variables giving the neutron energy threshold applied only to specific materials. They must be less than 0.0145 GeV if one wishes to calculate energy deposition correctly. Any material which does not have a corresponding entry here will have the global threshold applied as the default, with the value given by the `EMNEU` variable in the **`ENRG`** data card.

`RLCTNE` The values for the energy thresholds, indexed by `IM`, the order the materials are listed in the **`MATR`** card.

`MTGA` `RLCTGA(100)`

Real variables giving the electromagnetic energy threshold, for γ, applied only to specific materials. They must be less than 0.0145 GeV if one wishes to calculate energy deposition correctly. Any material which does not have a corresponding entry here will have the global threshold applied as the default, with the value given by the `EMIGA` variable in the **`ENRG`** data card.

`RLCTEM` The values for the energy thresholds, indexed by `IM`, the order the materials are listed in the **`MATR`** card.

`MTEL` `RLCTEL(100)`

Real variables giving the electromagnetic energy threshold, for $e^{\pm}$, applied only to specific materials. They must be less than 0.0145 GeV if one wishes to calculate energy deposition correctly. Any material which does not have a corresponding entry here will have the global threshold applied as the default, with the value given by the `EMIEL` variable in the **`ENRG`** data card.

`RLCTEM` The values for the energy thresholds, indexed by `IM`, the order the materials are listed in the **`MATR`** card.

`MTSM` `RLSTEM(100)`

Real variables giving the step length for boundry localization, applied only to specific materials. Any material which does not have a corresponding entry here will have the global step length applied as the default, with the value given by the `STEPEM` variable in the **`SMIN`** data card. The recommendation for the value of this parameter is the same as described for the `STEPEM` variable.

`RLSTEM` The values for the step lengths, indexed by `IM`, the order the materials are listed in the **`MATR`** card.

`MTSH` `RLSTEH(100)`

Real variables giving the pilot step length, applied only to specific materials. Any material which does not have a corresponding entry here will have the global pilot step length applied as the default, with the value given by the `STEPH` variable in the **`SMIN`** data card. The recommendation for the value of this parameter is the same as described for the `STEPH` variable.

`RLSTEM` The values for the step lengths, indexed by `IM`, the order the materials are listed in the **`MATR`** card.

MTQG ILAQM(100)

Integer variables that switch between the inclusive and exclusive event generators at nuclear inelastic interactions above the energy $ECEM$ (see the **ICEM** data card), applied only to specific materials. Any material which does not have a corresponding entry here will have the global switch IQGSM applied as the default (see the **ICEM** data card). For ILAQM(i)=0, the default inclusive model is used above the energy $ECEM$. For ILAQM(i)=1, exclusive modeling with the LAQGSM is done at $E > ECEMLQ = 1 + A/65$, or/and $A < 3$, and at all energies for $\overline{p}, K^{\pm}, d, t, ^3He, ^4He, \overline{n}$, hyperons and heavy ions; the CEM model is used otherwise. For ILAQM(i)=2, exclusive modeling with the LAQGSM code is always done for everything at all energies.

ILAQM — The values for the model switch, indexed by IM, the order the materials are listed in the **MATR** card.

MTEG KEMIN(100)

Integer variables that control the inclusive and exclusive modes in electromagnetic shower verticies as well as in $n - \gamma$ and other neutron interaction verticies (while in the MCNP mode) (see the **PHOT** data card), applied only to specific materials. Any material which does not have a corresponding entry here will have the global switch KEMINCL applied as the default (see the **PHOT** data card). For KEMIN(i)=0, the inclusive mode is used for all electromagnetic shower verticies. For KEMIN(i)=-1, exclusive modeling is done for all electromagnetic shower verticies; five EMS bias keys (see the **BIAS** data card) are automatically converted to -1, i.e. to the exclusive mode. For KEMIN(i)= $N \geq 1$, exclusive modeling is used for the first N generations and inclisive one for remaining higher generation levels of the electromagnetic shower.

KEMIN — The values for the model switch, indexed by IM, the order the materials are listed in the **MATR** card.

BIDC RLDEC(100)

Real variables for unstable particle decays, applied only to specific materials. Any material which does not have a corresponding entry here will have the global parameter applied as the default, with the value given by the PPIKDEC variable in the **BIAS** data card. **Currently, the global parameter PPIKDEC overwrites any values given on this card and is applied to all materials.**

RLDEC — The values for the unstable particle decay control, indexed by IM, the order the materials are listed in the **MATR** card.

BIPR RLPRM(100)

Real variables specifying the parameters of the inclusive (forced) prompt muon production in the inelastic nuclear interaction vertex, applied only to specific materials. Any material which does not have a corresponding entry here will have the global parameter applied as the default, with the value given by the PMUPRMT variable in the **BIAS** data card. Values of the RLPRM array specify prompt muon production modelling. It is forced with the appropriate statistical weight at RLPRM(i) = 1, forced with the Russian roulette at 0.001 $\leq$ RLPRM(i) $<$ 1, or suppressed at RLPRM(i) = 0. Default: 0.05.

RLPRM — The values for the prompt muon production control, indexed by IM, the order the materials are listed in the **MATR** card.

BIBH RLBEH(100)

Real variables specifying the parameters of the inclusive (forced) Bethe-Heitler muon production, applied only to specific materials. Any material which does not have a corresponding entry here will have the global parameter applied as the default, with the value given by the `PMUBEHE` variable in the **BIAS** data card. Values of the `RLBEH` array specify Bethe-Heitler muon production modelling. It is forced with the appropriate statistical weight at `RLBEH(i)` = 1, forced with the Russian roulette at 0.001 $\leq$ `RLBEH(i)` $<$ 1, modelled exclusively at `RLBEH(i)` = -1, or suppressed at `RLBEH(i)` = 0. Alternatively, the Bethe-Heitler muon production cross-section can be increased 1 to 500 times with -500 $\leq$ `RLBEH(i)` $<$ -1; an appropriate correction to a statistical weight assures correct results. It is automatically set to -1 for `KEMIN(i)` = -1. Default: 0.03.

`RLBEH` The values for the Bethe-Heitler muon production control, indexed by `IM`, the order the materials are listed in the **MATR** card.

BIGV `RLGVM(100)`

Real variables specifying the parameters of the inclusive (forced) muon production via vector mesons generated in photo-nuclear reactions at $E > 1.5$ GeV when `IND(10)=T`, applied only to specific materials. Any material which does not have a corresponding entry here will have the global parameter applied as the default, with the value given by the `PMUGVM` variable in the **BIAS** data card. Values of the `RLGVM` array specify muon production via vector mesons modelling. It is forced with the appropriate statistical weight at `RLGVM(i)` = 1, forced with the Russian roulette at 0.001 $\leq$ `RLGVM(i)` $<$ 1, modelled exclusively at `RLGVM(i)` = -1, or suppressed at `RLGVM(i)` = 0. It is automatically set to -1 for `KEMIN(i)` = -1. Default: 0.02 when the primary beam particle is photon, electron or positron (`I0` = 9, 10 or 11 on the `IPIB` card), otherwise it is 0.005 at $E > 20$ GeV, or 0 at $E \leq 20$ GeV.

`RLGVM` The values for the control of muon production via vector mesons, indexed by `IM`, the order the materials are listed in the **MATR** card.

BIAN `RLANN(100)`

Real variables specifying the parameters of the inclusive (forced) $e^+e^- \rightarrow \mu^+\mu^-$ annihillation events, applied only to specific materials. Any material which does not have a corresponding entry here will have the global parameter applied as the default, with the value given by the `PMUANN` variable in the **BIAS** data card. Values of the `RLANN` array specify $e^+e^- \rightarrow \mu^+\mu^-$ event modelling. It is forced with the appropriate statistical weight at `RLANN(i)` = 1, forced with the Russian roulette at 0.001 $\leq$ `RLANN(i)` $<$ 1, modelled exclusively at `RLANN(i)` = -1, or suppressed at `RLANN(i)` = 0. It is automatically set to -1 for `KEMIN(i)` = -1. Default: 0.03.

`RLANN` The values for the control of $e^+e^- \rightarrow \mu^+\mu^-$ annihillation events, indexed by `IM`, the order the materials are listed in the **MATR** card.

BIGA `RLPHN(100)`

Real variables specifying the parameters of the inclusive (forced) photo-nuclear events, applied only to specific materials. Any material which does not have a corresponding entry here will have the global parameter applied as the default, with the value given by the `PPHNUC` variable in the **BIAS** data card. Values of the `RLPHN` array specify photo-nuclear event modelling. It is forced with the appropriate statistical weight at `RLPHN(i)` = 1, forced with the Russian roulette at 0.001 $\leq$ `RLPHN(i)` $<$ 1, modelled exclusively at `RLPHN(i)` = -1, or suppressed at `RLPHN(i)` = 0. It is automatically set to -1 for `KEMIN(i)` = -1. Default: 0.02.

`RLPHN` The values for the photo-nuclear event control, indexed by `IM`, the order the materials are listed in the **MATR** card.

BIEA RLELN(100)

Real variables specifying the parameters of the inclusive (forced) electro-nuclear events, applied only to specific materials. Any material which does not have a corresponding entry here will have the global parameter applied as the default, with the value given by the PELNUC variable in the **BIAS** data card. Values of the RLELN array specify electro-nuclear event modelling. It is forced with the appropriate statistical weight at RLELN(i) = 1, forced with the Russian roulette at 0.001 $\leq$ RLELN(i) $<$ 1, modelled exclusively at RLELN(i) = -1, or suppressed at RLELN(i) = 0. It is automatically set to -1 for KEMIN(i) = -1. Default: 0.05 when the primary beam particle is electron or positron (I0 = 10 or 11 on the IPIB card), otherwise it is 0.003 at $E < 300$ GeV, or 0 at $E \geq 300$ GeV.

RLELN The values for the control of electro-nuclear events, indexed by IM, the order the materials are listed in the **MATR** card.

BIAP RLPBR(100)

Real variables specifying the parameters of the inclusive (forced) antiproton production, applied only to specific materials. Any material which does not have a corresponding entry here will have the global parameter applied as the default, with the value given by the PPBAR variable in the **BIAS** data card. Values of the RLPBR array specify antiproton production modelling. It is forced with the appropriate statistical weight at RLPBR(i) = 1, forced with the Russian roulette at 0.001 $\leq$ RLPBR(i) $<$ 1, or suppressed at RLPBR(i) = 0. Default: 0.05.

RLPBR The values for the antiproton production control, indexed by IM, the order the materials are listed in the **MATR** card.

4.2.5 Standard Geometry Zone Definition – ZMIN, NLNG, ZSEC, NLTR, RSEC, NAZM, AZIM, IRFL

ZMIN ZLEFT

Real variable – normally used if negative z-coordinates are present – giving the minimum (left-most) value for the z-coordinate. The value can be negative if necessary. The starting location for the primaries, ZINI in the **INIT** data card, may be moved to match this coordinate, or placed at a larger coordinate; this depends on the details of the user's model.

ZLEFT The left-most z-coordinate. Default: 0.

NLNG LZ NLZ

Integer variables which define the number of major longitudinal divisions of the MARS Standard geometry zones. See Section 4.3.1 for further discussion.

LZ The number of major longitudinal sections described in the accompanying **ZSEC** data card. The allowed range is 1$\leq$LZ$\leq$1250. Default: 1

NLZ The number of times the major sections described in the **ZSEC** card are repeated. Active only for NLZ$\geq$2. Default: 1

ZSEC ZSE(1250) IZN(1250) IZI(1250)

Real and integer variables which define the locations of the major longitudinal divisions of the MARS Standard geometry zones and the number of sub-sections within each. The materials in those zones are also specified by this card, when IND(2)=T, which is for a z-*sandwich* type geometry; see Section 4.3.1 for further discussion. Recall from the discussion in Section 4.1 that the three variables of dimension 1250 are mapped onto a single array of dimension 3750.

The boundary divisions are contiguous (no gaps) and must be listed in ascending order. The maximum z coordinate in the model, `ZMAX`, is the right-hand boundary of the last declared division, or the last `ZSE` value listed.

- `ZSE(i)` The z-coordinate (real number) of the right-hand boundary of the `i`th longitudinal section. `ZLEFT` in the **`ZMIN`** data card gives the left-hand coordinate of the 1st section. These values occupy elements 1:1250 of the mapped array. Default: 100., 1249*0.
- `IZN(j)` The integer number of minor subsections within a given major section. Elements `j`=1251:2500 of the mapped array correspond to the `i`=1:1250 major sections. Default: 1250*1
- `IZI(k)` The material index value `IM` of each major section. The minor subsections must be of the same material as their major section. Elements `k`=2501:3750 of the mapped array correspond to the `i`=1:1250 major sections. This variable is active only when `IND(2)=T`. Default: 1250*0

`NLTR` `LR`

Integer variable which defines the number of major radial divisions of the MARS Standard geometry zones. See Section 4.3.2 for further discussion.

- `LR` The number of major radial sections described in the accompanying **`RSEC`** data card.. The allowed range is 1≤`LR`≤50. Default: 1

`RSEC` `RSE(50) IRN(50) IRI(50)`

Real and integer variables which define the locations of the major radial divisions of the MARS Standard geometry zones and the number of sub-sections within each. The materials in those zones are also specified by this card, when `IND(2)=F`, which is for a r-$sandwich$ type geometry; see Section 4.3.2 for further discussion. Recall from the discussion in Section 4.1 that the three variables of dimension 50 are mapped onto a single array of dimension 150.

The boundary divisions are contiguous (no gaps) and must be listed in ascending order. The maximum r coordinate in the model, `RMAX`, is the outer boundary of the last declared division, or the last `RSE` value listed.

- `RSE(i)` The r-coordinate (real number) of the outer boundary of the `i`th radial section. The inner coordinate of the 1st section is assumed to be 0. These values occupy elements 1:50 of the mapped array. Default: 5.0, 49*0.
- `IRN(j)` The integer number of minor subsections within a given major section. Elements `j`=51:100 of the mapped array correspond to the `i`=1:50 major sections. Default: 50*1
- `IRI(k)` The material index value `IM` of each major radial section. The minor subsections must be of the same material as their major section. Elements `k`=101:150 of the mapped array correspond to the `i`=1:50 major sections. This variable is active only when `IND(2)=F`. Default: 50*0

`NAZM` `NF`

Integer variable which defines the number of azimuthal divisions of the MARS Standard geometry zones. This card is valid only when `IND(11)=T`.

- `NF` The number of azimuthal bins, 1≤`NF`≤60. Default: 1

`AZIM` `FIB(60)`

Real variables which give the angular size, in degrees, of each azimuthal division. This card is valid only when `IND(11)=T` and `NF`$\geq$2.

`FIB` The angular sizes, 0$\leq$`FIB(i)`$\leq$360. Default: 60*0.0

`IRFL` `IRFL(3)`

Integer variables which control the reflection of the geometry described for the positive x, y, z-directions to the negative ones. If `IRFL(1)`=1, the geometry description at $x \geq 0$ is reflected to the negative x. Similar reflections are done independently for y and z directions at `IRFL(2)`=1 and `IRFL(3)`=1, respectively. In the current version, only z-reflection for the MCNP geometry is allowed at `IRFL(3)`=1. Histograms are not affected by reflections because these are independent of geometry.

`IRFL` The reflection flags. Default: 3*0.

4.2.6 Importance Sampling – `IMPT, IMPZ, IMPR`

`IMPT` `EIMPT NIMPTZ NIMPTR`

Real and integer variables which define the threshold energy for hadron importance sampling and the numbers of surfaces for that. See Section 3.9. The surfaces and importances are defined in the accompanying **`IMPZ`** and **`IMPR`** data cards. Put these surfaces at large distances from the shower core to examine a deep penetration problem, never in vacuum or at a boundary with vacuum.

`EIMPT` The cutoff energy for hadron weight splitting and Russian roulette. Default: 10^{-4} GeV

`NIMPTZ` The number of the z-planes for hadron weight splitting and Russian roulette, 1$\leq$`NIMPTZ`$\leq$`10`. Default: 0

`NIMPTR` The number of the cylindrical surfaces for hadron weight splitting and Russian roulette, 1$\leq$`NIMPTR`$\leq$`10`. Default: 0

`IMPZ` `ZZIMPT(i),WZIMPT(i),i=1,NIMPTZ`

Real variables which define the importance sampling in the z-direction.

`ZZIMPT` z-coordinates which define the planes for hadron weight splitting and Russian roulette. Place them in the exponential attenuation region well beyond the shower maximum. $z_1 < z_2 < ... < z_{10}$. Recommended $\Delta z \approx \lambda_{in}$, where λ_{in} is a material-dependent inelastic nuclear mean free path. In typical thick shielding case use it for neutrons of energy of several hundreds MeV. Default: 10*0.0

`WZIMPT` Importance factors at the above z-planes which define the factors for hadron weight splitting and Russian roulette. If $\Delta z \approx \lambda_{in}$, then recommended `WZIMPT(i)` $\approx e \approx$2.7. Default: 2.0

`IMPR` `RRIMPT(i),WRIMPT(i),i=1,NIMPTR`

Real variables which define the importance sampling in the radial direction.

`RRIMPT` Radii which define the cylindrical surfaces for hadron weight splitting and Russian roulette. Place them in the exponential attenuation region at radii exceeding several λ_{in}. $r_1 < r_2 < ... < r_{10}$. Recommended $\Delta r \approx \lambda_{in}$. Default: 10*0.0

`WRIMPT` Importance factors at the above radii which define the factors for hadron weight splitting and Russian roulette. If $\Delta r \approx \lambda_{in}$, then recommended `WRIMPT(i)` $\approx e \approx$2.7. Default: 2.0

4.2.7 Histograms and Tabulation – NOBL, RZOB, NSUR, RZTS, TOFF, NHBK, HBKE, NCLD, IMNC, NDET, FLOC, TAPE, HTIR, RTIR, CFTD

NOBL `NOB MHIRFL NHSPE NR1 NZ1 NR2 NZ2 NR3 NZ3`

Integer variables which give the number of "special" rz-regions in which "RZ-volume type" histograms will be accumulated, the flag which controls creation of z-histograms, and the type of scale for spectral histograms. See Section 10.1 for the list of the histograms and their IDs. The size of the volume of each region is given in the accompanying **RZOB** data card.

`NOB` The number of "special" regions. 0≤`NOB`≤3. Default: 0

`MHIRFL` Currently not used.

`NHSPE` The type of scale for the particle energy spectra histograms for special regions and surface detectors: dN/dE if `NHSPE`=0 and $dN/d\log_{10}E$ if `NHSPE`=1. Default: 0

`NR1, NZ1` The numbers of bins in r and z directions for the first special region. Default: 120, 200.

`NR2, NZ2` The numbers of bins in r and z directions for the second special region. Default: 120, 200.

`NR3, NZ3` The numbers of bins in r and z directions for the third special region. Default: 120, 200.

RZOB `RZO(4,3)`

Real values which specify the location and size of each special region. It is currently required that these values coincide with boundaries of geometry zones. This card is valid only if `NOB` in the **NOBL** card is ≥1. See Section 10.1.

`RZO(1,i)` The minimum radius of the `i`th special region, `RMI(i)`, where 0≤`RMI`≤`RMAX`. Default: 0.0

`RZO(2,i)` The maximum radius of the `i`th special region, `RMA(i)`, where `RMI`≤`RMA`≤`RMAX`. Default: 0.0

`RZO(3,i)` The minimum (left-hand) z-coordinate of the `i`th special region, `ZMI(i)`, where `ZMIN`≤`ZMI`≤`ZMAX`. Default: 0.0

`RZO(4,i)` The maximum (right-hand) z-coordinate of the `i`th special region, `ZMA(i)`, where `ZMI`≤`ZMA`≤`ZMAX`. Default: 0.0

NSUR `NSURF NTOFF NZH NRH`

Integer variables which give the number of surfaces where all "surface type" histograms will be accumulated, and whether time-of-flight histograms will also be included. See Section 10.2 for the list of the histograms and their IDs. The location of each surface is given in the accompanying **RZTS** data card. The surfaces must have cylindrical symmetry, being either circles in the x-y plane (perpendicular to z-axis), or cylindrical shells oriented along the z-axis. Time-of-flight specifications are given in the accompanying **TOFF** data card.

`NSURF` The number of surfaces, 0≤`NSURF`≤100. Default: 0

`NTOFF` Turns on the generation of time-of-flight distributions for all declared surfaces when `NTOFF=1`. Default: 0

`NZH, NRH` The numbers of bins in z and r directions. Default: 150, 150.

RZTS `(RZTSUR(4,i) UNIT(i)), i=1,100)`

Four real values which specify the location and size of each surface, and a real giving an optional unit number for recording the particles crossing each surface. **It is currently required that these values (or at least some of them) coincide with boundaries of geometry zones**. This card is valid only if `NSURF` in the **NSUR** card is $\geq$1. See Section 10.2 for an example.

`RZTSUR(1,i)` The minimum radius of the `i`th surface, 0.0$\leq$minimum radius$\leq$`RMAX`. Default: 0.0

`RZTSUR(2,i)` The maximum radius of the `i`th surface, minimum radius$\leq$maximum radius$\leq$`RMAX`. Default: 0.0

`RZTSUR(3,i)` The minimum z-coordinate of the `i`th surface, `ZMIN`$\leq$minimum coordinate$\leq$`ZMAX`. Default: 0.0

`RZTSUR(4,i)` The maximum z-coordinate of the `i`th surface, minimum coordinate$\leq$maximum coordinate $\leq$`ZMAX`. Default: 0.0

`UNIT(i)` A real number (not integer), `i`, to open a file named FORT.I which holds a list of all particles crossing the `i`th surface. 81.$\leq$`i`$\leq$90.. If `i=0.`, then no file will be created. Default: 0.

TOFF `TOFMIN TOFMAX TOFSHF`

Real variables which define the time interval, in seconds, for the time-of-flight spectra accumulated for the surfaces declared by the **NSUR** and **RZTS** data cards, when `NTOFF=1`.

`TOFMIN` The begining of the time interval. Default: 0.0

`TOFMAX` The end of the time interval. Default: 1000.0

`TOFSHF` A time shift applied to the time spectra. Default: 0.0

NHBK `NHBK`

Integer variable which defines the number of materials for the "global" energy deposition histograms accumulated for the materials declared by the **HBKE** data card. There must be at least one declared material, entering `NHBK 1` into MARS.INP, when user-defined histograms are used, via the subroutines `MHSETU` and `MFILL`, even if standard histograms are not requested (NOB = NSUR = 0), and even if the user is not interested in the "global" energy deposition for the user defined histograms.

`NHBK` The number of materials. `1`$\leq$`NHBK`$\leq$`5`. Default: 0.0

HBKE `RIM(i),EHMIN(i),EHMAX(i),i=1,NHBK`

Real variables which defines, correspondingly, the material index, lower and upper energy deposition histogram boundaries for the total energy N(ED) (GeV) deposited in the entire system (RIM=0.) and in materials with index RIM$>$0 for histograms with 11$\leq$ID$\leq$15. Default: 0.0

`RIM` The material index. `0.`$\leq$`RIM`$\leq$`100.`. Default: 0.0

`EHMIN` The beginning of the energy deposition interval (GeV). Default: 10^{-6}

`EHMAX` The end of the energy deposition interval (GeV). Default: 1.5$\times E_0$. Note that for a heavy ion projectile E_0 is its total energy.

NCLD `NCLD`

Integer variable which sets the number of materials on the **IMNC** card in which detailed 2-D mass-charge nuclide distributions are calculated, and corresponding input files `NUCLIDES-IM` for the DETRA code generated.

`NCLD` The number of materials on the **IMNC** card for nuclide production calculations. 1≤`NCLD`≤20. Default: 1.

IMNC `IMNC(20)`

Integer variables which specify the material indicies where detailed 2-D mass-charge nuclide distributions are calculated and nuclide production rate files for DETRA generated. Example: `IMNC 1 5 14` to calculate and print nuclide production in first, fifth and fourteenth materials with a **NCLD** card as `NCLD 3`; three input files generated for DETRA have names `NUCLIDES-IM.1`, `NUCLIDES-IM.5` and `NUCLIDES-IM.14`, respectively.

`IMNC(i)` The index of the ith material. Default: 1,19*0.

NDET `NDE1`

Integer variable which sets the number of point-like detectors for the spectrum, flux and energy deposition of low-energy neutrons. The size and location of each detector is specified by the accompanying **FLOC** data card. Active only when `IND(5)=F`.

`NDE1` The number of detectors. `0`≤`NDE1`≤`10`. Default: 0

FLOC `RD XD(10) YD(10) ZD(10)`

Real variables which specify the size and location of each of the declared point-like low energy neutron detectors. Active only when `IND(5)=F` and `NDE1`≥`1`.

`RD` The radius of all declared detectors. Default: 0.5.

`XD(i), YD(i), ZD(i)` The coordinates of the ith detector. Default: 30*0.0

TAPE `NWEGH NWANS NWMJL NWPSI NWNEUN`

Integer flags which control the creation of various output files.

`NWEGH` Only one of the following files is generated in any single job, depending on the value. Default: 0
=17 Write a `MUON.EGH` file with muon-generated EMS for post-run processing and analysis.
=18 Write `TRACK.PLOT` file with particle tracks for `NSTOP` events or for the event `NHIPR` only if it is present on the **NEVT** card.
=19 Write `VERTEX.PLOT` file with particle interaction vertices.
=20 Write `MUON.PLOT` file with particles generated at muon interaction vertices. Those particles are not transported, while muons are.

`NWANS` When set=2, write `ANSYS.ED` file of energy deposition density to be used by the ANSYS system. Default: 0

`NWMJL` When set=12, write `EDMJL.GRA` graphics-oriented file of energy deposition density. Default: 0

`NWPSI` When set=13, write `PSINEU.GRA` graphics-oriented file of neutron spectra. When set=14, write `LENEUTRONS` file of neutrons with E ≤`RLCTNE(IM)`=0.0145 GeV at their origin in material with index `IM` for further transport with a standalone code such as MCNP. These neutrons are not tracked in MARS then, just dumped to the file instead. Default: 0

`NWNEUN` When set=3, write `NEUTRINO` file of generated neutrinos at their origin. Default: 0

HTIR `TIRH TICH`

Two real variables which specify the irradiation and cooling times for residual dose rate distributions scored in histograms and regions of the `MARS.OUT` and `MTUPLE*` files. Active only when `IND(1)=T`.

`TIRH`	The irradiation time (days). Default: 30.
`TICH`	The cooling time (days). Default: 1.

RTIR `TIR(7)`

Real variables which specify the irradiation times for 2-D residual dose rate distributions scored in materials of the `MARS.OUT` file. Active only when `IND(1)=T`.

`TIR(i)`	The `i`th irradiation time (days). Default: 0.5,1.,5.,30.,100.,365.,7300.

CFTD `IFTD`

Integer variable which specifies the sets for fluence-to-dose conversion and radiation weighting factors, w_R. The default provides the effective dose and w_R according to ICRP103 (10 CFR Part 835) as decribed by J.D. Cossairt and K. Vaziri, Fermilab-PUB-08-244-ESH-REV (Dec. 2008) for neutrons. The effective dose and w_R for other particles are calculated according to M. Pellicioni, Rad. Prot. Dos., v. 88 (2000) pp.279-297 and ICRP60. This is for all the "dose equivivalent" distributions scored in histograms and regions of the `MARS.OUT` and `MTUPLE*` files. The previous set - based on ICRP51 - is kept for reference and also used in those (rare) cases where the modern info is absent.

`IFTD`	The control variable: 0 for the modern definition of the Effective dose, and 1 for the old ICRP51 set. Default: 0.

4.2.8 Physics Control – `ALMX, BIAS, ICEM, MCSD, MUON, NUFR, PHOT, RZMN`

ALMX `ALMAXB(6)`

Maximal angles on a step due to magnetic field.

`ALMAXB(6)`	First five numbers are real numbers specifying the maximal angles (rad) allowed on a step due to magnetic field for five energy intervals $\Delta = \|E_0 - E\|/E_0$: <0.001, 0.001-0.01, 0.01-0.3, 0.3-0.99, >0.99. The sixth parameter specifies such an angle for unstable particles in the forced decay mode (`PPIKDEC`≥0.001); useful to control forced decays of an unstable particle beam in long drifts in presence of magnetic field. Some adjustments to the angles are done for all charged particles at low beam energies. Default: 0.0001,0.001, 0.01, 0.1, 0.3, 0.003.

BIAS `PPIKDEC PMUPRMT PMUBEHE PMUGVM PMUANN PPHNUC PELNUC PPBAR`

Variables which provide a global control of biasing in several important processes. Fine tuning of all the biasing schemes is done in user subroutine `BLOCK DATA BLPROCESS` (work in progress). Unstable particle decays are controlled by the `PPIKDEC` parameter and can be modelled either analogously or as forced events with a Russian roulette. The processes controlled by the remaining parameters on this card are modelled as forced events with a Russian roulette. The biasing parameters can be applied only to specific materials via the values on the group of cards **BIPR** through **BIAP** as described above. Such material-dependent parameters will overwrite the global values of the **BIAS** card for the specific material.

PPIKDEC Real number to control unstable particle decays: hadrons (except π^0), muons and heavy ions. It specifies decay modeling in the pure analog way at PPIKDEC=-1, and forced at PPIKDEC$\geq$0.001. For decays of charged pions, kaons and heavy ions on a path to an inelastic nuclear interaction, leakage or ionization absorption, the value of 0.001 $\leq$ PPIKDEC $\leq$ 1 is used as a parameter of the Russian roulette. All decays are suppressed if |PPIKDEC| $<$ 0.001. For muons, any value |PPIKDEC| $\geq$ 0.001 (including PPIKDEC=-1) results in the pure analog decay modeling, otherwise muon decays are suppressed (at |PPIKDEC| $<$ 0.001). Default: -1.

PMUPRMT Real number specifying the parameters of the inclusive (forced) prompt muon production in the inelastic nuclear interaction vertex. It is forced with the appropriate statistical weight at PMUPRMT = 1, forced with the Russian roulette at 0.001 $\leq$ PMUPRMT $<$ 1, or suppressed at PMUPRMT = 0. Default: 0.05.

PMUBEHE Real number specifying the parameters of the inclusive (forced) Bethe-Heitler muon production. It is forced with the appropriate statistical weight at PMUBEHE = 1, forced with the Russian roulette at 0.001 $\leq$ PMUBEHE $<$ 1, modelled exclusively at PMUBEHE = -1, or suppressed at PMUBEHE = 0. Alternatively, the Bethe-Heitler muon production cross-section can be increased 1 to 500 times with -500 $\leq$ PMUBEHE $<$ -1; an appropriate correction to a statistical weight assures correct results. It is automatically set to -1 for KEMINCL = -1. Default: 0.03.

PMUGVM Real number specifying the parameters of the inclusive (forced) muon production via vector mesons generated in photo-nuclear reactions at $E > 1.5$ GeV when IND(10)=T (default). It is forced with the appropriate statistical weight at PMUGVM = 1, forced with the Russian roulette at 0.001 $\leq$ PMUGVM $<$ 1, modelled exclusively at PMUGVM = -1, or suppressed at PMUGVM = 0. It is automatically set to -1 for KEMINCL = -1. Default: 0.02 when the primary beam particle is photon, electron or positron (I0 = 9, 10 or 11 on the IPIB card), otherwise it is 0.005 at $E > 20$ GeV, or 0 at $E \leq 20$ GeV.

PMUANN Real number specifying the parameters of the inclusive (forced) $e^+e^- \rightarrow \mu^+\mu^-$ annihillation events. It is forced with the appropriate statistical weight at PMUANN = 1, forced with the Russian roulette at 0.001 $\leq$ PMUANN $<$ 1, modelled exclusively at PMUANN = -1, or suppressed at PMUANN = 0. It is automatically set to -1 for KEMINCL = -1. Default: 0.03.

PPHNUC Real number specifying the parameters of the inclusive (forced) photo-nuclear events. It is forced with the appropriate statistical weight at PPHNUC = 1, forced with the Russian roulette at 0.001 $\leq$ PPHNUC $<$ 1, modelled exclusively at PPHNUC = -1, or suppressed at PPHNUC = 0. It is automatically set to -1 for KEMINCL = -1. Default: 0.02.

PELNUC Real number specifying the parameters of the inclusive (forced) electro-nuclear events. It is forced with the appropriate statistical weight at PELNUC = 1, forced with the Russian roulette at 0.001 $\leq$ PELNUC $<$ 1, modelled exclusively at PELNUC = -1, or suppressed at PELNUC = 0. It is automatically set to -1 for KEMINCL = -1. Default: 0.003 when the primary beam particle is electron or positron (I0 = 10 or 11 on the IPIB card), otherwise it is 0.003 at $E < 300$ GeV, or 0 at $E \geq 300$ GeV.

PPBAR Real number specifying the the parameters of the inclusive (forced) antiproton production. It is forced with the appropriate statistical weight at PPBAR = 1,

forced with the Russian roulette at 0.001 $\leq$ PPBAR $<$ 1, or suppressed at PPBAR = 0. Default: 0.05.

ICEM C1CEM C2CEM EMODEL IQGSM NEVTYPE

Variables which control hadron event generator and use of cascade-exciton and quark-gluon string models, and evaporation scheme.

C1CEM, C2CEM Real numbers specifying the energy $ECEM$ below which the CEM code is called on a nucleus with atomic mass $A > 3$, where $ECEM =$C1CEM$-$(C2CEM$\times Z$). Default: 5.0, 0.

EMODEL Real number specifying the transition energy between the use of high-energy and low-energy event generator algorithms. Default: 5 GeV.

IQGSM An integer that globally switches between the inclusive and exclusive event generators at nuclear inelastic interactions. For IQGSM=0, the default inclusive model is used. For IQGSM=1, exclusive modeling with the LAQGSM is done at $E > ECEMLQ = 1 + A/65$, or/and $A < 3$, and at all energies for $\overline{p}, K^{\pm}, d, t,^3 He,^4 He, \overline{n}$, hyperons and heavy ions; the CEM model is used otherwise. For IQGSM=2, exclusive modeling with the LAQGSM code is always done for everything at all energies. This control can also be done via the MTQG card individually for each material (see Section 4.2.4). IQGSM=1 and IQGSM=2 are recommended for particle production or precise physics studies. This mode can be time-consuming. Default IQGSM=0 is much more appropriate for shielding studies.

NEVTYPE The number of evaporated particle types to consider in each nuclear interaction. Minimum is six ($p, n, d, t,^3 He,^4 He$), maximum is 66 (up to ^{28}Mg). Default: 17 (up to ^{12}Be). It is recommended to use NEVTYPE=6 for shielding applications (to reduce CPU time), and NEVTYPE=66 if one needs precise nuclide mass/charge distributions at $7 < A < 20$.

MCSD KMCS KDLE MCSDLE INEXDL

Control of the ionization energy loss processes and scattering acting on muons, charged hadrons and heavy ions: nuclear elastic scattering (HES) at E>0.014 GeV for neutrons and at all energies for all other particles, and multiple Coulomb scattering (MCS) for all charged particles. This feature is primarily for study purposes, to isolate and study the modeling of specific phenomena. Aside from such studies, users are advised to leave these parameters at their default values, although the use of KDLE=0 can save a fare amount of CPU-time spent tracking high energy muons, if the details of the produced muons are not important to the user's results.

KMCS An integer whose value turns on or off the elastic and Coulomb scattering. For KMCS = -1, scattering is turned off. For KMCS = 0, only hadron elastic scattering is turned on. For KMCS $>$ 0, both hadron elastic scattering and Coulomb scattering are turned on, and in this case the value k of KMCS gives the number of levels of the particle cascade tree to which the scattering applies. (Default: KMCS=15).

KDLE Integer number controlling the ionization energy loss modeling. For KDLE=0 a simple continuous dE/dx process is applied. For KDLE>0 a more sophisticated algorithm is applied, which includes delta-electron production and direct e^+e^- pair production, and the simulation of the resulting induced electromagnetic showers for the first $k \leq$KDLE-levels of the hadron cascade tree and for all muons. For KDLE=-1 there is no transported charged particle energy loss at all (used for special studies only). (Default: KDLE=15).

`MCSDLE` Integer number controlling the correlation in ionization energy and Coulomb scattering. For `MCSDLE = 1`, correlated energy loss straggling and Coulomb scattering are modeled for all charged particles. For `MCSDLE = 0`, these two processes are modeled independently. Default: 1.

`INEXDL` Integer number controlling the modelling mode of delta-electron production. For `INEXDL = 1`, exclusive modelling is used. For `INEXDL = 0`, inclusive modelling is used. Default: 0.

`MUON` `EFLU IPAR KPHA ISOURCE`

Variables which control the modeling of bremsstrahlung and direct pair production by muons, extend of hadron cascades, and muon and neutrino beams.

`EFLU` The energy above which full modeling of bremsstrahlung and direct pair production by muons is done; at lower energies a continuous energy loss approximation for these two processes is used. Default: 5 GeV.

`IPAR` A parameter which – if equals zero – can be used to turn off the default modeling of direct pair production by muons and use instead the continuous energy loss approximation for this process at all energies. Default: `IPAR=1`.

`KPHA` The number of hadron generations to follow, i.e. the number of levels in the hA vertex tree. Default: 55.

`ISOURCE` A parameter which defines the beam type: original at `ISOURCE=0` and `ISOURCE=1`, broad at `ISOURCE=2`, μ decays in a straight section ($z < 0$, at `ZMIN`-0) at `ISOURCE=3`, and μ decays in a ring at `ISOURCE=4`. Default: 0.

`NUFR` `ZLNUFR ZUNUFR RUNUFR`

Real variables which define a cylindrical region where ν interactions are forced.

`ZLNUFR, ZUNUFR, RUNUFR` The cylindrical region where neutrinos are forced to interact with matter when `IND(8)=T`. Default: 0., 0., 0.

`PHOT` `ELSYN EMISYN EEGHM ISYNHMU ISYNEMS ISYWRT NOSTPMV KEMINCL`

Variables which control features of electro-magnetic shower development.

`ELSYN` Real number specifying the synchrotron emission for charge particle i of energy `E>ECUT=ELSYN`$\times m_i/m_e$, where m_e is electron mass. Default: 0.05 GeV.

`EMISYN` Real number specifying a synchrotron photon energy threshold. For `E>EMISYN` induced EMS are treated at photon energy; otherwise that energy is deposited locally. Can be as low as 10^{-6} GeV. Default: 2×10^{-4} GeV.

`EEGHM` Real number specifying an energy threshold. For `E>EEGHM`, full simulation of muon, δ-electron and e-production and their showers along hardon tracks, local deposition otherwise. Default: 0.001 GeV.

`ISYNHMU` Flag for synchrotron radiation generation by hadrons, muons and heavy ions. `ISYNHMU=0` provides analog (exclusive) generation; `ISYNHMU=1` provides inclusive generation. Default: 0.

`ISYNEMS` Flag for synchrotron radiation generation in electromagnetic showers. `ISYNEMS=0` provides analog (exclusive) generation; `ISYNEMS=1` provides inclusive generation. Default: 1.

`ISYWRT` Flag which activates, when `ISYWRT=1`, writing of parameters of first 30000 generated synchrotron photons into a `fort.79` file. Default: 0.

`NOSTPMV` Maximum number of steps for a particle in vacuum. Default: 100000.

`KEMINCL` The value for the electromagnetic shower vertex as well as $n-\gamma$ and other neutron interaction verticies (while in the MCNP mode) modelling switch. For `KEMINCL`=0, the inclusive mode is used. For `KEMINCL`=-1, exclusive modeling is done for all electromagnetic shower and $n-\gamma$ verticies; five EMS bias keys (see the **BIAS** data card) are automatically converted to -1, i.e. to the exclusive mode. For `KEMINCL`= $N \geq$1, exclusive modeling is used for the first N generations and inclisive one for remaining higher generation levels of the electromagnetic shower. Default: 10.

RZMN `RMI514 RMA514 ZMI514 ZMA514 RMINTR RMAXTR ZMINTR ZMAXTR`

Real variables which define two cylindrical volumes; each volume is defined by minimum and maximum radii, and minimum and maximum z coordinates. The first four variables define a volume where the `IND(5)=T` condition applies. The second four variables define a volume where the particles entering are recorded in `TRACK.PLOT` and `VERTEX.PLOT` files, if those files have been requested via the **TAPE** data card.

`RMI514, RMA514, ZMI514, ZMA514` The cylindrical region where `IND(5)=T` applies. Default: 0.0 `RMAX` 0.0 `ZMAX`

`RMINTR, RMAXTR, ZMINTR, ZMAXTR` The cylindrical region where particles are recorded. Default: 0.0 `RMAX` 0.0 `ZMAX`.

4.2.9 Termination – STOP

STOP (no parameters)

Terminates the normal MARS data card list. Any cards given after this are meaningless, unless one runs in a mode which uses an imported geometry description and which places the appropriate data lines after the **STOP** card. See Section 3.1.2 for a summary of imported geometries which can be used.

4.3 Examples Using MARS.INP Parameters

The following examples use only the `MARS.INP` input deck to fully describe the modeled geometry. The user subroutines are not utilized and left as dummies. In truth, most problems of interest will involve more complex geometry descriptions; however, these examples serve to outline the basics of the input deck. More complex examples which utilize the user subroutines in conjunction with parameters in the input deck are given in Section 5.21.

4.3.1 Z-Sandwich Geometry

z-sandwich geometry is one where the same material extends outward for all values of r. The material changes only along z boundaries. r-sandwich geometry is one where the same material extends along the z-axis, and the material changes only at concentric r boundaries. The user must choose either one or the other to represent the Standard zones in his model.

Figures 14 and 15 show a simple z-sandwich Standard geometry. The pictures are produced by the Mars GUI interface. Figure 14 is a color-filled diagram of the Standard zones, with each color being a distinct

material: light blue is air, orange is steel, and tan is concrete. Figure 15 is the same zone diagram, with no color fill, and with a 2-D histogram of the charged hadronic flux density super-imposed over it.

The input deck for this example is :

```
MARS15 Example:  Z-sandwich geometry
/home/mokhov/restricted/mars15/dat
INDX 2=T 6=T
NEVT 1000
ENRG 100.  0.05 0.05
IPIB 3 3
BEAM 5.0 5.0 0.1 0.1
SMIN 0.2 5.
NMAT 3
MATR 'FE' 'AIR' 'CONC'
NLNG 3
ZSEC 50.  150.  350.  1252=4 4 2501=2 1 3
NLTR 1
RSEC 100.  51=4
NOBL 1
RZOB 0.  100.  0.  350.
STOP
```

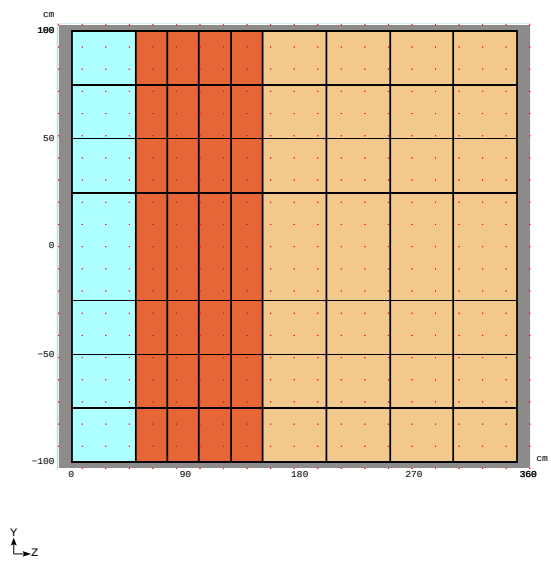


Figure 14: Plan View of the z-sandwich Example. The grid lines outline the declared Zones.

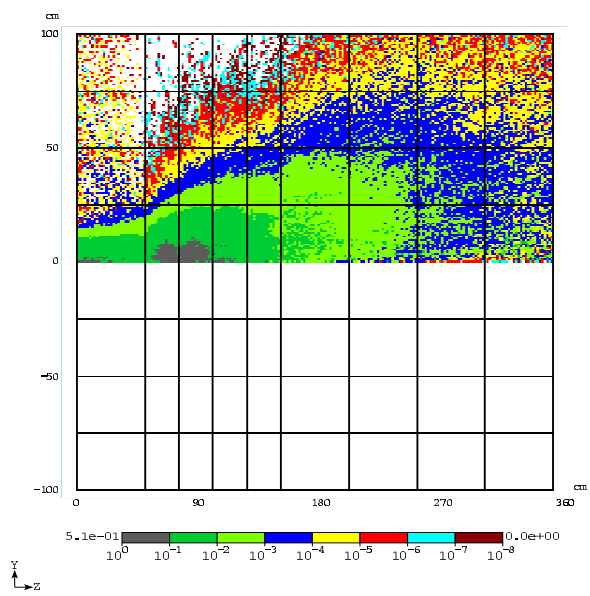


Figure 15: The same z-sandwich geometry with a histogram of the hadronic flux density overlaid.

At the top of the deck `INDX 2=T` declares this to be a z-sandwich type geometry, and this means the zone materials are declared in the **`ZSEC`** card. The **`NLNG`** and **`ZSEC`** cards declare 3 major z sections, with the 2nd and 3rd sections each sub-divided into 4 sub-sections (`1252= 4 4`). The 1st major section is composed of air, the 2nd steel, and the 3rd concrete (`2501=2 1 3`), where the `2 1 3` refer to the order those materials are listed in the **`MATR`** card. The **`NLTR`** and **`RSEC`** cards declare only 1 major r section, which is in turn divided into 4 sub-sections (`51=4`). These sections and sub-sections are delineated by the grid lines seen in Figures 14 and 15. The sections are always defined from minimum to maximum z, or, from left to right in the MARS coordinate system. The air section is $50cm$ deep, the steel $100cm$ deep, and the concrete $200cm$ deep; the `50. 150. 350.` values listed reflect the locations of the right-hand

boundary of each major section.

The input beam is composed of 100 GeV π^+, and the **IPIB** and **BEAM** cards define a fairly large, spreading beam - the example could be a model of a secondary pion beam striking a beam dump. Energy cutoffs of 0.05 GeV are given for both the EM and EPSTAM parameters in the **ENRG** card. This setting, in conjunction with IND 6=T means that the user is interested primarily in star density and in overall flux density patterns, and not in a detailed analysis of energy deposition. The zone sizes are relatively large, and the global step sizes in the **SMIN** card are set correspondingly. No special energy cutoffs or step sizes are specified beyond the global values.

The **NOBL** and **RZOB** cards declare one volume, encompassing all declared zones. A standard set of histograms then accumulates various parameters, and are written to a mars.hbook file. This file can be opened from the GUI interface, (a separate job from the one which produced the hbook file), and selected histograms projected onto the geometry, as in Figure 15. **NOBL** histograms are accumulated in a cylindrical geometry, and so display only from R=0 to R=Rmax. The GUI interface displays the geometry in X-Z, Y-Z, or X-Y planes. Thus, the color maps display only in the upper half of the plan or elevation view. The particular histogram shown is a map of the charged hadronic flux, each color being an order of magnitude in flux density. One can see the broad pion beam traversing the air (green and blue color bands), showering and spreading out in the steel and concrete of the beam dump, and some amount of back-scattered particles from the front face of the steel back into the air above the incoming beam. More details on both histograms and the GUI interface are given in Sections **??** and 12 respectively.

4.3.2 R-Sandwich Geometry

Next, examine a similar simple geometry, also a beam dump type, but using r-sandwich geometry. r-sandwich geometry is one where the same material extends along the z-axis, and the material changes only at concentric r boundaries. Again, the user must choose either z or r sandwich geometries to represent the Standard zones in his model.

Figures 16 and 17 show a simple r-sandwich Standard geometry. The pictures are again produced by the Mars GUI interface. Figure 16 is a color-filled diagram of the Standard zones, with the colors similar to the z-sandwich example: light blue is air, orange is steel, and tan is concrete. Figure 17 gives a cross-section view of the zone diagram.

The input deck for this example is :

```
MARS15 Example:  R-sandwich geometry
/home/mokhov/restricted/mars15/dat
INDX 6=T
NEVT 1000
ENRG 100.  0.05 0.05
IPIB 3 3
BEAM 5.0 5.0 0.1 0.1
SMIN 0.2 5.
NMAT 3
MATR 'FE' 'AIR' 'CONC'
NLNG 1
ZSEC 350.  1251=7
NLTR 3
RSEC 30.  80.  100.  51=2 2 101=1 3 2
NOBL 1
RZOB 0.  100.  0.  350.
STOP
```

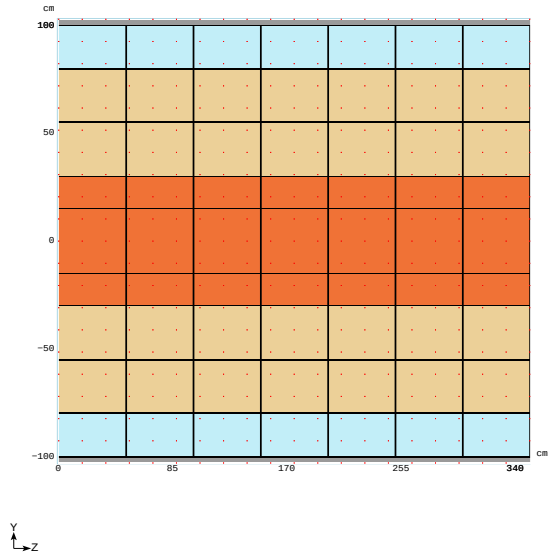


Figure 16: Plan View of the r-sandwich Example. The grid lines outline the declared Zones.

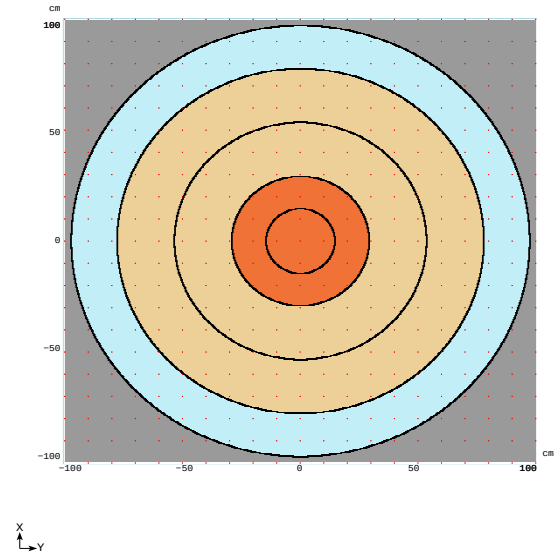


Figure 17: A cross-section view of the same r-sandwich geometry.

At the top of the deck `INDX 2=F` declares this to be a r-sandwich type geometry, and this means the zone materials are declared in the **RSEC** card. The **NLTR** and **RSEC** cards declare 3 major r sections, with the 1st and 2nd sections each sub-divided into 2 sub-sections (`51=2 2`). The 1st major section is composed of steel, the 2nd concrete, and the 3rd air (`101=1 3 2`). The materials are listed in the same order as for example 1, where the `2 1 3` refer to the order those materials are listed in the **MATR** card. The **NLNG** and **ZSEC** cards declare only 1 major z section, which is in turn divided into 7 sub-sections (`1251=7`). These sections and sub-sections are delineated by the grid lines seen in Figures 16 and 17. The sections are always defined from minimum to maximum r, or, from the center r =0 to the outermost r value. The inner steel section is 30cm in radius, the surrounding concrete section is 50cm thick, and there is a 20cm thick layer or air surrounding the concrete; the `30. 80. 100.` values listed reflect the locations of the outer boundary of each major r section.

The input beam is described identically to the z-sandwich example. This example could also be a model of a secondary pion beam striking a beam dump, only the emphasis is on mapping the radial deposition outward through the different layers of materials. The same energy cutoffs and global step sizes are used as for the z-sandwich example. Also the identical **NOBL** and **RZOB** cards are used, encompassing the entire declared geometry. However, since the materials are arranged in a different geometry, the hadronic flux map, Figure 18 appears slightly different from that for the z-sandwich example. Since the color maps display only from r =0 to r =`Rmax`, the frame of the view has been modified to show just the upper half of the geometry.

4.3.3 Thin Window

The example of a thin window shows the usage of material-dependent parameter settings, such as step lengths. The model is of a small proton beam, traveling through helium. The beam encounters a thin titanium window, and then a region of air. There are no interactions of significance in the helium or the air, and allowing large step sizes in those areas can reduce the overall running time. However, one must be careful of the combination of small-size zones and large step sizes. It is fairly obvious that if the step size is of the same order as a thin zone size, then some generated particles may ”jump over” the thin zone, and some results for the the thin zone, such as energy deposition, will be incorrect. What is less obvious is

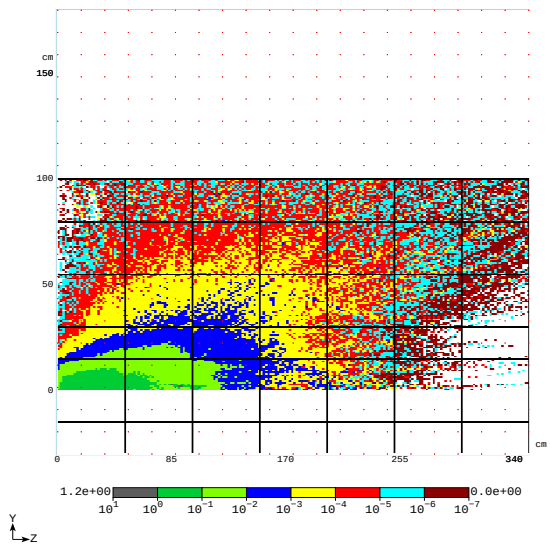


Figure 18: Plan View of the r-sandwich Example, with a map of the hadronic flux overlaid.

controlling the accuracy with which the zone boundary is located.

MARS uses a "zig-zag" approach to locate a boundary. The pilot step, the global STEPH, or the material dependent RLSTEH is first applied; if within that step the zone number changes, then MARS will iterate back and forth, in smaller and smaller steps, to locate the boundary between the zones. The minimum size of these iterative steps is given by the global STEPEM or the material dependent RLSTEM. Therefore, the zone boundary will be located to the accuracy of the size of STEPEM or RLSTEM. Moreover, one does not want to have a large value of STEPEM on one side of a boundary, and a smaller value of RLSTEM on the other side of a boundary. A better strategy is to make big steps until within a certain distance from the thin zone, and then reduce the step size on approach, step through the thin zone, continue taking small steps for a certain distance past it, and then increase the step size again. This requires dividing the material in front of and behind the thin material into sections where the step size can be adjusted.

The following deck sets up this sort of controlled boundary localization:

```
MARS15 Example:  Thin Window
/home/mokhov/restricted/mars15/dat
INDX 2=T
NEVT 1000
ENRG 50.
IPIB 4 3
BEAM 1.0 1.0 0.02 0.02
SMIN 0.2 1.0
NMAT 5
MATR 'HE' 'HE' 'TI' 'AIR' 'AIR'
NLNG 5
ZSEC 20.  22.  22.5 24.5 44.5 2501=1 2 3 4 5
NLTR 3
RSEC 1.  5.  10.  51=4 4 2
MTSM 2=0.01 0.01 0.01
MTSH 2=0.1 0.1 0.1
STOP
```

The deck is set up for z-sandwich geometry, given by `INDX 2=T` and by the fact that the assignment of materials to zones is given in the **ZSEC** card. There are only three unique materials among the five listed, and this is done so that different step sizes can be assigned to the zones of helium and air adjacent to the titanium window. The total amount of helium and air on either side is $22cm$, and is split into large zones $20cm$ deep and smaller zones $2cm$ deep adjacent to the window. The window itself is $0.5cm$ thick (not a very thin window, but the example is for illustrative purposes). Each of these five zones is given a unique material ID number. Next, the global step sizes, given by the **SMIN** card, are sized for the $20cm$ deep zones of helium and air. The **MTSM** and **MTSH** cards adjust the step sizes downward for materials 2, 3, 4, with the step sizes in those 3 materials being the same values in this example. These smaller step sizes are appropriate for the thickness of the thin window; taking similar small steps through the helium and air on either side of the window assures that the window boundary will be accurately located, and data like energy deposition correctly tabulated.

In keeping with a desire for accurate energy deposition data, the deck allows all the default energy thresholds, specifying only the energy of the incoming pion beam; the beam is described as a simple diverging gaussian. Radial zones are defined, in the **RSEC** card, so that the data can used to show how the energy deposition changes versus radial distance; an example of extracting and plotting this data is given in the output examples in Section 11.1.2.

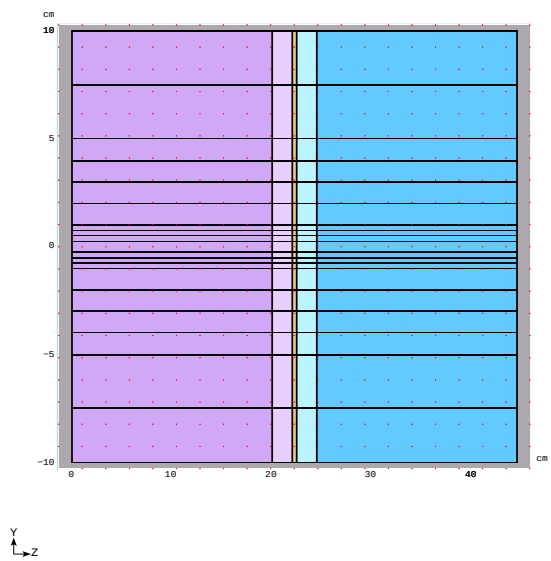


Figure 19: Plan View of the Thin Window example.

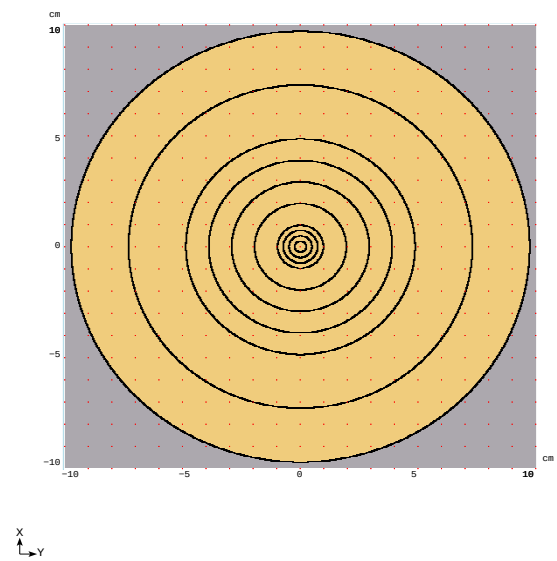


Figure 20: A cross-section view of the Thin Window example.

Figures 19 and 20 show the plan and cross-section views of the Thin Window described by the input deck. The helium zones are a light purple, and the air zones a light blue, with the lighter tones being the 2cm deep zones just in front of and behind the thin window, where the step lengths are reduced. The thin window itself is a gold color. The cross section view shows the radial zones in the thin window, where data on parameters like energy deposition are accumulated.

4.4 Extended Geometry `GEOM.INP` Input File

Extended Geometry zones are another way to describe the user's model, when the Standard zones described by the **ZSEC** and **RSEC** cards in the `MARS.INP` file are insufficient. The term "extended" refers to an extension beyond the r-z-ϕ symmetric Standard zones. Extended Geometry zones are constructed from

a set of contiguous or overlapping geometrical shapes, which currently consists of box, cylinder, sphere, cone, tetrahedron, elliptical tube, toroid, elliptical cone and conical sector. This is similar to the geometry description method used by GEANT. There can be up to 10^5 Extended Geometry shapes declared, where each declared shape is counted as a MARS zone. A single shape can be sub-divided into sub-zones, with each sub-zone counting towards the maximum limit.

Only extended geometry option provides exact crossing of particle tracks with surfaces that prevents small regions within a large volume from being skipped over. In other four options, boundary localization is based on iterative algorithm and user needs to take care of appropriate region numbering, pilot steps and localization parameters. Zone overlapping in the extended geometry mode means in particular that in this mode a user doesn't need to care about zone numbering when one large zone is filled with small objects.

The specifications for the geometric shapes are controlled by lines in the input file `GEOM.INP`. As indicated in Section 4.2.2, setting `IND(3)=T` in the `MARS.INP` file activates the reading of the `GEOM.INP` file and the implementation of the Extended Geometry Zones described by it.

The user is reminded that the overall outer boundary of the model ("mother volume") is set by the **`ZSEC`** and **`RSEC`** cards in the `MARS.INP` file. There must always be at least one Standard zone, and all the declared Extended Geometry zones must be contained within this outermost Standard zone boundary. In assigning zone numbers, MARS begins with the Standard zones, numbered $1:$ `NFZP`, the maximum number of declared Standard zones. Extended zones are next, numbered starting from `NFZP`$+1$. There can be no more than $100,000$ declared Extended zones. MARS will assign zone numbers to Extended Geometry shapes in the order in which they appear in the `GEOM.INP` file. For an introduction to zone numbering, see the discussion in Section 3.1.

First line of the `GEOM.INP` file is a title, consisting of no more than 80 characters. All the subsequent lines are geometric information, either *shape* data lines or *transformation* data lines. A *shape* data line is used to declare and describe a single Extended Geometry shape. A *transformation* line describes a transformation matrix, translation and rotation, which can be applied to any Extended Geometry shape. Each shape can be subdivided into a number of sub-regions. Blank lines can be inserted between data lines, for file organizational purposes, and will be skipped by the routines reading the file. Comments are allowed at the end of any line, by starting them with an exclamation sign.

The Extended Geometry shape data lines can be listed in any order, for example they are not required to be listed from left to right in the coordinate system; however there are two important points to keep in mind. First, the MARS zone number will be assigned according to the order in which the lines appear in the file. Second, overlapping shapes are layered according to the order the lines appear in the file, with the top layer shapes being listed at the top of the file, and so on down (see Fig. 21 as an example). The user needs to keep these points in mind, and order shapes appropriately so as to obtain the desired result.

Table 7: Shape types, listed by the shape ID number, which are implemented in Extended Geometry and used in the `GEOM.INP` file, with their descriptors, their Reference Point RP location, and definition of their Local Coordinate System LCS.

Shape	`abs(NT)`	Parameter List, RP and LCS definitions	
Box	1	`C1`	the box half-size along the LCS x axis
		`C2`	the box half-size along the LCS y axis
		`C3`	the box length (`NT`>0) or half-length (`NT`<0) along the LCS z axis

continued on next page

Table 7: *table continued*

Shape	abs(NT)	Parameter List, RP and LCS definitions	
		NXSB	the number of subdivisions in x, default: 1
		NYSB	the number of subdivisions in y, default: 1
		NZSB	the number of subdivisions in z, default: 1
		RP	defined as the center of the plane perpendicular to the z-axis and with the minimum coordinate location
		LCS	the box's x and y LCS axes are parallel to two perpendicular edges of the minimum coordinate location z plane; the z axis points from the center of the minimum coordinate location z plane to the center of the maximum coordinate location z plane. The x, y, z axes of the LCS are parallel to those of the GCS.
Cylinder	2	C1	the inner radius of the cylinder, $= 0.$ for a solid cylinder, and $> 0.$ for a cylindrical tube
		C2	the outer radius of the cylinder
		C3	the cylinder length (NT>0) or half-length (NT<0) along the LCS z axis
		NZSB	the number of subdivisions in LCS z, default: 1
		NRSB	the number of radial subdivisions, default: 1
		RP	defined as the center of the plane perpendicular to the z-axis and with the minimum coordinate location
		LCS	the LCS z-axis is parallel to the GCS. The z axis points from the center of the minimum coordinate location cylinder end to the center of the maximum coordinate location cylinder end. The orientation of the LCS x and y axes is arbitrary.
Sphere	3	C1	the inner radius of the sphere, $= 0.$ for a solid sphere, and $> 0.$ for a spherical shell
		C2	outer radius of the sphere
		NRSB	the number of radial subdivisions, default: 1;
		RP	defined as the sphere center
		LCS	the orientation of the axes is arbitrary.
Cone	4	C1	the inner radius at the lowest LCS z, the cone base; $= 0.$ for a solid, and $> 0.$ for a hollowed cone
		C2	the outer radius at the lowest LCS z, the cone base
		C3	the inner radius at the highest LCS z, the cone top; $= 0.$ for a solid or closed shape, and $> 0.$ for a tubular cone
		C4	the outer radius at the highest LCS z, the cone top
		C5	the cone length (NT>0) or half-length (NT<0) along the LCS z axis
		NZSB	the number of subdivisions in LCS z; default: 1
		NRSB	the number of radial subdivisions; default: 1
		RP	defined as the center of the minimum coordinate location LCS z plane, which is the cone base
		LCS	The z-axis points from center of the cone base to the center of the cone top; the LCS z-axis is parallel to the GCS z-axis. The orientation of the LCS x and y axes is arbitrary.
Tetrahedron	5		Defined by a set of 4 verticies. The user selects one vertex to be the shape's Reference Point RP, and enters those coordintes as (XR,YR,ZR) in the Shape specification data line
		C1:C3	coordinates of the second of 4 verticies, $(x2, y2, z2)$
		C4:C6	coordinates of the third of 4 verticies, $(x3, y3, z3)$
		C7:C9	coordinates of the fourth of 4 verticies, $(x4, y4, z4)$

continued on next page

Table 7: *table continued*

Shape	abs(NT)	Parameter List, RP and LCS definitions	
		RP	defined by the user, as the $x1, y1, z1$ coordinates of the first vertex
		LCS	identical to GCS
Elliptical Tube	6	C1	the inner radius along vertical x-axis
		C2	the inner radius along horizontal y-axis
		C3	the tube wall thickness along one of the axes;
			C3$= 0$ correspondes to a solid (filled) ellipse defined by C1 and C2
		C4	the tube length (NT$>$0) or half-length (NT$<$0)
			along the LCS z axis
		NZSB	the number of subdivisions in LCS z, default: 1
		RP	defined as the center of the plane perpendicular to the z-axis
			and with the minimum coordinate location
		LCS	the LCS z-axis is parallel to the GCS. The z axis points from the
			center of the minimum coordinate location cylinder end to the center of the
			maximum coordinate location cylinder end.
Toroid	7		currently unavailable
Elliptical Cone	8	C1	the inner radius along horizontal y-axis at the lowest LCS z, the cone base;
		C2	the outer radius along horizontal y-axis at the lowest LCS z, the cone base;
		C3	the outer radius along vertical x-axis at the lowest LCS z, the cone base;
			if C3=C2 then the cone is round, otherwise all other lateral dimensions
			are squashed appropriately; the ratio C3/C2= is the scaling factor
		C4	the inner radius along horizontal y-axis at the highest LCS z, the cone top;
		C5	the outer radius along horizontal y-axis at the highest LCS z, the cone top
		C6	the cone length (NT$>$0) or half-length (NT$<$0)
			along the LCS z axis
		NZSB	the number of subdivisions in LCS z; default: 1
		NRSB	the number of radial subdivisions; default: 1
		RP	defined as the center of the minimum coordinate location LCS z plane,
			which is the cone base
		LCS	The z-axis points from center of the cone base to the center of
			the cone top; the LCS z-axis is parallel to the GCS z-axis.
Conical sector	9	C1	the inner radius at the lowest LCS z, the cone base;
		C2	the outer radius at the lowest LCS z, the cone base;
		C3	the inner radius at the highest LCS z, the cone top;
		C4	the outer radius at the highest LCS z, the cone top
		C5	the cone length (NT$>$0) or half-length (NT$<$0)
			along the LCS z axis
		NZSB	the number of subdivisions in LCS z; default: 1
		NRSB	the number of radial subdivisions; default: 1
		NASB	the number of azimuthal subdivisions; default: 1
		PHI1	first angle (degrees) defining the sector; default: 0.
		PHI2	second angle (degrees) defining the sector; default: 360.
			if PHI1 $<$ PHI2, a clockwise sector is defined,
			otherwise, a counterclockwise sector is defined
		RP	defined as the center of the minimum coordinate location LCS z plane,
			which is the cone base
		LCS	The z-axis points from center of the cone base to the center of
			the cone top; the LCS z-axis is parallel to the GCS z-axis.

Currently the Conical Sector (shape no. 9) can't overlap the adjacent zones while the opposite is allowed.

Each Extended Geometry shape is specified by a single line of data; the data is unformatted, and sepa-

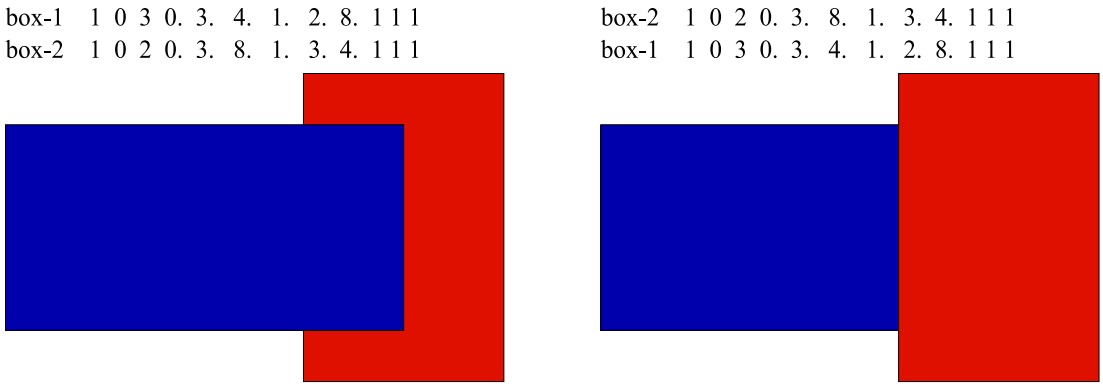


Figure 21: The diagrams demonstrate shape overlapping based upon the order in which the shape definitions appear in the `GEOM.INP` file.

rated by blank spaces. The syntax of an Extended Geometry shape volume description line is

`VNAM NT NTR IM XR YR ZR C1 C2 ...  Cn NSB1 NSB2 NSB3`

where

- `VNAM` is the user's identifying name for the shape, in format (A8). In the current version, to avoid a conflict with transformation matrix descriptions, it must not start with `TR`.
- `NT` is the shape type ID number, an integer $1 \leq |\texttt{NT}| \leq 6$. Table 7 has the correspondence between shapes and ID numbers. If `NT`> 0, the reference point (RP) is on the front surface as described in Table 7. If `NT`< 0, the reference point (RP) is at the shape center (not used for sphere and tetrahedron).
- `NTR` is the transformation matrix ID number; 1 stands for the transformation matrix described on the `TR1` card, 2 stands for the transformation matrix described on the `TR2` card, etc; enter 0 if this shape has no transformation applied.
- `IM` is the material number (index) for the material this shape is made of. This is the same material index as defined by the order of materials in the **`MATR`** card in the `MARS.INP` deck; $0 \leq \texttt{IM} \leq 100$
- `XR,YR,ZR` are the coordinates of a reference point (RP) in the Local Coordinate System (LCS) for this shape. Table 7 gives the location of the RP on each shape type. Table 7 also defines the LCS for each shape, and how it relates to the MARS coordinates (also referred to here as the Global Coordinate System, GCS).
- `C1:Cn` are additional parameters describing the shape. Each shape has a different number of parameters, and these are defined in Table 7. The data line should hold only as many parameters as are defined for the shape. If a parameter is left at it's default value, *and all the following parameters are also left at their defaults*, then they do not need to be listed on the data line; otherwise all parameters used by the shape definition must be listed.
- `NSB1,NSB2,NSB3` are the numbers of optional subdivisions in LCS along corresponding directions (see shape descriptions in the above table).

Transformation matricies are implemented via `TR` cards whose syntax is

`TRn XT YT ZT` α β γ

where

- `TRn` is the transformation matrix identifier, and always starts with "`TR`" followed by

an integer n, 0≤n≤500. Integer n is the transformation ID NTR used in the shape volume data line, for shapes which utilize a transformation.

XT,YT,ZT — are the translation coordinates, applied to the shape's Reference Point RP

α,β,γ — are the rotation angles (in degrees) between the shape's LCS axes and the GCS axes around x-axis ("yaw"), y-axis ("pitch"), and z-axis ("roll") with z-direction of movement; clockwise (positive) or counterclockwise (negative). If there is a rotation around more than one axis, the rotation sequence is γ (around z-axis), then β (around y-axis), and then α (around x-axis)

A **STOP** card, one it's own line and with no parameters, terminates the GEOM.INP data card list.

Remember that there must be at least one Standard zone defined in the MARS.INP file, and this one zone, or the largest of the defined Standard zones, must contain all other defined volumes, including the Extended Geometry zones. The user must determine the extent of his Extended zones, and define an appropriate Standard zone to surround all of them.

If Non-Standard zones also exist in the users model, then these have priority over the Extended zones. When MARS has a particle location (x, y, z) the user routine REG1 is checked first; if code within that routine describes a zone which contains the (x, y, z) location, then MARS is given this Non-Standard zone number, and the Extended Geometry module is not called. If instead the program exits from the REG1 subroutine with no zone number assigned, then the Extended Geometry module will be called. The reverse does not occur – if upon exiting the Extended Geometry module a zone number still has not been assign to the input (x, y, z) location, MARS will not return to the REG1 subroutine; it will likely assign that location a negative zone number, which becomes a $blackhole$ material, and any particle at that location will be dropped from the tracking list.

By default, there is no optimization in Extended Geometry, meaning that the shapes are always analyzed in the order they appear in the GEOM.INP file. Shapes defined first have a higher priority than shapes defined in subsequent lines. MARS enters the Extended Geometry module with the location of some particle (x, y, z); at the first shape encountered which contains that (x, y, z), the code exits the geometry description, handing MARS the zone number corresponding to that shape. At the next entry, with a new location, the geometry description code begins again at the top of the GEOM.INP file. The user needs to think about descriptions where shapes overlap, and order them appropriately so as to obtain the final desired zone shapes and numbers.

If none of the Extended Geometry elements overlap, then an optimization can be applied. It is turned on by supplying "**OPT**" on the second line of the GEOM.INP file, after the title line. When optimization is turned on, then the next entry to the Extended Geometry module will start where the previous entry exited, i.e. the optimization assumes that the next (x, y, z) location is just a step away from the previous (x, y, z) location. It is up to user to check that no defined shapes are overlapped when using this optimization mode.

MARS must be provided with the volumes of all Extended Geometry zones. In the special case of no overlapping shapes *and* when optimization is turned on, then the Extended Geometry module will calculate the shape volumes and hand the results to MARS. In all other cases the user has to provide the volumes of shapes, either "by hand" or by running MARS in "volume MC" mode, described in Sections 3.1 and 4.2.2.

4.4.1 Example of Extended Geometry

Given here is an example of an Extended Geometry description. First is a simple MARS.INP file used in conjunction with the example. Following is a GEOM.INP containing a sampling of shapes. The GUI picture corresponding to this file is shown in Fig. 22.

```
Extended geometry test 10/22/02
/home/mokhov/restricted/mars15/dat
INDX 3=T
CTRL 0
NEVT 5
ENRG 8.  5=0.001
IPIB 1
SMIN 0.01 3.
ZSEC 160.
RSEC 80.  101=0
NMAT 6
MATR 'CU' 'FE' 'AIR' 'CONC' 'CH2' 'CH'
STOP


Exgeo test
sphere-0    3 0 0 0.  0.  10.  0.  2.
sphere-1    3 0 1 0.  0.  10.  2.  5.
sphere2a    3 0 2 0.  30.  15.  0.  4.  2
sphere2b    3 0 1 0.  30.  15.  4.  10.  3 !  ball-2b
sphere-3    3 0 4 0.  0.  60.  0.  15.  5
box-1       1 0 3 1.  -30.  15.  4.  4.  17.  2
box-2       1 0 5 -3.  -40.  45.  7.  12.  10.  5 !  box-2
cone-1      4 0 6 0.  0.  90.  2.  5.  6.  12.  20.  4
cone-2      4 1 3 0.  30.  90.  6.  12.  2.  5.  30.  6
cyl0        2 1 1 0.  30.  90.  0.  2.  30.  2 2 0
cone-4      4 1 4 0.  30.  90.  2.  6.  2.  2.  30.  2
cone-3      4 0 0 0.  0.  90.  0.  2.  0.  6.  20.
cyl-1       2 0 1 0.  0.  120.  0.  5.  40.  10 10 !  cyl-1
cyl-2a      2 0 2 0.  50.  120.  1.  5.  30.  10 2
cyl-2b      2 0 1 0.  50.  120.  0.  1.  30.  !  cyl-2b
cyl-3       2 0 3 0.  30.  130.  0.  5.  20.  4 5 !  cyl-3
cyl-4a      2 2 4 0.  -40.  125.  0.  5.  25.  5 2 !  cyl-4a
cyl-4b      2 3 4 0.  0.  0.  0.  5.  25.  5 2 !  cyl-4b
TR1         0.  0.  0.  -14.32 0.  0.  !  TRF-1
TR2         0.  0.  0.  -11.46 0.  0.  !  TRF-2
TR3         0.  -60.  115.  11.46 0.  0.  !  TRF-3
STOP
```

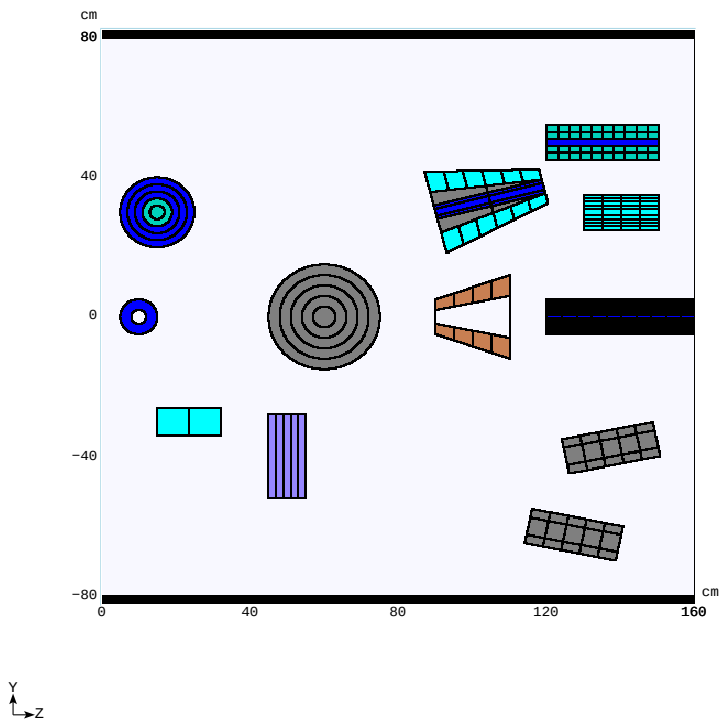


Figure 22: Plan View of the Extended geometry Example.

5 User Subroutines

The user subroutines are collected in the file **m1510.f**. The default version of this file, distributed as a part of the MARS code, contains dummy versions of each of the user subroutines. Some of the subroutines contain comments and commented-out sample code as reminders for their use, but the user should consult this manual for details on the full usage of the subroutines.

Table 8 lists the names of the user subroutines, and a brief description of their purpose.

The user is free to split file **m1510.f** up into other files, for example, combining all his customized subroutines into a single file, and storing all the unused dummy user subroutines in another file. The user can also create his own subroutines which are called from the user subroutines. The user can create his own set of common blocks to carry information among his own subroutines; none of his values or variables will be carried back into the main MARS code, however. With a few exceptions, the communication between MARS and the user subroutines is via subroutine arguments.

If the user rearranges subroutines into files other than **m1510.f**, then he must remember to also modify the default GNUmakefile appropriately, so the compiler-linker knows to pick up the local user files. This manual assumes the user knows how to compile and link code using gmake or make on unix or linux systems.

It is important to remember that MARS uses double precision in all calculations, and all real variables are declared double precision. The user will utilize MARS real variables delivered via subroutine arguments, and rather than copy them to local single precision variables, it is highly recommended that all user real variables be likewise declared double precision. This means that all real value assignments must be of the form `XVar=0.24D0` or `Parameter(YLimit = 435.76D0)`. The exception to this is when user-defined histograms are employed; real variables loaded into `HBOOK` must be single-precision.

Table 8: User subroutines.

Subroutine name	Description
MARSMAIN	Main program - contains names of the main I/O files
MARS15NN	”initialize-run-write output” steering subroutine, nn=release #
MIXTUR	define a compound material
BEG1	redefine the primary particle distribution
REG1	define user-encoded non-Standard zones
REG3	redefine the material content of Standard zones
FIELD	define magnetic fields
SUFI	read in magnetic field data
LEAK	tag/score particles which reach selected blackhole zones
ALIGN	discrete fictitious scattering between selected zones
SAGIT	fictitious scattering within a zone
RFCAVT	RF kick at the boundary of selected zones
EDGEUS	edge scattering
VFAN	define the volumes of all non-Standard or ovelapped zones
MHSETU	booking of user defined histograms
MFILL	filling of user defined histograms
HISTO_DB	a Block Data for parameters of user-defined histograms
WRTSUR	tag/score particles which cross selected user defined surfaces
TAGPR	particle origin tagging
TAGGING	place holder for user-defined tagging
KILLPTCL	kill particle under certain user-defined conditions
BLPROCESS	Block Data of switches to control exclusive generators
HISTDUMP	volume and surface histogram binning and DUMP regions
MARS2BML	MARS to Beam-line transformation
BML2MARS	Beam-line to MARS transformation
BLINIT	MMBLB initialization
BLGEOINIT	MMBLB element registration
TUNNELGEO	MMBLB independent tunnel description

5.1 Main Program MARSMAIN.F

The names of the main I/O files are here, and can be modified by the user.

5.2 Subroutine MARS1510

Subroutine M1510 is the main ”steering” routine, which directs the running mode of the executable. There are three possible running modes: normal, GUI interface, and stand-alone event-generator. Switching between these modes is controlled by the **CTRL** card in the input deck. The default mode is normal, where the primary particle interacts with the described geometry, secondary particles are tracked through the geometry, and various parameters are ”scored” and tabulated for output. The GUI interface, Section 12, is very useful for checking a complex encoded geometry, and for displaying results. No events are generated in this mode, but the user’s encoding of the modeled geometry, and zone and material assignment, is executed and displayed. The stand-alone event generator is useful for ???? .

The three modes can be seen in the 3-way `IF-ELSE` block within the subroutine. The first segment in the block, which calls `STTCL`, is the GUI interface mode. The second segment in the block, which calls `EVTGEN`, is the stand-alone generator mode. The third segment in the block, which calls `MARSON`, is the normal running mode. For normal and GUI modes, the user does not need to customize anything in routine MARS1510. The only items which might be modified, as needed, are the declared size of the `PAWC` common block, and the names of the input, output and histogram files. Some of the settings for the stand-alone generator mode are customized from this routine.

5.2.1 Stand-alone event generator

The stand-alone generator, as mentioned above, is useful for ??? To make use of this mode, the user must first set the card **`CTRL`** `2=IM` in the input deck, where `IM` is the material index of the target nucleus the event generator will use. The primary particle type and energy are specified in the usual way using the **`ENRG`** and **`IPIB`** cards in the input deck. For example, if the user wants to investigate the details of protons striking a graphite (carbon) nucleus, then the pertinent cards in the input deck might be

```
CTRL 2=4
NEVT 100
ENRG 250.
IPIB 1 3
NMAT 5
MATR 'AIR' 'AL' 'FE' 'C' 'CONC'
```

where **`CTRL`** `2=4` indicates that the target nucleus is the 4th material listed in the **`MATR`** card, which is `'C'` for carbon. The primary particle is a 250 GeV proton, and 100 events have been requested.

The user next sets variables in routine MARS1510 to control the generator details and the results which will be reported. The choices are given by comments within the routine: Here, the differentials `DY`, `DMT`, `X` and `E` are for rapidity, transverse mass, Feynman x, and energy, respectively. Selecting one of these determines the contents of the tables of data which are the result of the calculation.

5.3 Subroutine MIXTUR - Custom Materials

Most commonly used compounds for modeling accelerator and detector components are listed in Table 4. A compound, or mixture, not listed can be defined by the user via subroutine MIXTUR. First, the user must list his custom mixture(s) in the input deck, as one of the items in the `MATR` card. The four-letter name of the mixture(s) must start with `MIX` followed by numbers or letters: `MIX1`, `MIX2`, `MIX3`, ..., `MIX0`, `MIXA`,`MIXB`, etc. Each of these names is assigned an `I` index, according to their order in the list of materials.

The user must then know the chemical formula for each mixture. For example, if a target is made from a special alloy, the chemical formula of the alloy is required. From the chemical formula, the user obtains the information necessary for this subroutine: the number of elements in the compound `MIX`, where `MIX` is less than 20, the atomic masses `A(MIX)` of each element, the atomic numbers `Z(MIX)` or each element, and the relative fractions, `W(MIX)`, by weight, of each element in the compound. One can look up `A` and `Z` in the periodic table, but then must calculate the relative weight fractions, using $WMIX_k = (P_k A_k)/AMOL$, where P_k is the total number of k atoms in the compound, and $AMOL = \sum(P_k A_k)$.

The example here shows a case where the 1st three materials are built-in, and the fourth is SiO_2 with 5% of water H_2O by weight. There are thus 1 silicon atoms, 3 oxygen atoms, and 2 hydrogen atoms in the mixture. The input deck would contain a line

```
NMAT 5
```

```
          MATR 'C' 'FE' 'AL' 'MIX1' 'CONC'
```

The users customized MIXTUR subroutine is then :

```
      SUBROUTINE MIXTUR(I, MIX, AMIX, ZMIX, WMIX)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
      DIMENSION AMIX(*), ZMIX(*), WMIX(*)
C
      IF(I.EQ.4) THEN
C
C 1=Silicon, 1 atom; 2=Oxygen, 3 atoms; 3=Hydrogen, 2 atoms
      MIX=3
      ZMIX(1)=14.0D0
      ZMIX(2)=8.0D0
      ZMIX(3)=1.0D0
      AMIX(1)=28.0855D0
      AMIX(2)=15.9994D0
      AMIX(3)=1.00794D0
      AMOL1=AMIX(1) + 2.D0*AMIX(2)
      AMOL2=AMIX(2) + 2.D0*AMIX(3)
      AMOL=0.95D0*AMOL1 + 0.05D0*AMOL2
      WMIX(1)=0.95D0*AMIX(1)/AMOL
      WMIX(2)=(0.95D0*2.D0*AMIX(2) + 0.05D0*AMIX(2))/AMOL
      WMIX(3)=0.05D0*2.D0*AMIX(3)/AMOL
C
      END IF
      RETURN
      END
```

On entry to MIXTUR, there must be a test on the material index `I`. Exit the routine for all built-in materials, and use a conditional statement to isolate custom mixtures from each other if there are more than one. Remember that all MARS variables, such as those passed by subroutine arguments, are double precision.

5.4 Subroutine BEG1 - Primary Beam Description

Routine BEG1 is used to customize the primary particle beam. The subroutine arguments are the variables used to fully describe the trajectory and properties of a given primary particle: `JJ` is the particle type index given in Table 2; `W` is the relative weight given to the particle; `E` is the energy of the particle; `X,Y,Z` is the particle's starting location; `DX,DY,DZ` are the direction cosines; `TOFF` is ?; `INTA` is a flag to impose a point-like interaction ? ; `NREG1` is a Zone number where the source term is located ? . When the primary beam is described using Input Deck `IPIB` and `BEAM` cards, MARS uses those input deck parameters on beam spot size and shape to generate a particle trajectory within that specified envelope. The trajectory is described by the starting position X,Y,Z and the direction cosines DX,DY,DZ. These variables are passed to BEG1 and can be further modified to describe beam envelopes more complex than what can be specified only by the Input Deck cards.

For example, consider a beam which has a gaussian distribution of particles, but has been clipped in one view. The input deck cards are used to describe the gaussian properties. In routine BEG1 the user examines the X,Y position. If the X or Y are beyond the clipped edge, then the trajectory can be regenerated randomly in BEG1, rechecked, and accepted only when it fits within the users specification. Another example is where the beam is uniformly distributed in one view but gaussian distributed in the other view. The Input Deck cards can do one or the other distribution for both views together, but not separately. Again, the user would use the input deck to specify either the uniform or gaussian characteristics, and then modify those characteristics as needed in routine BEG1.

Question: if you set W=0.0 for primaries which don't fit your criteria, does that eliminate that particular primary particle, and force the generation of a new one?

Another way to use routine BEG1 is to read in a file which contains a list of primary beam particles. The user can create such a file in any custom format; it must simply contain the same parameters as the BEG1 subroutine arguments, or contain sufficient information to extract those parameters. The user in this case must also install the code to properly open, read, and close the file.

Such a file can be generated by MARS itself, using the LEAK subroutine. In this case, one uses MARS in two (or more) consecutive steps. The first step generates a beam into a geometry and creates the resulting showers. Within that geometry is a Zone with a specific Zone number which acts as a flag for subroutine LEAK - any particle crossing into that zone will have it's parameters written to a file. The parameters are exactly those which are the BEG1 subroutine arguments. The first step process ends, resulting in this file of particles which reached that particular Zone. A second MARS process is started, consisting of the geometry "downstream" of the first process's LEAK zone; the file of particles from step 1 is the input for routine BEG1 in step 2. This two or multi-step process is useful when modeling labyrinths or shielding penetrations: step 1 generates many particles, a few of which reach the entrance of a labyrinth (the LEAK zone) and are written to output; step 2 works only with the particles which enter the labyrinth, and models only the labyrinth portion of the geometry. In this case different energy thresholds can be applied in steps 1 and 2 in order to optimize execution time, for example to concentrate only on thermal neutrons in the labyrinth. The multi-step process is also useful when investigating the effects of beam loss distributions: step 1 models the actual beam loss and records the particles which are created in that interaction shower; step 2 tracks only the shower to investigate it's effects on the surrounding area. Another example if the LEAK - BEG1 multi-step process is to separate the primary event generator from the processes which occur in the rest of the geometry: model a primary beam hitting a target, use DPMJET or ISAJET to model what happens in the target, and record, via LEAK, everything which exits the target; input the resulting particle list into step 2, which describes the geometry downstream of the target. In this way one can compare different production models at the target separately from the rest of the geometry.

5.5 Subroutines REG1 - non-Standard Zone Description

REG1 is the interface between MARS and the users's encoding of a complex geometry. MARS hands to the routine, via the subroutine arguments, the current location X,Y,Z of some particle; the user's job is to determine in REG1 what particular volume that X,Y,Z belongs to, and what the corresponding user-assigned non-Standard zone number `N` and material index `IM` for that volume are, and return those numbers to MARS. Subroutine VFAN is used to assign a volume to each of the user-assigned zones at program start-up. Recall, from Section 3.1, that for MARS to determine the flux of particles through any particular zone, it must have a value for the volume of the zone. **REG1 can also be used in the extended geometry mode to define subregion numbers `NIM` inside a given zone, with zone numbers and material indecies are determined automatically. These `NIM` can be in particular used in subroutines RFCAVT and ALIGN as a control switch.**

The user can encode as complex a geometry as necessary, and can do so in his own set of subroutines which get called from REG1. The user has as many as 100,000 non-Standard zones at his disposal; each declared zone gets the full complement of results accumulated, and these are tabulated in the `MTUPLE.NON` output file. Therefore, the boundaries of the user's non-Standard zones will correspond to various regions of interest where specific results are desired. For example, two separate zones of soil on either side of a beam dump can be set up, so that one can compare the star densities in the two locations. Or, set up several zones within an irregularly shaped target holder assembly, so that energy and heat deposition patterns resulting from the beam-target interaction can be studied. There are of course a few requirements which must be met

for the proper use of the subroutine. The non-Standard zones must be enclosed within, or have boundaries equal to, the outermost Standard zones. Each zone, of any type, can hold one and only one material. Each zone must have a value for it's volume. The initializations concerning the material content and volumes of the non-Standard zones are performed on the first call to routine REG1.

A simple example will serve to illustrate the non-Standard zone initialization process, as well as the geometry encoding process. It is strongly suggested that the user create a conceptual sketch of his geometry before starting to encode it, filling in the zone boundary locations, materials, zone numbers, and zone volumes, as this will assist in the coding process. Once a geometry is encoded, the user is advised to utilize the MARS GUI visual interface to check that the zone boundaries, and the zone number and material assignments, match the conceptual sketch. Figure 23 shows a sketch of a simple beam dump, made from steel and concrete blocks. These rectagular shapes become non-Standard zones. The dump is surrounded by soil, and the outer edge of the soil is modeled as a cylinder; this can therefore be a Standard zone specified by cards in the input deck (this cylinder is not shown in the sketch).

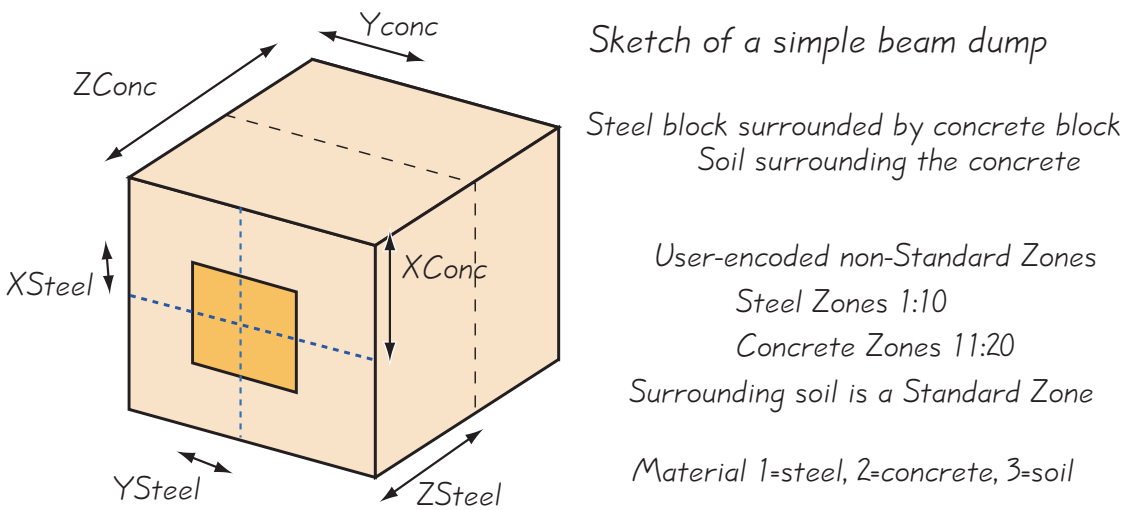


Figure 23: A user's sketch of a simple beam dump, with dimension variables, and with preliminary non-Standard zone assignments.

The following section of code is the array and variable declaration section from subroutine `reg1` corresponding to this example of a simple beam dump. The user has decided that material index 1 is steel, material index 2 is concrete, and material index 3 is soil. Consequently, the material cards in the input deck are set to

```
NMAT 3
MATR 'FE' 'CONC' 'SOIL'
```

The user has set the maximum number of non-Standard zones to 20. The first 10 non-Standard zones will be used for steel, and the second 10 for concrete.

```
      SUBROUTINE REG1(X, Y, Z, N, NIM)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C
      INCLUDE 'blreg1.inc'
      INCLUDE 'tally1.inc'
C
      SAVE NENTER
```

```
      DATA NENTER/0/
C
C ===Put actual max local non-standard zone number here !!!
      PARAMETER (M\_MAX = 20)
      PARAMETER (M\_MAX1 = M\_MAX + 1)     !+++Don't touch !!!
C
      CHARACTER*8 VNAME, VNAMEBUF
      DATA VNAMEBUF/'        '/
C
      DIMENSION IMUN(1:M\_MAX1)   ! local array for material indicies
      DATA INCREM/1/
      DATA IMUN/ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
     >           2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
     >           0/
```

The local array `IMUN` is declared and sized to `M_MAX + 1`. The contents of `IMUN` are the materials assigned to each of the Mth non-Standard zones, with the value being the index of the material, given by it's order in the input deck list. In other words, `IMUN(M)` is the material index for the Mth zone, where $M = 1$: `M_MAX` - a local zone number. The actual zone number, in the MARS program array space, depends on how many Standard and Extended Geometry zones have also been declared. The following section of subroutine `REG1` initialization code demonstrates this.

```
      IF(NENTER.EQ.0) THEN
         CALL REG3
         NCELMX = NFZPEX + M_MAX
         NENTER = 1
         IF(M_MAX .EQ. 0) INCREM=-1
         IF(M_MAX .GT. 0) THEN
            INUG = 1
            WRITE(*,*)'There are non-standard zones M_MAX= ',M_MAX
            DO L = 1, M_MAX, INCREM
               MATIND(NFZPEX + L) = IMUN(L)
               VOLNM (NFZPEX + L) = VNAMEBUF
            END DO
            NVTEST = 1
            CALL VFAN (NVTEST,V)
         ELSE
            INUG = 0
            WRITE(*,*)'There are no non-standard zones in this run !'
            RETURN
         END IF
      END IF
```

The non-Standard zone initialization process is one of the few places where the user directly sets variables held in a MARS program common block. Variable `NCELMX`, and others, are passed between routine REG1 and the rest of MARS via `blreg1.inc`, which contains various MARS parameters and common blocks. The line `NCELMX = NFZPEX + M_MAX` is how MARS is informed of the number of declared non-Standard zones. The total number of all zones, Standard, Extended and non-Standard, is held in variable `NCELMX`. All of the arrays which hold accumulated MARS results are one-dimensional and indexed by `NCELMX`. The one-dimensional arrays are logically divided into three blocks, holding first the results for Standard zones, then Extended, then non-Standard. The boundaries of these three blocks are defined by additional variables which are the total numbers of zones of each type. The total number of Standard zones is held in variable `NFZP`; the total number of Extended zones is in `NEXG2`; the total of Standard and Extended zones together is given by variable `NFZPEX`. MARS calculates `NFZP`, `NEXG2`, and `NFZPEX` during its startup initialization, based upon the declarations made in the MARS.INP and GEOM.INP input decks. On the first call to routine REG1, the user declares the total number of all zones, and the number of non-Standard zones, by setting `NCELMX`.

Notice that the local IMUN array, where materials are assigned to non-Standard zones, is copied to MARS array MATIND, offset in that array's space by NFZPEX. Array MATIND is what MARS uses to determine the material content of each zone, not IMUN. However, it is easiest for the user to think of his own non-Standard zone assignments as being numbered 1 : M_MAX, and fill his own local version of the materials array, since the absolute zone number which MARS uses can change if the user changes the number of Standard or Extended zones in his model from one execution of the program to another. All that MARS requires is that the user's version of the materials array be transfered to the MATIND array, with the correct offset, at some point in the REG1 initialization process.

Before continuing with the zone initialization process description, jump ahead to the geometry definition code within REG1 for the simple beam dump example. The user makes a list of parameters to hold the dimension values of his model (remember the basic length unit is centimeters, and all variables double precession) -

```
      Parameter (ZConc = 400.0D0)
      Parameter (ZSteel = 150.0D0)
      Parameter (ZSteel_middle = 75.0D0)
      Parameter (XSteel = 30.0D0)
      Parameter (YSteel = 35.0D0)
      Parameter (XConc = 100.0D0)
      Parameter (YConc = 120.0D0)
```

The encoding of the simple beam dump is then -

```
      M=0
      AX = Abs(X)
      AY = Abs(Y)
C
C Return immediately if we are outside the User non-Standard geometry
      If (AX.ge.XConc .or. AY.ge.YConc .or. Z.ge.ZConc) Return
C
C Is the current position in the steel core?
      If (AX.lt.XSteel .and. AY.lt.YSteel .and. Z.lt.ZSteel) Then
         If (Z.ge.0.0D0 .and. Z.lt.ZSteel_middle) Then
            M = 1
            Go to 200
         Else If (Z.ge.ASteel_middle .and. Z.lt.ZSteel) Then
            M = 2
            Go to 200
         End If
      End If
C
C Is the current position in the concrete surrounding the steel?
      If (Z .lt. ZSteel) Then
         If (AX.lt.XConc .and. (Ay.ge.YSteel .and. AY.lt.YConc)) Then
            M = 11
            Go to 200
         Else If ((AX.ge.XSteel .and. AX.lt.XConc) .and. AY.lt.YConc) Then
            M = 11
            Go to 200
         End If
C Is the current position in the concrete downstream of the steel?
      Else If (Z.ge.ZSteel .and. Z.lt.ZConc) Then
         If (AX.lt.XConc .and. AY.lt.YConc) Then
            M = 13
            Go to 200
         End If
      End If
C
```

```
C If all User-zone-area X,Y,Z are covered by the above statements, then the
C code should never get to this point. If M=0 here, then there is a hole in the
C logic. If M>0 and gets to here, then there is a missing Go To 200 statement
C Print a warning
      write (*,*) ' **** Logical Error in Reg1 **** ',M
      Return
C
C Set and Return N, the Mars Zone number corresponding to input X,Y,Z
 200  Continue
      IF (M .GT. 0) THEN
         N = NFZPEX + M
      ELSE IF (M .LT. 0) THEN       ! Non-standard blackhole
         N = M
      END IF
      RETURN
      END
```

Although this code is simple, there are several necessary steps which apply to any use of this subroutine. When REG1 is called, MARS passes to it the position X, Y, Z of the current particle. There is a local variable M which is set to 0 on entry to REG1; M is set to a non-zero value somewhere in the code's logic, and it's value is between 1 and M_MAX. Just before exiting REG1, the statement N = NFZPEX + M sets and returns to MARS the absolute zone number N which corresponds to the input X, Y, Z.

There are other steps in this example which, while not necessary, are highly recommended. At the top, there is a statement (or series of statements) which asks whether the current X, Y, Z is within the regions where non-Standard zones are defined; if not, then the routine is exited. Up to 80% of the program's running time is spent in geometry modules, including REG1. It is far more efficient to determine immediately whether X, Y, Z is outside the overall non-Standard zone region, than to let all possible X, Y, Z locations fall through the code. In the case where X, Y, Z is outside the non-Standard zone region, the input absolute zone number N must be returned to MARS **unchanged**. Another recommendation is to construct the logic of the code so that satisfied conditions 'jump out', and errors 'fall-through'. In this example, once a condition is satisfied, and a zone number assigned, the code 'jumps out' of the conditional statements to the end, where the absolute zone number N is assigned. If none of the conditional statements are satisfied, the code 'falls through' to the end of this list of statements. If the conditional statements are coded correctly, then the code should never reach the 'fall through' point. One can use this fact to catch errors. While this tactic may seem unnecessary for a simple example, the logic to encode more intricate geometries can rapidly become very complex, and difficult to debug.

Now that the encoding and zone assignment of the simple beam dump is defined, return to the zone initialization process. Every non-Standard zone where the user wants results to be tabulated must have it's volume declared to MARS. The default method for handing volume values to MARS is to use subroutine VFAN. As shown in the above code section, the default version of REG1 is set up to call VFAN in it's initialization section, and similarly, VFAN contains an initialization section of it's own, executed this first time the routine is called. For the simple beam dump example, subroutine VFAN is as follows -

```
      SUBROUTINE VFAN (N, V)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C
      INCLUDE 'blreg1.inc'
      INCLUDE 'tally1.inc'
C
      DATA NENTER/0/
      SAVE NENTER
C
      Parameter (ZConc = 400.0D0)
      Parameter (ZSteel = 150.0D0)
      Parameter (ZSteel_middle = 75.0D0)
```

```
      Parameter (XSteel = 30.0D0)
      Parameter (YSteel = 35.0D0)
      Parameter (XConc = 100.0D0)
      Parameter (YConc = 120.0D0)
C
C- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      N1=NFZPEX   ! NUMBER OF STANDARD+EXTENDED REGIONS
C
C one-time initialization section
      IF(NENTER.EQ.0) THEN
         NENTER=1
C
C Steel core - both sections the same volume
         VOLUME (N1 + 1) = (2.0D0*XSteel)*(2.0D0*YSteel)*ZSteel_middle
         VOLNM (N1 + 1) = 'UpstCore'
         VOLUME (N1 + 2) = Volume (N1 + 1)
         VOLNM (N1 + 2) = 'DnstCore'
C
C The concrete around the sides of the steel core
         A1 = 2.0D0*((2.0D0*YConc)*(XConc-XSteel))      ! Top and Bottom area
         A2 = 2.0D0*((YConc-YSteel)*(2.0D0*XSteel))     ! two sides area
         VOLUME (N1 + 11) = (A1 + A2) * ZSteel
         VOLNM (N1 + 11) = 'Conc Zn1'
C
C The concrete in back of the steel core
         VOLUME (N1 + 13) = (2.0D0*YConc) * (2.0D0*XConc) * (ZConc - ZSteel)
         VOLNM (N1 + 13) = 'Conc Zn2'
C
         Return
C
C non-initialization call - return the Volume for Zone N
      Else
         V = VOLUME (N)
      End If
C
      RETURN
      END
```

The same parameter list, used in `REG1`, to describe the dimensions of the beam dump, is also included here to calculate the volumes of the beam dump zones; all volumes are cm^3. Zone volumes are placed directly into the MARS array `VOLUME`; these non-Standard zone volumes are written with an array offset of `NFZPEX`, just as the material content of each non-Standard zone was entered into array `MATIND`. At program startup, MARS initializes all elements of array `VOLUME` to zero. MARS then fills the Standard and Extended zone sections of array `VOLUME` based on settings in the `MARS.INP` and `GEOM.INP` input files. Finally, the user's code is called to fill the non-Standard zone portion.

Also filled is character array `VOLNM`, which holds an 8-character string name for each zone. This array is initialized to hold blanks in the initialization section of subroutine `REG1`. In subroutine `VFAN` this array gets filled with zone names for each zone which gets a volume assigned. These zone names are used in the non-Standard zone output file `MTUPLE.NON`, and simply make it easier to locate the line holding the results for a specific zone.

The non-Standard output file `MTUPLE.NON` holds a table of results for all declared non-Standard zones. Only zones which have been properly declared will have an entry in the table of results, and the trigger for what MARS considers proper declaration is a non-zero value for the volume of the zone. So, even if there is a material indicated for the zone entered into the `MATIND` array, even if code exists within `REG1` (or called from it), which describes the boundaries of the zone, and this code returns a non-Standard zone number

assignment, no results for that zone will be reported in the output unless the zone has a non-zero volume entered into array VOLUME. And, for the results to be correct, the volume value cannot be arbitrary - it must be the correct volume, corresponding to the encoded boundaries of the zone. As discussed in section 3, that is is particle path length which is accumulated as particles are tracked through zones, and to obtain the particle flux in the zone, path length is divided by the zone volume. An incorrect value for the zone volume will give incorrect results for the flux, and for all other results derived from particle fluxes.

The same local zone index is used when defining volumes.

can be greater than the total number of non-Standard zones actually used, because the declared array space does not need to be close-packed.

M_MAX is set by the user, and is the size of the MARS array space being reserved for the user's non-Standard zones. The user can define M_MAX in a DATA statement, as done in the distributed REG1, or via his own parameter list or common block.

The user must be careful to coordinate the physical outer boundaries of his non-Standard volume with the Standard zone(s) and/or Extended geometry zones defined via the input decks. Non-Standard zones must all fit inside Standard zones, or be aligned along the same boundaries; non-Standard zones cannot extend beyond the outermost defined Standard zone. This is because, as stated above, the outermost extent of the Standard zones defines for MARS the outermost extent of the entire model. If the entire model is defined by the user from REG1, then one must still have at least 1 Standard zone, and the maximum geometrical values in the ZSEC and RSEC cards in the input deck must correspond to the maximum geometrical limits set by the user within REG1. Up to 80% of the cpu-time is spent in geometry modules, including REG1, when propagating particles. If the user's non-Standard zones fit within larger Standard zones, then the user should test the input X,Y,Z on entry to REG1, and if the current location is outside the overall non-Standard volume, RETURN to MARS, with the input zone number N unchanged. This is far more efficient than letting all possible X,Y,Z locations fall through the user's code.

If one wants to study cascades in a very complex geometry not embraced by the above options, user subroutines REG1 and REG2 must be provided. MARS, version 13(95), allows the user to place geometrical objects of almost any complexity inside the pre-defined $standard$ (r-z-ϕ) or $extended$ geometry. By convention the total number of $standard$ regions NFZP must be NFZP$\geq$1. With the help of his/her own subroutines REG1(X,Y,Z,N,NIM) and REG2(N,IM,MAG) the user can describe arbitrary physical regions numbered from the NMIN$\leq$N$\leq$NMAX interval. Here NMAX=10000; NMIN=NFZP+1 for the Standard geometry sector and NMIN=NEXGM+NVOLUM for the $extended$ geometry sector. Each region can be divided into any number of arbitrary subregions with index NIM with a default value NIM=0. This feature provides the possibility of distinguishing geometrical zones without scoring results there. The user can successfuly substitute this with the HBOOK histogramming (see MAIN).

For each call, a subroutine REG1(X,Y,Z,N,NIM) finds the position of the given point (X,Y,Z) in the system: it determines the corresponding physical region number N (each with its own material index) and, if one wishes, the subregion number NIM. The last parameter can be used in a subroutine FIELD to determine the type of magnetic field (uniform, dipole, quadrupole etc.) in the region N. Default: MAG=0 (no magnetic field in the region).

By convention the region outside of the global volume has a number N=0 and properties of the *black hole*. In some applications it is useful to tag and to score the leakage out of the non-standard regions into the *black holes* labeled with N≤ -1 to use these negative tags in a user routine LEAK. The user must pay a special attention to careful programming of the subroutine REG1, because the geometrical modules consume usually about 80 % of the CPU time, as is typical of cascade Monte Carlo programs.

A simple example of subroutine REG1 is:

```
      SUBROUTINE REG1(X,Y,Z,N,NIM)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C     NON-STANDARD GEOMETRY MODULE
C     FINDS THE PLACE OF GIVEN POINT IN THE SYSTEM
C     INPUT: X, Y, Z
C     OUTPUT:
C     N - PHYSICAL REGION NUMBER, NMIN≤N≤NMAX
C     NMIN=NFZP+1 (STANDARD)
C     OR
C     NMIN=NEXGM+NVOLUM (EXTENDED)
C     NMAX=10000
C     N≤-1 DEFINES NUMBERED LEAKAGE OUT OF THE SYSTEM
C     IN NON-STANDARD SECTOR
C     NIM -GEOMETRICAL SUBREGION NUMBER, 0≤NIM
C
C     REVISION: 28-FEB-1995
C
      R=SQRT(X*X+Y*Y)
      IF(R.LT.10.)  RETURN
      M=500
      IF(R.LT.150.)  THEN
      IF(Z.LT.490.)  THEN
C     STEEL SHIELDING:
      N=M+1
C     TUNNEL:
      IF(R.GT.30.)  N=M+2
      END IF
      ELSE
C     CONCRETE SHELL:
      N=M+3
      END IF
      RETURN
      END
```

5.6 Subroutine VFAN - Specifying Zone Volumes

The code calculates volumes in arbitrary (overlapped) regions in a short Monte Carlo session once one puts `CTRL 3=1` in the `MARS.INP` file. As described in Section-4, these regions are contained in a cylinder defined by the the **RZVL** card. The results go into an output file `VOLMC.NON` in the format ready for inclusion into the user routine `VFAN`. Statistical errors for the calculated volumes are also included to see if a number of events `NVTRIAL` in this session is enough.

Here is an example of a `VOLMC.NON` to be included in `VFAN`:

```
 VOLUME(      7) =   0.8712684D+02    ! +/-   1.39%,  M=       1
 VOLUME(      8) =   0.8915421D+02    ! +/-   1.37%,  M=       2
 VOLUME(      9) =   0.8840023D+02    ! +/-   1.38%,  M=       3
 VOLUME(     10) =   0.8766300D+02    ! +/-   1.38%,  M=       4
 VOLUME(     11) =   0.8761274D+02    ! +/-   1.38%,  M=       5
 VOLUME(     12) =   0.8754572D+02    ! +/-   1.38%,  M=       6
 VOLUME(     13) =   0.8628908D+02    ! +/-   1.39%,  M=       7
 VOLUME(     14) =   0.8806513D+02    ! +/-   1.38%,  M=       8
 VOLUME(     15) =   0.8917097D+02    ! +/-   1.37%,  M=       9
 VOLUME(     16) =   0.8922123D+02    ! +/-   1.37%,  M=      10
 VOLUME(     17) =   0.8677498D+02    ! +/-   1.39%,  M=      11
 VOLUME(     18) =   0.8863480D+02    ! +/-   1.37%,  M=      12
```

```
VOLUME(     19) =   0.8769651D+02    ! +/-   1.38%,  M=      13
VOLUME(     20) =   0.8774678D+02    ! +/-   1.38%,  M=      14
VOLUME(     21) =   0.8752896D+02    ! +/-   1.38%,  M=      15
VOLUME(     22) =   0.8853427D+02    ! +/-   1.38%,  M=      16
VOLUME(     23) =   0.9057840D+02    ! +/-   1.36%,  M=      17
VOLUME(     24) =   0.8818241D+02    ! +/-   1.38%,  M=      18
VOLUME(     25) =   0.8972389D+02    ! +/-   1.37%,  M=      19
VOLUME(     26) =   0.8778029D+02    ! +/-   1.38%,  M=      20
VOLUME(     27) =   0.4521213D+03    ! +/-   0.61%,  M=      21
VOLUME(     28) =   0.4606161D+03    ! +/-   0.60%,  M=      22
VOLUME(     29) =   0.4611858D+03    ! +/-   0.60%,  M=      23
VOLUME(     30) =   0.4583374D+03    ! +/-   0.60%,  M=      24
VOLUME(     31) =   0.4584715D+03    ! +/-   0.60%,  M=      25
VOLUME(     32) =   0.4570305D+03    ! +/-   0.61%,  M=      26
VOLUME(     33) =   0.4621576D+03    ! +/-   0.60%,  M=      27
VOLUME(     34) =   0.4569300D+03    ! +/-   0.61%,  M=      28
VOLUME(     35) =   0.4548021D+03    ! +/-   0.61%,  M=      29
VOLUME(     36) =   0.4573154D+03    ! +/-   0.61%,  M=      30
VOLUME(     37) =   0.4591919D+03    ! +/-   0.60%,  M=      31
VOLUME(     38) =   0.4571311D+03    ! +/-   0.61%,  M=      32
VOLUME(     39) =   0.4592087D+03    ! +/-   0.60%,  M=      33
VOLUME(     40) =   0.4649390D+03    ! +/-   0.60%,  M=      34
VOLUME(     41) =   0.4573824D+03    ! +/-   0.61%,  M=      35
VOLUME(     42) =   0.4620738D+03    ! +/-   0.60%,  M=      36
VOLUME(     43) =   0.4534784D+03    ! +/-   0.61%,  M=      37
VOLUME(     44) =   0.4571981D+03    ! +/-   0.61%,  M=      38
VOLUME(     45) =   0.4580358D+03    ! +/-   0.60%,  M=      39
VOLUME(     46) =   0.4558744D+03    ! +/-   0.61%,  M=      40
VOLUME(     47) =   0.1557392D+03    ! +/-   1.04%,  M=      41
VOLUME(     48) =   0.1562921D+03    ! +/-   1.04%,  M=      42
VOLUME(     49) =   0.1551360D+03    ! +/-   1.04%,  M=      43
VOLUME(     50) =   0.1569288D+03    ! +/-   1.03%,  M=      44
VOLUME(     51) =   0.1539464D+03    ! +/-   1.04%,  M=      45
VOLUME(     52) =   0.1565435D+03    ! +/-   1.03%,  M=      46
VOLUME(     53) =   0.1562586D+03    ! +/-   1.04%,  M=      47
VOLUME(     54) =   0.1573310D+03    ! +/-   1.03%,  M=      48
VOLUME(     55) =   0.1549015D+03    ! +/-   1.04%,  M=      49
VOLUME(     56) =   0.1559905D+03    ! +/-   1.04%,  M=      50
VOLUME(     57) =   0.5464025D+03    ! +/-   0.55%,  M=      51
VOLUME(     58) =   0.5460674D+03    ! +/-   0.55%,  M=      52
VOLUME(     59) =   0.5474246D+03    ! +/-   0.55%,  M=      53
VOLUME(     60) =   0.5488823D+03    ! +/-   0.55%,  M=      54
```

In a $standard$ geometry sector the volumes of the regions needed, for example, to compute energy deposition density in GeV/g, are calculated at the final stage in the SERVN subroutine. The array of volumes `VV` of the $non-standard$ regions `M` should be provided by the user in a subroutine `VFAN(N,V)`. The same subroutine can be used in specific activity and residual dose rate calculations at `IND(13)=T`. The VFAN routine can look as:

```
      SUBROUTINE VFAN(N,V)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C     FIND VOLUME V(N),CM**3 OF REGION N OF THE NON-STANDARD GEOMETRY
C     VOLUME(N) ARE DEFINED IN SERV FOR N <= NFZP
      INCLUDE 'tally1.inc'
      DATA NENTER/0/
      SAVE NENTER
      N1=NFZPEX (NUMBER OF STANDARD+EXTENDED REGIONS)
      IF(NENTER.EQ.0) THEN
      NENTER=1
C     PUT HERE THE REAL VOLUMES (CM**3) FOR ALL THE NEEDED ===
C     NON-STANDARD REGIONS <= M_MAX, I.E. REDEFINE ANY OF THE
C     PRE-DEFINED VOLUME(L)=0.D0
C     For example:
C     VOLUME(N1+2) =PI*RCOL*RCOL*180.D0
C     VOLUME(N1+3) =PI*(400.D0-RCOL*RCOL)*60.D0
C     VOLUME(N1+365)=VOLUME(N1+3)
      END IF
      V=VOLUME(N)
      RETURN
      END
```

5.7 Subroutine REG3 - Standard Zone Re-numbering

In some cases it is convenient to re-define a few material and/or magnetic indices assigned to the $standard$ or $extended$ regions at the initilization stage. This can be easily done in a user routine REG3, e.g.:

```
      SUBROUTINE REG3
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C     RE-DEFINES IM AND MAG FOR STANDARD SECTOR
      IF(N.EQ.137) IM=14
      IF(N.EQ.2503) IM=2
      RETURN
      END
```

5.8 Subroutines FIELD & SUFI - Description of Magnetic & Electric Fields

To describe the magnetic field components (`BX,BY,BZ`) in the regions with parameter `MAG`$\neq$`0` the user puts `IND(4)=T` and provides a subroutine FIELD. The same routine is used to describe an electrical field. One can use a corresponding map or analytical expressions to find the field components in the point (`X,Y,Z`). Parameter `MAG` defined in REG2 can speed up the search. It indicates the type of the field in the region. The unit for magnetic field is $Tesla$. A 2-D or 3-D field map is read in a user routine SUFI at the initilization stage and transfered to the routine FIELD via appropriate `COMMON` block.

An example of the FIELD subroutine is:

```
      SUBROUTINE FIELD(N,X,Y,Z,BX,BY,BZ,BBB)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C     FINDS COMPONENTS OF MAGNETIC FIELD
C     INPUT: MAG -MAGNETIC INDEX AT GIVEN POINT
C     X,Y,Z - POINT'S COORDINATES (OPTIONAL)
C     FIELD MAPS
C     QUADS GRADIENTS IN T/CM
C     G≥0 FOR FOC QUADS, G≤0 FOR DEFOC QUADS
C     OUTPUT: BX,BY,BZ,BBB IN TESLA
C     REVISION: 16-DEC-1994
C
      COMMON/BLINT2/JJ,KK,MAG
      DATA RAQ,BQUA,G1/8.,2.,0.53/
      BX=0.
      BY=0.
      BZ=0.
      IF(Z.LT.622.)  THEN
      BX=5.
      IF(MAG.EQ.2) BX=-5.
      ELSE
      CALL QUAD(X,Y,R,RAQ,G1,BX,BY)
      CALL DIPOLE(X,Y,R,RAQ,BQUA,BX1,BY1)
      BX=BX+BX1
      BY=BY+BY1
      END IF
      RETURN
      END
```

5.9 Subroutine RFCAVT - Description of Accelerating Components

```
      SUBROUTINE RFCAVT(JJ,VECT,VOUT,E,TOFF,NIB,NIM,IFLAG)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C     RF CAVITY KICK:
C     DISCRETE AT BOUNDARIES BETWEEN REGIONS WITH SPECIAL NUMBERS NIB AND NIM
C     NIB - PRIOR CROSSING, NIM - AFTER CROSSING
C     INPUT : JJ, VECT (X,Y,Z,DCX,DCY,DCZ,P), E, TOFF, NIB, NIM
C     OUTPUT: VOUT (X,Y,Z,DCX,DCY,DCZ,P), E, IFLAG
C     VECT(1)-VECT(7) AND E ARE RE-DEFINED TO VOUT(1)-VOUT(7) AND E
C     IFLAG=1 MUST BE RAISED
C     IFLAG=2 - E_new < 0
      INCLUDE 'cmasnsg.inc'
      DIMENSION VECT(7),VOUT(7)
C     DO L=1,7
C     VOUT(L)=VECT(L)
C     END DO
      RETURN
      END
```

5.10 Fictitious Scattering (ALIGN, SAGIT)

In some applications components of the considered system can be turned or shifted with respect to each other. Say, the arc of any circular accelerator is built of the magnets turned by a fixed angle. MARS, version

13(95), allows an elegant way to handle such systems. The user describes the geometry as the straight along the z-axis. Then, with the help of a routine ALIGN, he/she creates the angular or space kicks at the required boundaries in the directions *opposite* to the real ones. One can easily see that the resulting coordinates and angles of any particle are identical to those in the real *bent* geometry. It is convenient to use spare NIM parameters to control such "a fictitious scattering" at the required boundaries with NIB≠NIM as that parameter in a previous region. Of course, any coordinate can be used to locate the needed boundary.

The following example shows use of the ALIGN routine for a single kick:

```
      SUBROUTINE ALIGN(VECT,VOUT,NIB,NIM,IFLAG)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C     FICTITIOUS SCATTERING:
C     DISCRETE AT BOUNDARIES BETWEEN REGIONS WITH SPECIAL NUMBERS NIB AND NIM
C     NIB - PRIOR CROSSING, NIM - AFTER CROSSING
C     VECT,VOUT: X,Y,Z,DCX,DCY,DCZ,P
C     VECT(1)-VECT(6) CAN BE RE-DEFINED TO VOUT(1)-VOUT(6),
C     NORMALLY AT IND(4)=T
C     IF SO, VOUT(1)-VOUT(6) MUST BE FILLED AND IFLAG=1 MUST BE RAISED
      DIMENSION VECT(7),VOUT(7)
C     DO L=1,6
C     VOUT(L)=VECT(L)
C     END DO
      RETURN
      END


      SUBROUTINE SAGIT(CHARGE,STEP,VECT,VOUT,NREG,IFLAG)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C     FICTITIOUS SCATTERING:
C     ON STEP IN REGION NUMBER NREG
C     VECT,VOUT: X,Y,Z,DCX,DCY,DCZ,P
C     VECT(1)-VECT(6) CAN BE RE-DEFINED TO VOUT(1)-VOUT(6),
C     NORMALLY AT IND(4)=T
C     IF SO, VOUT(1)-VOUT(6) MUST BE FILLED AND IFLAG=1 MUST BE RAISED
      DIMENSION VECT(7),VOUT(7)
C     DO L=1,6
C     VOUT(L)=VECT(L)
C     END DO
      RETURN
      END
```

The same method is used in MARS to handle objects with saggita, i.e. continuously bent. "Fictitious scattering" defined in a user subroutine SAGIT as an *anti-kick* at every step along charged and neutral particle trajectories, simplifies the geometry description and makes a precise correspondence to the real bent objects.

5.11 Edge Scattering (EDGEUS)

If the "edge scattering problem" is considered with IND(9)=T and the geometry includes $non-standard$ insertions defined with REG1 and REG2, the user should supply a subroutine EDGEUS(X,Y,Z,DX,DY,DZ,U,V). It finds the distance U from the given trajectory point (X,Y,Z) to a non-standard surface, and the projection V of the direction vector (DX,DY,DZ) to the normal to the surface which passes in its positive direction through the point (X,Y,Z).

This example shows use of the EDGEUS routine for the surface Y=YPL>0 and for the particle with

Y>YPL and DY<0:

```
      SUBROUTINE EDGEUS(X,Y,Z,DX,DY,DZ,U,V)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C     EDGE-SCATTERING PROBLEM
      U=Y-YPLANE
      V=DY
      RETURN
      END
```

5.12 Subroutine LEAK - Creating custom source terms

IM= -1 and region number $N \leq 0$ to $blackholes$, which are either all material outside of the simulated volume ($N = 0$) or any region inside marked with $N = -1, -2, -3, ...$ where entering particles are killed. This feature is used for special scoring, or for forming a source term for the cosequent runs, both in subroutine LEAK.

A user subroutine LEAK(N,K,JJ,W,E,X,Y,Z,DCX,DCY,DCZ,TOFF) handles particles which escape from the system (N=0) or from the $non-standard$ regions tagged with N≤ -1. Parameter K is the tree vertex level (generation number) for the hadronic part of the calculated cascade. If IND(14)=T, then K has the same meaning as the above if given particle was generated in hA vertex at E>0.0145 GeV, otherwise it is forced to be K=$-$205, if the leaked neutron was produced in sequent interactions below 0.0145 GeV. JJ is particle type from Table 2. If K=$-$205, the particle type is neutron by default and JJ is its energy group number from Table 1. W is leaked particle statistical weight, E is its kinetic energy, (X,Y,Z) and (DCX,DCY,DCZ) - its coordinates and direction cosines, respectively.

The routine LEAK used in an example in Section 5.21 to collect generated antiprotons in the file PBAR.OUT looks like:

```
      SUBROUTINE LEAK(N,K,JJ,W,E,X,Y,Z,DCX,DCY,DCZ,TOFF)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C     PARTICLE LEAKAGE SPECIAL SCORING
C     IF(K=-205):  JJ=NGROUP, E=0.0005*(EN(JJ)+EN(JJ+1))
C     FOR L.E.NEUTRONS (E≤0.0145 GEV)
C     REVISION: 02-NOV-1994
      DATA POPT,DP0,DZ0/8.9,0.022,0.99/
C
      IF(JJ.NE.12) RETURN
      IF(DZ.LE.DZ0) RETURN
      P=SQRT(E*(E+1.87656))
      DP=ABS((P-POPT)/POPT)
      IF(DP.GT.DP0) RETURN
      WRITE(9)E,W,X,Y,Z,DCX,DCY,DCZ,K
      RETURN
      END
```

5.13 Subroutines MHSETU, MFILL, HISTODB - User Histograms

```
      SUBROUTINE MHSETU
C     SET UP HISTOGRAM ARRAYS
C     HISTOGRAM ENTRY FOR USER-DEFINED HISTOGRAMMING
C     HISTOGRAM ID AVAILABLE: 700<ID<999
C     HISTOGRAM TYPE:
C     IHTYP = 1 - COLLISION
C     IHTYP = 2 - STEP
C     IHTYP = 3 - ENERGY DEPOSITION
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C     REMEMBER, HBOOK IS A SINGLE PRECISION ENGINE
C     DON'T FORGET THE 'REAL' DECLARATIONS SUCH AS:
C     REAL AA,ELB,X1,Y1,Y1,Y2
C     CALL HBOOKB(ID,AA,NEB,ELB,0.)
C     CALL HBOOK2(ID,TITLE,NX,X1,Y1,NY,Y1,Y2,0.)
      RETURN
      END


      SUBROUTINE MFILL
C     (IHTYP,NREG,IM,JJ,E1,E2,DELE,W,X1,Y1,Z1,X2,Y2,Z2,DCX,DCY,DCZ,STEP,TOF,NI,IDPRC)
C     HISTOGRAM ENTRY FOR USER-DEFINED HISTOGRAMMING
C     HISTOGRAM ID AVAILABLE: 700<ID<999
C     CALL TYPE:
C     IHTYP = 1 - COLLISION
C     IHTYP = 2 - STEP ("TRACK-LENGTH")
C     IHTYP = 3 - ENERGY DEPOSITION (LOCAL OR ON THE STEP)
C     NREG - REGION NUMBER FOR COLLISION OR STEP START
C     STEP - STEP (cm)
C     E1 - ENERGY BEFORE STEP (GeV)
C     E2 - ENERGY AFTER STEP (GeV)
C     DELE - ENERGY DEPOSITED (GeV), LOCALLY OR ON THE STEP
C     W - STATISTICAL WEIGHT
C     X1,Y1,Z1 - COORDINATES AT THE STEP START (cm)
C     X2,Y2,Z2 - COORDINATES AT THE STEP END (cm)
C     DCX,DCY,DCZ - DIRECTION COSINES AT THE STEP START
C     TOF - TIME-OF-FLIGHT AT THE STEP END (sec)
C     NI - HISTORY NUMBER
C     IDPRC - PROCESS ID
C     IHTYP=1:  E1=E2, DELE=0, STEP=0, X1=X2, Y1=Y2, Z1=Z2
C     IHTYP=3:  E1.ne.E2, DELE>0; STEP=0 (X1=X2, Y1=Y2, Z1=Z2) OR STEP>0 (X1.ne.X2, Y1.ne.Y2,
C     IDPRC PROCESS
C     ************************************************************************
C     1 DPA ELASTIC AND INELASTIC
C     2 RECOIL NUCLEUS LOCAL DEPOSITION (NON-LAQGSM)
C     3 Local deposition of heavy ions at AA-vertex (NON-LAQGSM)
C     4 STAR DENSITY
C     5 FISSION CEM LOCAL (NON-LAQGSM)
C     6 FISSION >5 GEV LOCAL (NON-LAQGSM)
C     7 HEAVY FRAGMENTS LOCAL DEPOSITION (NON-LAQGSM)
C     8 d, t, He3 and He4 LOCAL DEPOSITION (NON-LAQGSM)
C     9 SURFACE CROSSING BY HADRONS AT IND(6)=T
C     10 STAR DENSITY AT IND(6)=T
C     11 FLUENCE AT IND(6)=T
```

```
C  12 ENERGY DEPOSITION by SUB-THRESHOLD hadrons
C  13 SURFACE CROSSING BY NEAR-THRESHOLD HADRONS
C  14 FLUENCE BY NEAR-THRESHOLD HADRONS
C  15 ENERGY DEPOSITION ON STEP BY NEAR-THRESHOLD HADRONS
C  16 SURFACE CROSSING BY NEUTRAL HADRONS OR IN VACUUM
C  17 SURFACE CROSSING BY CHARGED HADRONS IN MATERIAL
C  18 ENERGY DEPOSITION ON STEP BY CHARGED HADRONS
C  19 FLUENCE BY HADRONS
C  20 ENERGY DEPOSITION BY SUB-THRESHOLD e+e- from hadrons
C  21 e+e- vertex by muon
C  22 FLUENCE BY MUONS
C  23 SURFACE CROSSING BY MUONS
C  24 ENERGY DEPOSITION ON STEP BY MUONS
C  25 BREMS VERTEX BY MUONS
C  26 NUCL VERTEX BY MUONS
C  27 ENERGY DEPOSITION BY SUB-THRESHOLD BREMS OR NUCL PRODUCTS OF MUONS
C  28 ENERGY DEPOSITION BY SUB-THRESHOLD MUONS
C  29 ENERGY DEPOSITION BY HEAVY FRAGMENTS FROM MUON CAPTURE
C  30 MUON DECAY VERTEX
C  31 SURFACE CROSSING BY EMS
C  32 FLUENCE BY EMS
C  33 ENERGY DEPOSITION ON STEP BY e+ and e-
C  34 ENERGY DEPOSITION BY SUB-THRESHOLD EMS
C  35 DELTA-ELECTRON VERTEX
C  36 LE neutron vertex
C  37 LE NEUTRON FLUENCE ON STEP
C  38 LE NEUTRON SURFACE X-ING
C  39 LE NEUTRON SURFACE CROSSING AT IND(6)=T
C  40 LE NEUTRON FLUENCE AT IND(6)=T
C  41 LE NEUTRON LOCAL ED: no new n or gamma generated
C  42 LE NEUTRON LOCAL ED: recoil proton below 50 keV (E1=Ep)
C  43 LE NEUTRON LOCAL ED: deuteron in non-LAQGSM (E1=Ed)
C  44 LE NEUTRON LOCAL ED: vertex non-fission MCNP (E1=Eterm)
C  45 LE NEUTRON LOCAL ED: capture on Li or B in non-LAQGSM (E1=Ehi)
C  46 LE NEUTRON LOCAL ED: sub-threshold (E1=En)
C  47 LE NEUTRON LOCAL ED: SUB-THRESHOLD AT IND(6)=T
C  48 LE NEUTRON LOCAL ED: fission
C  49 LE NEUTRON LOCAL ED: vertex non-fission BNAB
C  50 FLUENCE BY NEUTRINO
   IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
C  REMEMBER, HBOOK IS A SINGLE PRECISION ENGINE
C  DON'T FORGET CONVERSIONS OF THE FOLLOWING TYPE:
C  REAL EEH,WWH,XL,YL,WH
C  EEH=REAL(E1)
C  WWH=REAL(W)
C  CALL HFILL(ID,EEH,0.,WWH)
C  CALL HF2(ID,XL,YL,WH)
   RETURN
   END
```

5.14 Subroutine WRTSUR - Custom Output

```
      SUBROUTINE WRTSUR(IUNIT,NI,JJ,E,W,X,Y,Z,DCX,DCY,DCZ,TOFF)
C     PARTICLE AT THE SURFACE DETECTOR
C     FOR WRITING, HISTOGRAMMING ETC.
C     UNIT=81-90 WRITE: fort.81 - fort.90 RESERVED FOR SURFACE DETECTOR FILES
C     INTAG parameter of tally2.inc can be used as a flag for a source
C     particle to propogate it through the code to WRTSUR, RAYN, RAYMU etc.
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)
      INCLUDE 'cmasnsg.inc'
      INCLUDE 'tally2.inc'
      COMMON/BPRIMHIT/PRIMEHITX,PRIMEHITY,PRIMEHITZ
      COMMON/BPRTAG/XORIG,YORIG,ZORIG,WORIG,EORIG,IORIG,KORIG,NRORIG,IMORIG
      PARAMETER (CLIGHT=29979245800.D0)
      PARAMETER (IWRTYP=1) !  momentum components (particle production)
C     PARAMETER (IWRTYP=2) !  energy and cosines (source term for staging)
C     PARAMETER (IWRTYP=3) !  energy, cosines etc for heavy ions
      IF(IWRTYP.EQ.1) THEN
      ET=E+PM(JJ)
      PA=SQRT(E*(E+2.D0*PM(JJ)))
      CTOFF=CLIGHT*TOFF
      PX=PA*DCX
      PY=PA*DCY
      PZ=PA*DCZ
      WRITE(IUNIT,101)NI,JJ,X,Y,Z,PX,PY,PZ,ET,CTOFF,W
 101  FORMAT(I8,I3,3F11.3,6(1PE14.6))
      ELSE IF(IWRTYP.EQ.2) THEN
      WRITE(IUNIT,102)NI,JJ,E,W,X,Y,Z,DCX,DCY,DCZ,TOFF
 102  FORMAT(I8,I3,5(1PE13.5),3F14.10,1PE13.5)
      ELSE IF(IWRTYP.EQ.3) THEN
      TH=ACOS(DCZ)
      IF(HZA1.GT.0.D0) THEN
      EA=E/HZA1
      ELSE
      EA=E
      END IF
      WRITE(IUNIT,103)NI,JJ,HZZ1,HZA1,W,EA,TH,X,Y,DCX,DCY
 103  FORMAT(I11,I7,2F8.2,5(1PE12.4),2(1PE15.7))
      END IF
      RETURN
      END
```

5.15 Subroutine TAGPR - Particle Origin Tagging

Origin tagging for hadrons, muons, heavy ions and electromagnetic showers (EMS). Particle JJ of weight W and energy $E1$ makes a step $STEP$ starting from point (X, Y, Z) in region $NREG$. Both this point and the step fully belong to the region $NREG$ with material index IM. Particle energy at the end of the step is $E2$. The current history number is NI. $STEP = 0$ for sub-threshold particles. $E1 = E2$ for neutrals and in vacuum.

This particle (JJ is not 9, 10 or 11) or the EMS (JJ is 9, 10 or 11) was originated by particle with ID=$IORIG$ of energy $EORIG$ and weight $WORIG$ in the process $KORIG$ at the point

($XORIG, YORIG, ZORIG$) in the region $NRORIG$ with material index $IMORIG$. $STEP > 0$ for tracking, while $STEP = 0$ for local energy deposition ($\Delta E = E1 - E2$).

```
      SUBROUTINE TAGPR(NREG,IM,JJ,W,E1,E2,X,Y,Z,STEP)
C.................................................
*     Origin tagging for hadrons, muons, heavy ions and electromagnetic showers (EMS).
*
*     Particle JJ of weight W and energy E1 makes a step STEP starting from
*     point (X,Y,Z) in region NREG. Both this point and the step fully belong
*     to the region NREG with material index IM. Particle energy at the end
*     of the step is E2. The current history number is NI.
*     STEP=0 for sub-threshold particles. E1=E2 for neutrals and in vacuum.
*
*     This particle (JJ is not 9, 10 or 11) or the EMS (JJ is 9, 10 or 11) was
*     originated by particle with ID=IORIG of energy EORIG and weight WORIG in
*     the process KORIG at the point (XORIG,YORIG,ZORIG) in the region NRORIG with
*     material index IMORIG.
*
*     STEP > 0 for tracking
*     STEP = 0 for local energy deposition (Delta_E = E1 - E2)
*
*     PARTICLE ID (JJ):
*  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18   19  20   21
*  p   n  pi+ pi-  K+  K- mu+ mu-  g  e-  e+ pbar  pi0  d   t  He3 He4 num nuam nue nuae
*  22  23  24  25  26  27   28   29   30   31   32   33   34   35   36   37   38   39    40
* K0L K0S  K0 AK0 LAM ALAM SIG+ SIG0 SIG- nbar KSI0 KSI-  OM- sib- si0b sib+ AKSI0 ksib+ omb+
*
*  JJ    is PARTICLE ID: = 1-40 for tabulated "elementary" ones
*                        = 1000*Z1 + A1 - Z1 for A1>4
*
*     IORIG = IKORIG(1,...) - projectile ID
*     KORIG = IKORIG(2,...) - origin ID
*     NRORIG= IKORIG(3,...) - origin region number
*     IMORIG= IKORIG(4,...) - origin material index
*
*     KORIG = 0 - primary beam
*             1 - muons,   unstable particle decay
*             2 - muons,   prompt at hA-vertex
*             3 - muons,   Bethe-Heitler pair
*             4 - muons,   e+e- annihilation
*             5 - hadrons, hA-vertex
*             6 - hadrons, elastic
*             7 - hadrons, from muons
*             8 - hadrons, unstable particle decay
*             9 - hadrons, EMS
*            10 - hadrons, recoil LEN
*            11 - hadrons, from neutrinos
*            12 - EMS,     induced by photons from pi0-decay
*            13 - EMS,     induced by synchrotron photons
*            14 - EMS,     induced by g,e+,e-, at hA vertex
*            15 - EMS,     induced by knock-on electrons from muons or hadrons
*            16 - EMS,     induced by g,e+,e- from unstable particle decay
*            17 - EMS,     induced by prompt e+e- from muons or hadrons
*            18 - EMS,     induced by brems photons from muon
*            19 - EMS,     induced by photons from stopped muons
*            20 - EMS,     induced by photons from low-energy neutrons
*            21 - muons,   vector mesons

C-----
C     CREATED:     11-AUG-2003 BY NVM
```

```
C                  28-APR-2006 BY NVM
C     LAST CHANGE: 26-APR-2007 BY NVM
C-----
C..................................................
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)

      COMMON
     & /BPRTAG/XORIG,YORIG,ZORIG,WORIG,EORIG,IORIG,KORIG,NRORIG,IMORIG
     & /HIST/NI,NSTOP,NUPRI,NHIPR
C- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

C=== Example ===
C      if(nreg.eq.9) then
C         write(1,100)NI,NREG,IM,JJ,W,E1,E2,STEP,
C     &        XORIG,YORIG,ZORIG,WORIG,EORIG,IORIG,KORIG,NRORIG,IMORIG
C 100     format(' NI,NREG,IM,JJ,W,E1,E2,STEP = ',i7,I5,2i3,4(1pe11.4)/
C     &        ' XORIG,YORIG,ZORIG,WORIG,EORIG,IORIG,KORIG,NRORIG,IMORIG'
C     &        ,'= ',5(1pe11.4),2i3,2i5)
C      end if

      RETURN
      END
```

5.16 Subroutine TAGGING - Energy Deposition Tagging

5.17 Subroutine KILLPTCL - Selective Particle Destruction

5.18 Block Data BLPROCESS - Biasing Control

```
    BLOCK DATA BLPROCESS
 C  IPRCEM(K) = 0 - exclusive
 C  IPRCEM(K) = 1 - inclusive with a probability PRCEM(K)
 C  IPRCEM(0) - global control
 C  Global bias control is currently done via BIAS card with parameters in
 C  COMMON/BLBIAS/PPIKDEC,PMUPRMT,PMUBEHE,PMUANN,PPHNUC,PPBAR
 C  in process.inc :  INTEGER NPRCEM,NPRCHT,NPRCMT,NPRCNT,NPRCHA
 C  in process.inc :  INTEGER IPRCEM,IPRCHT,IPRCMT,IPRCNT,IPRCHA
 C  in process.inc :  DOUBLE PRECISION PRCEM,PRCHT,PRCMT,PRCNT,PRCHA
    INCLUDE 'process.inc'
 C  cems
 C  1 2 3
 C  set EMS global bias flag:
 C  0 - exclusive
 C  1 - LPB
 C  2 - HALF
 C  DATA IPRCEM/0,NPRCEM*0/
    DATA IPRCEM/1,NPRCEM*0/
    DATA PRCEM/NPRCEM*0.D0/
 C  hadron-nucleus vertex
    DATA IPRCHA/0,NPRCHA*0/
    DATA PRCHA/NPRCHA*0.D0/
    RETURN
    END
```

5.19 Block Data HISTDUMP - Histogram Bin and DUMP Region Specification

```
      BLOCK DATA HISTDUMP
C.......................................................
C     BASELINE VOLUME-SURFACE DETECTOR BINNING
C                AND
C     SELECTED DUMP REGION NUMBERS (0 < NBMA < 300)
C     IN ADDITION TO THE STANDARD GEOMETRY SECTOR
C-----
C     CREATED:     21-NOV-2001 BY NVM
C     LAST CHANGE: 14-FEB-2002 BY NVM
C-----
C.......................................................
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)

      COMMON/BHBKRZ/NZBIN(3),NRBIN(3),NXH,NYH
      COMMON/BLDUMP/NBU(300),NBMA

C=== VOLUME R-Z DETECTOR BINNING (at NOB > 0) ===
      DATA NZBIN/200,200,200/   ! Recommended
      DATA NRBIN/120,120,120/   ! Recommended

C=== SURFACE X-Y DETECTOR BINNING (at NSURF > 0) ===
      DATA NXH/150/   ! Recommended
      DATA NYH/150/   ! Recommended

C=== SELECTED DUMP REGION NUMBERS (0 < NBMA < 300) ===
      DATA NBMA/0/
      DATA NBU/300*0/

C    For example:
C      DATA NBMA/6/
C      DATA NBU/115,116,117,921,1050,4380,
C     &   294*0/
C
      END
```

5.20 MAD-MARS Interface User Routines

Routines used to assist the Mars-Mad beamline builder.

5.20.1 Subroutine MARS2BML

```
      SUBROUTINE MARS2BML(POS,W,BLPOS,BLW)
C.........................................................
C     MARS-TO-BEAMLINE TRANSFORMATION
C     CALLED IF IND(13)=T
C
C-----
C     CREATED:     2002 BY IT
C     LAST CHANGE: 22-APR-2003 BY NVM
C-----
C.........................................................
      IMPLICIT NONE
```

```
      DOUBLE PRECISION POS(3), W(3), BLPOS(3), BLW(3)
      LOGICAL FIRSTCALL
      DATA FIRSTCALL/.TRUE./
      SAVE FIRSTCALL
      INTEGER IT
C- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

C=== EXAMPLE OF REAL WORK ===

*      INCLUDE 'beamOffset.inc'
*      DOUBLE PRECISION MADVEC(3)
*      DOUBLE PRECISION XMAD,YMAD,ZMAD
*      EQUIVALENCE (XMAD,MADVEC(1)),(YMAD,MADVEC(2)),(ZMAD,MADVEC(3))
*      DOUBLE PRECISION aaa
*      DOUBLE PRECISION MADDIR(3)
*      DOUBLE PRECISION WXMAD,WYMAD,WZMAD
*      EQUIVALENCE (WXMAD,MADDIR(1)),(WYMAD,MADDIR(2)),(WZMAD,MADDIR(3))
*
*      call mars2mad(POS(1),POS(2),POS(3),XMAD,YMAD,ZMAD)
*      call mars2mad(W(1),W(2),W(3),WXMAD,WYMAD,WZMAD)
*
*      if(POS(3).GT.NuMI_2_Z0) then
*         call set_current_beamline(2)
*      else
*         call set_current_beamline(1)
*      endif
*      if(MADVEC(3).GE.11894.92D0.AND.MADVEC(3).LE.11894.94D0) THEN
*         call glob2flat(MADVEC,MADDIR)
*      else
*         call glob2flat(MADVEC,MADDIR)
*      endif
*      call mad2mars(XMAD,YMAD,ZMAD,BLPOS(1),BLPOS(2),BLPOS(3))
*      call mad2mars(WXMAD,WYMAD,WZMAD,BLW(1),BLW(2),BLW(3))
*      if(BLPOS(3).GE.931.0626d0) THEN
*         aaa=1.d0
*      ENDIF

C=== COMMENT THIS MODULE FOR REAL WORK ===
      IF(FIRSTCALL) THEN
         WRITE(*,*)' MARS2BML: IND(13) is active, but transformation'
         WRITE(*,*)' to the beamline coordinate system is not defined.'
         WRITE(*,*)' See example in MARS2BML subroutine!'
         FIRSTCALL=.FALSE.
      ENDIF
      DO IT=1,3
         BLPOS(IT)=POS(IT)
         BLW(IT)=W(IT)
      END DO
```

```
C=========================================

      RETURN
      END
```

5.20.2 Subroutine BML2MARS

```
      SUBROUTINE BML2MARS(BML_VEC,BML_DIR,MARS_VEC,MARS_DIR)
C.......................................................
C     BEAMLINE-TO-MARS TRANSFORMATION
C     CALLED IF IND(13)=T
C
C-----
C     CREATED:     2002 BY IT
C     LAST CHANGE: 08-NOV-2002 BY NVM
C-----
C.......................................................
      IMPLICIT NONE
      DOUBLE PRECISION BML_VEC(3),BML_DIR(3),MARS_VEC(3),MARS_DIR(3)
      INTEGER IT
C- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

C=== EXAMPLE OF REAL WORK ===

*      DOUBLE PRECISION CBUF(3),DBUF(3)
*      IF(BML_VEC(3).GE.1234.0989) THEN
*         call set_current_beamline(2)
*      ELSE
*         call set_current_beamline(1)
*      ENDIF
*      CALL mars2mad(BML_VEC(1),BML_VEC(2),BML_VEC(3),
*     &     CBUF(1),CBUF(2),CBUF(3))
*      CALL flat2glob(CBUF,DBUF)
*      CALL mad2mars(CBUF(1),CBUF(2),CBUF(3),
*     &     MARS_VEC(1),MARS_VEC(2),MARS_VEC(3))

C=== COMMENT THIS DO-LOOP FOR REAL WORK ===
      DO IT=1,3
         MARS_VEC(IT)=BML_VEC(IT)
         MARS_DIR(IT)=BML_DIR(IT)
      END DO
C=========================================

      RETURN
      END
```

5.20.3 Subroutine BLINIT

```
      SUBROUTINE BLINIT(M_MAX, IMUN)
C.......................................................
C     MMBLB INITIALIZATION
C     CALLED IF IND(13)=T
C
C-----
C     CREATED:     2001 BY KRIOL
C     MODIFIED:    17-OCT-2003 BY MAK
C     LAST CHANGE: 04-DEC-2003 BY NVM
C-----
C.......................................................
      IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER (I-N)

      INTEGER IMUN(*)

      include 'blreg1.inc'
      include 'tally1.inc'

C=== EXAMPLE OF REAL WORK (uncomment and modify * whereever needed) ===

*      INCLUDE 'SIMPLE_TUNNEL.INC'
*      INITZONE= 1
*      CALL BEAMLINES_IN_USE(2)  ! EXTENSION TO  SEVERAL BEAMLINES
*      CALL SET_CURRENT_BEAMLINE(1)

c= Angle of rotation (azimuth) about vertical axis (MARS X-axis), between
c= Z axis and projection of the beamline direction vector onto (Z,Y) plane.
c= A positive angle forms a right hand screw with MARS X-axis.

*      BL1_THETA= 0.D0

c= Elevation angle, i.e. the angle between beamline direction and its
c= projection onto (Z,Y) plane. If it equals to zero, the beamline remains
c= in horizontal plane, positive phi correspond to increasing X.

*      BL1_PHI  = 0.D0

c= Roll angle about the beamline direction vector, i.e. angle between line
c= formed by intersection of beamline (x,y)-plane with (Z,Y) plane and
c= beamline x axis. A positive angle forms a right hand screw with beamline
c= direction vector

*      BL1_PSI= 0.D0

*      x_start  = 0.d0
*      y_start  = 0.d0
*      z_start  = 0.d0
```

```
*      call buildbl( 'optics1',
*     &     x_start, y_start, z_start,
*     &     bl1_theta, bl1_phi, bl1_psi,
*     &     InitZone)

*      write(*,*)'REG1:INFO: use beam Ekin=',
*     &     BeamKineticE,
*     &     ' ;beam mass',
*     &     BeamMass

*      BeamKineticE = 100.0d0
*      BeamMass     = 0.93827231d0

*      call setebl( BeamKineticE, BeamMass, 5.6780d+03 )

*      call blmaxmat( nblnzmax )
*      if ( nblnzmax .gt. M_MAX ) then
*         write(*,*) 'beamline 1: ',
*     &        'nblnzmax as read from blmatmax ',
*     &        nblnzmax
*         write(*,*) 'is greater then M_MAX', M_MAX
*         stop 'STOP'
*      else if ( nblnzmax .lt. M_MAX ) then
*         write(*,*) 'beamline 1: ',
*     &        'nblnzmax as read from blmatmax ',
*     &        nblnzmax
*         write(*,*) 'is less then M_MAX', M_MAX
*      endif

*      if ( nblnzmax .gt. 0 ) then
*         call blmat(IMUN)
*      else
*         write(*,*) 'Number of zones for NuMI beamline <= 0'
*         stop
*      endif

*      call set_current_beamline(2)

*      bl2_theta = 0.0d0
*      bl2_phi   = 0.0d0
*      bl2_psi   = 0.0d0

*      x_start  = 0.d0
*      y_start  = -300.d0
*      z_start  = 3000.d0

*      InitZone = 2000
```

```
*      call buildbl( 'optics2',
*     &     x_start, y_start, z_start,
*     &     bl2_theta, bl2_phi, bl2_psi,
*     &     InitZone)

*      call setebl( BeamKineticE, BeamMass, 5.6780d+03 )

*      call blmaxmat( nblnzmax )
*      if ( nblnzmax .gt. M_MAX ) then
*         write(*,*) 'beamline 1: ',
*     &        'nblnzmax as read from blmatmax ',
*     &        nblnzmax
*         write(*,*) 'is greater then M_MAX', M_MAX
*         stop 'STOP'
*      else if ( nblnzmax .lt. M_MAX ) then
*         write(*,*) 'beamline 1: ',
*     &        'nblnzmax as read from blmatmax ',
*     &        nblnzmax
*         write(*,*) 'is less then M_MAX', M_MAX
*      endif

*      if ( nblnzmax .gt. 0 ) then
*         call blmat(IMUN)
*      else
*         write(*,*) 'Number of zones for NuMI beamline <= 0'
*         stop
*      endif

*      do i=1, st_nof_zones
*         call simple_tunnel_mat(i, IMUN(i+st_first_zone-1))
*      end do


c...  volume names
*      call set_current_beamline(1)
*      call blname(VOLNM(NFZPEX+1))

*      call set_current_beamline(2)
C*     call blname(VOLNM(NFZPEX+1))

      RETURN
      END
```

5.20.4 Subroutine BLGEOINIT

```
      SUBROUTINE BLGEOINIT(BEAMLINE_ID)
C.........................................................
C     MMBLB ELEMENT REGISTRATION
```

```
C     CALLED IF IND(13)=T
C
C-----
C     CREATED:     2001 BY KRIOL
C     MODIFIED:    17-OCT-2003 BY MAK
C     LAST CHANGE: 04-DEC-2003 BY NVM
C-----
C.......................................................
      IMPLICIT NONE
      INTEGER BEAMLINE_ID

C=== EXAMPLE OF REAL WORK (uncomment and modify * whereever needed) ===

*      INCLUDE 'DRIFT.INC'
*      INCLUDE 'RBENDEPB.INC'

*      EXTERNAL
*     &     DRIFT_NAME_FUNC,
*     &     DRIFT_INIT_FUNC,
*     &     DRIFT_MAT_FUNC,
*     &     DRIFT_FIELD_FUNC,
*     &     DRIFT_VOL_FUNC,
*     &     DRIFT_GEO_FUNC,
*     &     DRIFT_ZONENAME_FUNC

*      IF (BEAMLINE_ID .EQ. 1) THEN
*         CALL MARS_EL_REGISTER( 1,
*     &        DRIFT_NOF_ZONES,
*     &        DRIFT_NAME_FUNC,
*     &        DRIFT_INIT_FUNC,
*     &        DRIFT_GEO_FUNC,
*     &        DRIFT_MAT_FUNC,
*     &        NO_FIELD,
*     &        DRIFT_VOL_FUNC,
*     &        DRIFT_ZONENAME_FUNC )

*         CALL MARS_EL_REGISTER( 2,
*     &        RBENDEPB_NOF_ZONES,
*     &        RBENDEPB_NAME_FUNC,
*     &        RBENDEPB_INIT_FUNC,
*     &        RBENDEPB_GEO_FUNC,
*     &        RBENDEPB_MAT_FUNC,
*     &        RBENDEPB_FIELD_FUNC,
*     &        RBENDEPB_VOL_FUNC,
*     &        RBENDEPB_ZONENAME_FUNC )

*      ELSE IF(BEAMLINE_ID .EQ. 2) THEN
*         CALL MARS_EL_REGISTER( 1,
*     &        DRIFT_NOF_ZONES,
```

```
*     &        DRIFT_NAME_FUNC,
*     &        DRIFT_INIT_FUNC,
*     &        DRIFT_GEO_FUNC,
*     &        DRIFT_MAT_FUNC,
*     &        NO_FIELD,
*     &        DRIFT_VOL_FUNC,
*     &        DRIFT_ZONENAME_FUNC )

*         CALL MARS_EL_REGISTER( 2,
*     &        RBENDEPB_NOF_ZONES,
*     &        RBENDEPB_NAME_FUNC,
*     &        RBENDEPB_INIT_FUNC,
*     &        RBENDEPB_GEO_FUNC,
*     &        RBENDEPB_MAT_FUNC,
*     &        RBENDEPB_FIELD_FUNC,
*     &        RBENDEPB_VOL_FUNC,
*     &        RBENDEPB_ZONENAME_FUNC )
*      END IF

      RETURN
      END
```

5.20.5 Subroutine TUNNELGEO

```
      SUBROUTINE TUNNEL_GEO(XLOCMAD,YLOCMAD,ZLOCMAD,XGLMAD,YGLMAD,
     &     ZGLMAD,NZ,BLINDX,ELINDX,ELBEGINS,ELLENGTH,ANGLE,PREVANGLE,
     &     NEXTANGLE)

      IMPLICIT NONE

      INTEGER NZ, BLINDX, ELINDX
      DOUBLE PRECISION XLOCMAD,YLOCMAD,ZLOCMAD,XGLMAD,YGLMAD,ZGLMAD,
     &     ELBEGINS,ELLENGTH,ANGLE, PREVANGLE, NEXTANGLE

      RETURN
      END
```

5.21 User Subroutine Examples

5.21.1 Simple Model of Beam on a Target

Example 1. Calculate antiproton production with 9-cm long 1-cm diameter copper target irradiated with 120-GeV proton beam. Beam R.M.S spot size is σ_x=0.005 cm and σ_y=0.007 cm. In addition to forced antiproton production the user is interested in energy deposition calculation including knock-on electron and e^+e^--pair production by hadrons. To create a file of antiprotons generated on the target in a given phase space, the user adds in MAIN:

```
OPEN (UNIT=9,FILE='PBAR.OUT',STATUS='UNKNOWN')
```

and a few statements in the LEAK routine (Section 5.12). The geometry and scoring is described in the *standard* mode. The MARS.INP file can look as:

```
Pbar Cu Target, sigx=0.005, 02-Nov-1994
INDX T 5=T 12=T
NEVT 100000
ENRG 120.
IPIB 1 2
BEAM 0.005 0.007
SMIN 0.001 3.
MATR 'CU'
ZSEC 9.  51=3
NLTR 5
RSEC 0.002 0.005 0.01 0.1 .5
STOP
```

The geometry is very simple here, completely adequate to the *standard* mode. Instead of the above bining and writing to a file PBAR.OUT, one can use HBOOK (see MAIN) for analyses of energy deposition and generated antiprotons. Then the solid target can be described just as:

```
ZSEC 9.
RSEC 0.5
```

5.21.2 Beam Dump

Intense 800-GeV proton beam hits a graphite dump followed by aluminum and steel absorbers, by 210-cm vacuum gap, by 5-cm polyethylene slab, and finally by 1 meter of a wet dirt. The user is interested in energy deposition calculations, in maximum amount of the output, including partial distributions, temperature rise and estimation of a residual dose rate, in intermediate result dumps. A compound material is defined in the MIXTUR routine (Section 5.3). The routine REG1 (Section 5.5) defines the dump as surrounded with a steel shield sitting at the axis of a hypothetical cylindrically symmetrical tunnel with 30-cm thick concrete walls. With these three user routines a *standard* MARS.INP file can look as (*Option 1*):

```
Tevatron C0 Dump, 02/28/95
INDX T T 8=T
NEVT 500000 5
IPIB 1 2
ENRG 800.
SMIN 0.01 8.
BEAM 0.0416 0.0944
VARS 0.  1.  300.  4.E13
NMAT 6
MATR 'C' 'AL' 'FE' 'CONC' 'CH2' 'MIXT'
26=2.1
NLNG 7
ZSEC 350.  425.  475.  490.  700.  705.  805.
51=14 5 10 1 3 2 5 101=1 2 3 2 0 4 5
NLTR 9
RSEC 0.03 0.1 0.3 1.  3.5 10.  30.  150.  180.
PLOT 5.  300.  450.  805.
STOP
```

Option 2. The geometry description is simpler and the output is more sophisticated if one uses HBOOK activated in MAIN. The last lines in MARS.INP can be re-written as:

```
NLNG 7
ZSEC 350.  425.  475.  490.  700.  705.  805.
101=1 2 3 2 0 4 0 5
NLTR 4
RSEC 10.  30.  150.  180.
```

6 MCNP Mode

MCNP is a code package for precise simulation of low energy neutrons. MARS uses portions of the MCNP package by default when neutrons fall below 14 MeV, however this default mode is not sufficient for certain applications. In particular, when predicting residual dose rates, and when predicting prompt dose rates in labyrinth type geometries, the full MCNP package should be used to take advantage of it's detailed models of the interactions of interest. Residual rates are sensitive to the materials present, and prompt dose rates are sensitive to the $n - \gamma$ reactions which occur when slow neutrons are stopped and captured; the MCNP code package excels at the simulation of both these effects. MCNP code contains very detailed material handling, taking into account the nuclear physics details that occur in interactions with different nuclei, and this affects residual rate calculations. MCNP also includes $n-\gamma$ reactions when the neutron is finally stopped and captured. These reactions affect the prompt dose calculations in labyrinth type geometries where slow neutrons are dominant.

MARS can be used in conjunction with the full MCNP code package: MARS generates the initial neutron but sends it to MCNP for tracking and interactions. Once tracked to capture or to it's cutoff energy, the accumulated information is handed back to MARS for compilation of tallies and results. In general, the model geometry is specified using the usual collection of Mars zones, and MCNP handles the details of the declared materials. The user gives MCNP the information it needs on materials via MCNP data entry cards appended to the `MARS.INP` file after the **`STOP`** card. However, if the user already has a MCNP geometry description file, then the model geometry can be described using that syntax, which is also appended to the end of the MARS.INP file. Both these modes of using MCNP are described below in Section 6.1.

Using MARS with the full MCNP code requires that the user obtain and install the MCNP code library. The code is obtained from RSICC [84], the Radiation Safety Information Computational Center, at the Oak Ridge National Laboratory, on the web at www-rsicc.ornl.gov, or from the NEA Databank in Europe [85]. The code package includes the manual, which is not available online. Once the code library is installed, the `GNUmakefile` distributed with MARS must be used to link to the MCNP libraries and produce the non-default MARS-MCNP executable. To run the executable, the user must set up pointers to the MCNP data library. See Section 6.2 for installation and compile/build/run details.

6.1 Setting up `MARS.INP` for use with MCNP

Switches must be set on the **`INDX`** card in the `MARS.INP` file to activate the calling of the MCNP modules, even if the executable has been built against the full MCNP libraries. MCNP data entry cards are then appended to the end of the `MARS.INP` file, after the **`STOP`** card. Which switches and data cards are used depends on which of the two MCNP sub-modes is being used.

The first sub-mode has all the geometry description handled by MARS, with MCNP handling the details of neutron interactions; in this case only MCNP material correspondence data cards are appended to the `MARS.INP` file. Details on the syntax of MCNP material data cards is in Section 6.1.1.

In the second sub-mode the MCNP geometry module is used to describe the arrangement of materials where detailed neutron interactions are performed. So in addition to the material data cards, MCNP geometry data cards are also present. These data lines are defined in Section 6.1.2. In this second mode, the full MARS physics modeling is still performed within the geometry described by the MCNP geometry data lines, and there can be portions of the model's geometry defined by the usual MARS zones, outside the MCNP geometry description, where MCNP will continue to handle neutron interactions.

All of the MCNP data cards are located after the `MARS.INP` **`STOP`** card, and must be contained between the MCNP data delimiter lines: the start card, `*MCNP START` and the stop card `*MCNP STOP`. A MCNP

data card, either the material or geometry type, is a free-format input up to 80 characters on a single line; a single data card can continue past a single line by using the continuation indicator & at the end of a line. Data cards of a given type must be grouped together on sequential lines, and not intermixed. Blank lines are treated as delimiters between the groups of material and geometry card types.

6.1.1 Mode 1 – MCNP with Material description only

To trigger the calling of the MCNP subroutines, the **INDX** card in the `MARS.INP` file must have `5=T` in it's list of switches.

Next one inserts the MCNP start and stop delimiter cards after the `MARS.INP` **STOP** card. Between the MCNP delimiter cards, the MCNP material data cards are inserted.

Each MCNP material data card is made up of an identifier, followed by pairs of numbers, each pair consisting of an integer and a real. The integer identifies an isotope, and the real represents the proportion of that isotope within the material being described. Each number is separated by one or more spaces. There can be as many number pairs as necessary, but a single line cannot exceed 80 characters. If the data requires more than one line, then the line is terminated by a continuation character, which is the ampersand &; the ampersand is included within the 80 character limit. The card's data continues directly on the following line, with no blank line between.

The material data card syntax is –

mIM $\texttt{Iso}_1$ $\texttt{Frac}_1$ $\texttt{Iso}_2$ $\texttt{Frac}_2$ ... $\texttt{Iso}_n$ $\texttt{Frac}_n$ [$\texttt{C}_n$]

where

mIM	is the material identifier. The prefix "m" is always present, followed by an integer IM, where this integer corresponds to the order materials are listed on the **MATR** card in the MARS portion of the `MARS.INP` file
$\texttt{Iso}_n$	an integer, is the isotope identifier part of each number pair. Isotopes in MCNP are described in terms of `ZZAAA` notation. The first two digits give the atomic number while the last three ones give the atomic mass. Thus, 92238 describes uranium-238, 5010 describes boron-10. If all of the last three digits are zero, then the natural mixture of isotopes for that element is assumed.
$\texttt{Frac}_n$	a real number, is the proportion of this isotope in the material being described. The proportions can be given by either volume percentages, or by weight fractions. The first are positive values, the second are negative values. All of the proportions for a given material must be one or the other, i.e. a single material data card cannot mix volume and weight fractions.
$\texttt{C}_n$	are optional parameters which select certain conditions for the material, for example "gas=1" would indicate the state of the material. See the MCNP manual for details.

So, for example, the line

```
m1 26000 0.8 6012 0.2
```

is the MCNP material card describing the first MARS material by volume, with 80% of atoms being natural iron (26000) and 20% of atoms being carbon-12 isotope. The following line is for the same material, but instead using weight fractions:

```
m1 26000 -0.95 6012 -0.05
```

The atom or weight fractions entered within a material card can be unnormalized. MCNP performs re-normalization by default.

In the MCNP mode, any material used by the geometry must have an MCNP material card. The MCNP material data cards do not need to be listed in the same order as the materials listed on the **MATR** card. The index IM must simply match between the two.

Vacuum regions are identified in MARS by either the material index $IM = 0$, or by a 'VAC' entry in the **MATR** card. If the geometry contains vacuum in the region of interest to MCNP, then 'VAC' should be listed on the **MATR** card. The corresponding MCNP material card must comply with the syntax, but the data values entered will be ignored, and at the preprocessing stage MCNP will be given vacuum for that material.

The following is the Z-sandwich Standard zone geometry example from Section 4.3.1, and shown in Figure 14, with MCNP material data cards added which correspond to the materials declared on the **MATR** card.

```
MARS15 Example:  Z-sandwich geometry with MCNP material cards
/home/mokhov/restricted/mars15/dat
INDX 2=T 5=T 6=T
NEVT 1000
ENRG 100.  0.05 0.05
IPIB 3 3
BEAM 5.0 5.0 0.1 0.1
SMIN 0.2 5.
NMAT 3
MATR 'FE' 'AIR' 'CONC'
NLNG 3
ZSEC 50.  150.  350.  1252=4 4 2501=2 1 3
NLTR 1
RSEC 100.  51=4
STOP
*MCNP START
m1 6000 -0.001 14000 -0.001 25055 -0.004 26000 -0.982 &
28000 -0.010 29000 -0.002 cond=1
m2 7014 0.78443 8016 0.21076 18000 4.671E-3 6000 1.39E-4 gas=1
m3 1001 -0.006 6000 -0.030 8016 -0.500 11023 -0.010 13027 -0.030 &
14000 -0.200 19000 -0.010 20000 -0.200 26000 -0.014
*MCNP END
```

The file `xsdir`, which is installed with the MCNP package, contains the library of material data. However it can contain several references to different evaluated nuclear data sets for the same material. By default, MCNP reads in the first reference to the corresponding nuclear data set encountered in the file `xsdir`. Specific suffixes are used differentiate between the data sets, and these suffixes can be appended to the isotope identifier as a means to select a particular data set for that material. For example, if one listed

```
m1 26000.42c 1
```

then the specified "42c" nuclear data set would be used for this iron material. See the MCNP manual for details.

6.1.2 Mode 2 – MCNP with both Material and Geometry description

Just as for the material data only mode, the **INDX** card in the `MARS.INP` file must have `5=T` in it's list of switches, but in addition the 16=T switch must also be set. Setting both these switches triggers calls to both the MCNP material modeling and geometry description subroutines.

The MCNP geometry has simularities to MARS Extended Geometry, in that items are built from a collection of geometric shapes. For those familiar with MCNP, MARS15 supports all of the MCNP geometry

capabilities except lattices and universes. The usual set of regular surfaces is present: plane, sphere, cylinder, cone, ellipsoid, hyperboloid, paraboloid, and elliptical and circular torus. In addition, it is possible also to use so-called macrobodies, which are composed of the regular surfaces mentioned, just like a box is composed of six different planes. The following macrobodies are allowed at present: arbitrarily oriented orthogonal box, rectangular parallelpiped, sphere, right circular cylinder, right hexagonal prism.

Using these shapes to define a geometry adds two more groups of data to the MCNP section of the `MARS.INP` file, where the first group is the material data cards described in the previous Section. The two added groups are to define geometric zones, called cells in MCNP, and to define the surfaces which bound the zones. Each of the three groups of data cards is required to be present using MCNP material+geometry mode; data cards of each group cannot be intermixed, but each sit in their own block of cards, with the card groups separated by a blank line delimiter.

The surface data cards are a shorthand syntax of the equation $f(x, y, z) = 0$, where equality to zero is assumed and not written. Each of the cards contains a mnemonic name of the surface and numerical entries indicating the extent or location of the surface; consult the MCNP manual, distributed with the code, for a listing of surface names and their numerical descriptors. The surface data card syntax is –

`ID`$_s$ `Sname Val`$_1$ `Val`$_2$ `... Val`$_n$

where

`ID`$_s$	an integer, is the surface ID number. Each surface has a unique ID number. Surface data cards do not need to be listed in order by their ID, and there can be gaps in the ID number values for a given group of cards.
`Sname`	a short character string, which is a mnemonic name of the surface, such as "`pz`" for an infinite plane orthogonal to the Z-axis. Refer to the MCNP manual for the collection of names.
`Val`$_n$	a string of one or more numbers, integer or real, depending on the surface being described. The values give information on the size and/or location of the surface, according to the equation used to describe the surface. For example, in the case of a plane orthogonal to the z-axis, the single value `10.` would be entered, as this is the shorthand for the equation $z - 10. = 0$ which describes such a plane located at z =10.

In general surfaces are defined in the Mars global coordinate system. If the surface cards are defined in a local coordinate system, then a transformation matrix between the local and global systems must be supplied by the user.

Zones are built from surfaces. Each surface has two sides, a positive side where points satisfy $f(x, y, z) > 0$, and a negative side where all points satisfy $f(x, y, z) < 0$. To first order, a zone is constructed by stating which sides of a selection of defined surfaces bound it. This is done by attaching a $+$ or $-$ sign to the surface ID numbers, `ID`$_s$, to indicate which of the sides, positive or negative, is assigned to the zone (the $+$ is optional while the $-$ is mandatory). In addition, the boolean operators `AND` and `OR` are used to describe how intersecting surfaces define the zone. One or more blank spaces between any two surface ID numbers implies the `AND` operation applied to their spaces, to form an intersection of the spaces. A colon ":" separating two surface ID numbers implies an `OR` operation, to form a union of the spaces. Intersections are performed first and then unions. Complicated structures, where the hierarchy of the boolean operations are nested, are allowed. The main restriction on these operations is that the entire space of a zone must have the same sign relative to each of its bounding surfaces; in other words, the zone cannot be partly positive and partly negative relative to its bounding surfaces.

The zone data card syntax is –

`ID`$_z$ `IM` ρ `ID`$_s$`-string IMP`

where

`ID`$_z$	an integer, is the zone card ID number. Each zone has a unique card ID number, *however this is not the assigned Mars zone number*. The card IDs do not need to be listed in order, and there can be gaps in the ID numbers for a given group of cards.
`IM`	the material index for the material contained within this zone. This is the same index from the order materials are listed on the **`MATR`** card, and the same integer as used for the MCNP material ID, `m`IM
ρ	the density of the material. A negative value implies the units are g/cm^3, while a positive value implies units of atom density in 10^{24} atom/cm^3.
`ID`$_s$`-string`	a string of integers, which are the defined surface IDs, separated by spaces or by ":", which define the zone boundaries according to the rules in the above paragraph
`IMP`	a string of characters which define the relative importance for particles which enter this zone – importance in terms of their contribution to the result. By default the values of importance are set equal to unity. For a card describing a blackhole zone, both the material number and importance must be zero, and the material density value must be omitted. See the MCNP manual for more details.

These MCNP zones are treated by MARS as Non-Standard zones. They are numbered by MARS in the order in which they appear in the MCNP section of the `MARS.INP` file. Non-Standard zones are numbered starting from $NFZP + 1$ where $NFZP$ is the number of Standard zones. Recall that there must be at least one Standard Zone defined in the `MARS.INP` file, a container zone which surrounds all other defined zones. Also keep in mind that the MCNP geometry cards can be used in conjunction with all other methods which MARS uses to define zones. If the model contains encoded Non-Standard zones in addition to the MCNP geometry zones, then the user must check for conflicts between the zone numbers assigned by the users code and the zone numbers MARS assigns to the MCNP zones.

Also recall that MARS must have the volumes of all defined zones in order to calculate various quantities. For any MCNP zones having a cylindrical symmetry, MCNP itself calculates volumes, and hands this information to Mars. Otherwise the user must provide the zone volumes. Volumes can be calculated "by hand" or by running Mars in "volume MC mode" once all the zones are defined; see Section 3.1.1. Once the MCNP zones volumes are known, their values can be listed in the VFAN subroutine with the other Non-Standard zones (Section 5.6), or they can be listed via MCNP volume data cards, which are grouped with the MCNP material data cards. The syntax is –

`vol` V_1 V_2 V_3 `...` V_n

where "`vol`" identifies this as a MCNP volume card, and the V_n are the volume values for each MCNP zone, listed the order in which the zones are listed.

An example of a simple MCNP geometry description is given below.

```
*MCNP START
10 1 -0.00129 1 -2 -3 imp:n,p=1
20 2 -7.87 1 -2 3 -4 imp:n,p=1
90 0 -1:2:4 imp:n,p=0

1 pz 0
2 pz 100
3 cz 5
4 cz 5.2

m1 7014 -0.7494 8016 -0.2369 18000 -0.0129 1001 -0.0008
m2 26000 1
vol 7854 640.9
*MCNP END
```

The sample represents a segment of a beam-pipe filled with air. Material 1 is air and material 2 is iron. The surface cards define four surfaces, two planes and two cylinders. Both planes are orthogonal to the z-axis, with the first plane at z =0 and the second at z =100. Both cylinders are aligned on the z-axis and are infinite in length, with the first having a radius of 5 and the second a radius of 5.2. Zones 10 and 20 use the surface IDs to define two cylindrical regions 100-cm long, situated from 0 up 100 cm along z-axis. In radius the zones extend from 0 to 5 cm and from 5 to 5.2 cm. The two zones are filled with air and iron, respectively. The third zone is defined to lie outside of the outer cylinder, and contains $blackhole$ material; this zone has been given an importance of zero. The volumes of the zones (in cm^3) have been entered on a MCNP volume card, listed after the MCNP material cards.

6.2 MCNP Installation

The MCNP code package, with corresponding libraries, can be obtained from the RSICC center in the USA [84] or NEA Databank in Europe [85]. The installation procedure for the MCNP code is described in the installation guide, included with the package. The MCNP manual is also included. The user must run the **install** procedure and follow the on-screen instructions carefully. While configuring and installing MCNP, choose the following options:

```
Dynamic memory:  Off with default mdas size

Geometry Plotter:  Off

Tally Plotter:  Off

64-Bit Data:  Off

Multiprocessing:  Off
```

After completion this procedure and successful running the built-in test problems, next step is library compilation.

To build the MCNP libraries, usually only Fortran and C compilers are needed. However, for use in conjunction with MARS, it is recommended to follow the procedure described below, and in this case two additional software packages are required: a `GNU make` from the Free Software Foundation [99], and a **fsplit** utility for splitting the fortran code into separate subroutines. Some Linux distributions do not include **fsplit**, therefore, a free utility source code from the BSD is included in the installation package as described below. All these files can be downloaded from the secure section of the MARS web page [100].

Create a new directory and copy from the MCNP installation directory the following components:

- `mcnpf.id`
- `prpr`
- `patchf`

Copy to the same directory the following files downloaded from the MARS web page:

- `GNUmakefile`
- `source.patch`
- `fsplit.c`
- `lsignal.c`

Check `GNUmakefile` for commands and compiler options appropriate for you system. If there is no a **fsplit** utility on your system, create it by means of the following command

`cc -o fsplit fsplit.c`

Now run the `GNUmakefile` and wait for several minutes until the procedure is complete. Upon its successful completion, a MCNP library called, for example, **libmcnp4Csun.a** on a Sun system, will be created. The library must be moved to the directory .../restricted/mcnp4c/os/lib, where "os" is sun, linux, dec, aix, or irix depending on the OS of your computer, and "..." looks like /home/mokhov for most MARS installations.

6.2.1 Building and running MARS-MCNP

To create a MARS-MCNP executable, the user copies the distributed MARS user-subroutines and `GNUmakefile` to his local directory, as described in Section 3.11. This is the `GNUmakefile` which is distributed with the MARS code, not the one used to build the MCNP libraries. One specifies the desired non-default executable by entering it's name on the command line

make rmars-mcnp-os

instead of simply using "make" which results in the standard Mars executable.

The **xsdir** file or a link to it must be present in the directory where the user runs the MARS-MCNP executable. This file is a set of neutron, photon, and electron data libraries which are included with the MCNP distribution package. The user should copy the file from the MCNP installation area in the directory .../restricted/mcnp4c/data in the MARS installation area. Then, insert the following line at the top of the file **xsdir** file:

`datapath=.../restricted/mcnp4c/data`

The MCNP code creates three output files after the run. First, `mcnp.buf`, is temporary file generated at the initialization stage. It is re-written during each run, however if problems occur a bug report is written here. The second file is `outp`. It contains information on how the `mcnp` code parsed it's lines from the `MARS.INP` file, and can be useful for checking the specified atomic and weight percentage of material mixtures. The third file, `runtpe`, contains binary information required for a hot restart, and this file can be safely deleted. All physics results are printed to the usual MARS output files, tabulated by zone number.

7 FLUKA Geometry Mode

MARS15 can read in an input geometry description in the FLUKA format and perform full MARS modeling in such a setup. An example of a MARS model of the CMS collider detector based on a FLUKA input is shown in Fig. 24.

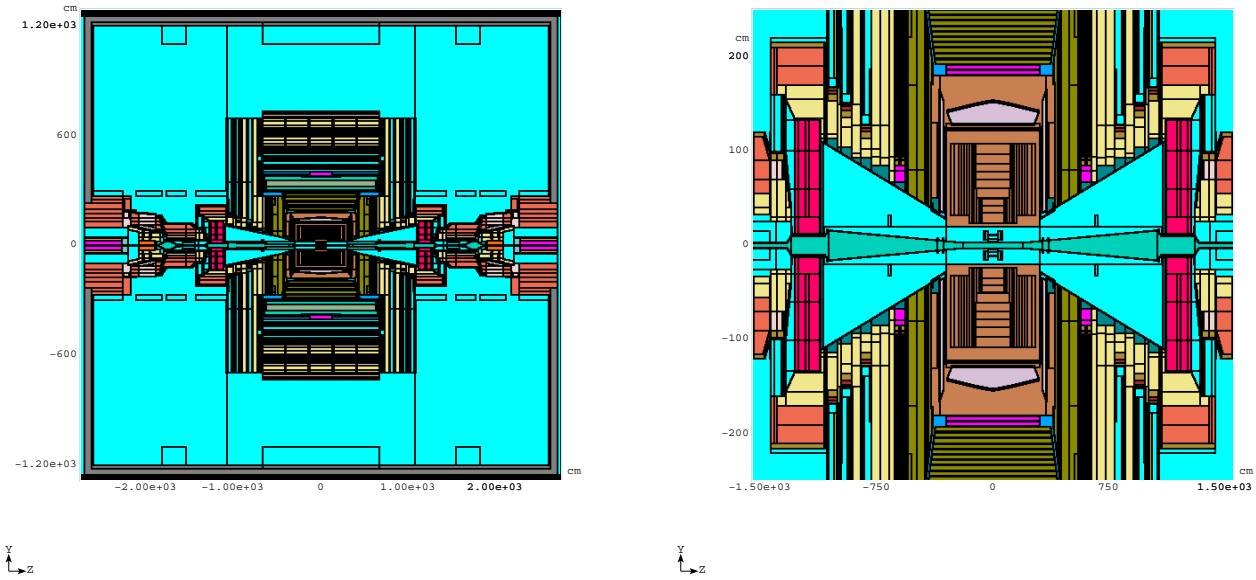


Figure 24: Mars GUI illustration of the CMS detector, based on a FLUKA input geometry description: overall view (left) and a central fragment (right).

8 LAQGSM and DPMJET Modes

8.1 LAQGSM

The 2007 version of the Los Alamos Quark-Gluon String Model code, LAQGSM07, was implemented into MARS15 for particle and heavy-ion projectiles [11]. Such processes as π^--capture and $\overline{p}$-annihilation on nuclei are treated down to essentially zero enrgy. Details of this event generators can be found elsewhere. This implementation provides the capability of full theoretically consistent modeling of exclusive distributions of secondary particles, spallation, fission and fragmentation products.

8.2 DPMJET

8.3 Event Generator Output

Section 5.2 explained the use of the MARS stand-alone event generator. Here the output produced by that running mode is described in more detail. The output files produced are `TMA.OUT`, `TDNDX.OUT`,

The `TMA` file always contains a table with the heading `AVERAGE MULTIPLICITY AND ENERGY`, with columns labeled by produced particle type, and several rows. The rows are defined as

`BLACK`	I haven't a clue
`GREY`	likewise
`SHOWER`	enter appropriate descriptions here
`FORW,CS>0`	
`X>0,BT>0.2`	
`X<0,BT0.2`	
`TOTAL`	
`E(X>0,BT>0.2)`	
`E(X<0,BT>0.2)`	
`ETOT`	
`SUBCUT NO.`	
`SUBCUT E`	
`ELAE,EBIND`	
`YIELD(pi+ ..)`	
`YIELD(pi- ..)`	
`SUM`	
`EG,EXGAM,EXWGA`	

If the user set `IDNDX=2` in routine MARS1400, then the `TMA` file also contains a series of seven tables. Each of the seven tables is labeled by `J=1`, `J=2`, etc., where the value of the index identifies the produced particle type; these are the same seven listed at the top of the output file. Each table has the structure

```
DEGR=0.0     5.00       10.00      15.00      20.00      30.00      60.00      90.00     135.00     180.00

EE(GEV)                                                D2N/DEDO                                                qDN/DE
5.000      7.49E-02   8.06E-02   9.04E-02   8.61E-02   1.77E-01   4.30E-01   3.31E-01   2.03E-01   9.10E-02   3.13E-01
10.000     3.64E-02   2.47E-02   4.10E-03   0.00E+00   0.00E+00   0.00E+00   0.00E+00   0.00E+00   0.00E+00   1.30E-02
```

The tables hold the production cross section $d^2N/dEd\Omega$ for the given particle type, as used by the generator, with the columns being bins in degrees, and the rows being bins in energy. The last column is dN/dE. The number of rows which appears is determined by the value of `NEVBIN` set in routine MARS1400. If instead the user set `IDNDX=3` in routine MARS1400, then the `TMA` file contains a different series of seven tables, where the columns are Ed^3N/d^3P in bins of P_t and the rows are bins in X_f. The last column is dN/dX

9 MAD-MARS Beam Line Builder

The MAD [101] lattice description language has become the *lingua franca* of computational accelerator physics. Any new developments in accelerator physics computational codes and libraries should have the requirements to read and understand lattice descriptions written in MAD. The ideas and modules of Ref. [102] are used in a new interface, which is able to read, parse, and store in memory MAD lattice descriptions, with the ability to generate an output file which translates those descriptions for input to MARS, and can also be used as input to other tracking and CAD applications.

The interface system—*MAD-MARS Beam Line Builder* (MMBLB)—reads a MAD lattice file and puts the elements in the same order into MARS geometry. Each element is assigned six functions which provide information about the element type/name, geometry, materials, field, volume and initialization. The user specifies the element type and an optional element name. If no name is specified, the element is considered to be a generic one. A building algorithm first tries to match the type/name pair and then substitute a generic element if needed. Once an element is described, it is registered with the system and its name is binded with the respective geometry, materials, volume and field descriptions. For each region search during tracking, MMBLB finds the corresponding type/name pair and calls its appropriate functions. MMBLB calculates a local rotation matrix $\mathcal{R}_i$ and a local translation vector $\mathcal{L}_i$. Then a global rotation matrix $\mathcal{M}_i$ and a position $\mathcal{P}_i$ are calculated and stored for each element

$$\mathcal{M}_i = \mathcal{M}_{i-1} \times \mathcal{R}_i, \quad \mathcal{M}_0 = \mathcal{U}$$
$$\mathcal{P}_i = \mathcal{M}_{i-1} \times \mathcal{L}_i + \mathcal{P}_{i-1}$$

where $\mathcal{U}$ is the unit matrix. $\mathcal{R}_i = \mathcal{U}$ for all elements, except RBEND and SBEND.

The newest version of MMBLB was substantially enhanced and is well described in the User's Guide [14]. The set of supported element types includes now almost all the elements supported by MAD. An arbitrary number of beam lines – arbitrary positioned and oriented – can be put in a MARS15 model. More sophisticated algorithms and new data structures enable more efficient searches through the beam line geometry. Tunnel geometry can now follow the beam line or be described independently of it.

For more details and use of MMBLB the user must read Ref. [14].

10 Histogram Input and Output Files

In many cases it is worthwhile to initialize the HBOOK package to use the output file MARS.HBOOK for interactive analysis with the MARS-GUI or PAW systems. All the powerful features of MARS-GUI are described in Section 12, while those of PAW are described in [103]. Both can be used for comprehensive physics analysis of the run session.

Histograms can be requested by the appropriate cards in `mars.inp` and `xyzhis.inp`. **At least one of the non-zero cards `NOBL`, `NSUR` or `NHBK` must be presented in `mars.inp` for that.** Then there will be an output file, `mars.hbook`. The MARS code is linked to the CERN library and uses the `hbook` package [104] to produce the `mars.hbook` file. There are several methods to use which each generate different histogram sets. Some histogram sets are "built-in": MARS defines and fills the set of histograms and the user cannot change the bin specifications or contents, only specify the geometric area the histograms cover. Other sets of histograms or ntuples are completely under the users control, and require the use of certain user-subroutines. All histogram sets, however, are by default written to the same `mars.hbook` file. All these histogram sets are described further below.

By default MARS sorts histograms by following classes:

- histogram type (vertex, fluence, energy deposition and energy spectrum);
- particle class (hadron, electromagnetic and muon);
- charge (neutrals, charged and total).

The main program allocates the dynamic memory for HBOOK and gives control to the whole system. Histogram set up and entry routines can be re-defined by the user according to his/her specific needs.

10.1 Built-in Volume Histograms

These histograms are filled for the whole system or for the `NOB` special regions defined in the MARS.INP file, and initialized by presence of non-zero `NOBL` and `RZOB` cards there. Residual dose histogram is normalized to $AINT(p/s)$ and is scored for the irradiation `TIRH` and cooling `TICH` times (days) defined on the `HTIR` card (default is 30 and 1 days, respectively). The histogram list in the file MARS.HBOOK with their IDs and functional contents is obtained with the PAW command `hi/list`. The list is self-explainatory. Default histogram IDs are shown in Table 10.

Table 10: Default histogram ID at `NOB=1`.

Particle	Vertex	Fluence	Energy Dep.	Spectrum
n	1	101		301
$h^{\pm}$ (w/o p)	2	102		302
p	3	103	203	303
γ		104		304
$e^{\pm}$		105		305
Total EM	6	106	206	
$\mu^{\pm} fromEMS$				307
$\mu^{\pm} total$	8	108	208	308

ALL HISTOGRAMS DIVIDED BY VOLUME, SURFACE AREA, OR/AND DELTA-E.

HISTOGRAM TYPE:

IHTYP = 1 - VERTEX, (STAR/CM3) 1<ID<100

IHTYP = 2 - FLUENCE AT STEP, (1/CM2) 101<ID<200

IHTYP = 3 - ENERGY DEPOSITION, (GEV/G) 201<ID<300

IHTYP = 4 - ENERGY SPECTRUM, (1/CM2/DEL) 301<ID<400

IHTYP = 5 - SURFACE ENERGY SPECTRA, (1/CM2/DEL) 401<ID<500

IHTYP = 6 - SURFACE TIME SPECTRA, (1) 501<ID<600

PARTICLE INTERACTION CLASS:

ICL = 1 - HADRON

ICL = 2 - ELECTROMAGNETIC

ICL = 3 - MUON

CHARGE CONTROL:

NSG = 0 - NEUTRONS, PHOTONS

NSG = 1 - CHARGED PARTICLES

NSG = 2 - TOTAL

VOLUMETRIC:

HISTOGRAM ID = NU1+NSG+3*(ICL-1)+100*(IHTYP-1)+1000*(NRE-1)

10.2 Built-in Surface Histograms

These histograms are filled for the plane and cylindrical surface detectors for the `NSURF` surfaces defined in the MARS.INP file, and initialized by presence of non-zero `NSUR` and `RZTS` cards there. There is also a possibility to write down parameters of particles crossing these surfaces into the files defined on the `RZTS` cards, with formats and various filters of the `WRTSUR` routine.

SURFACE:

HISTOGRAM ID = NU1+NSG+3*(ICL-1)+100*(IHTYP-1)+1000*(NSUR-1)

NU1=1 (DEFAULT)

PARTICLE SPECTRA HISTOGRAMMING IN 80 BINS: 75 LOG + 5 LIN

NEUTRONS: 28 (1.E-11 - 0.0145) + 52 (0.0145 - E0) GEV

OTHERS: 80 (5.E-4 - E0) GEV

NHSPE=0 - dN/dE, DIVIDED BY DEL=DELTA(E) IN (1/CM2/GEV)

E*dN/dE, DIVIDED BY DEL=DELTA(LOG10(E))

FOR NEUTRONS IF IND(14)=T IN (1/CM2)

NHSPE=1 - E*dN/dE, DIVIDED BY DEL=DELTA(LOG10(E))

FOR ALL PARTICLES IN (1/CM2)

SURFACE CROSSING ESTIMATOR (IHTYP=5, 6):

SURFACE TIME SPECTRA IN (TMIN-TMAX) INTERVAL (sec)

HISTOGRAMS GENERATED ARE UNIFORM IN 80 BINS (in nsec!)

Table 11: Example of surfaces in `MARS.INP`.

`NSUR`	11			
`RZTS`	6.15	6.15	-200.	200.
	200.	200.	-450.	450. 0.
	250.	250.	-450.	450. 0.
	6.15	60.	450.	450. 0.
	60.	200.	450.	450. 81.
	60.	60.	-200.	200. 0.
	230.	230.	-450.	450. 0.
	300.	300.	-450.	450. 0.
	400.	400.	-450.	450. 0.
	450.	450.	-450.	450. 0.
	470.	470.	-450.	450. 0.

10.3 Other Built-in Histograms

There are two additional groups of the built-in histograms:

- Histograms filled for the "global" energy deposition distributions initialized by the `NHBK` and `HBKE` cards for the entire system or for up to 5 materials specified on the `NHBK` card.
- Ntuple number 16 contains the normalization factors used for each histogram, unity or `AINT` which can be re-defined on-line in a GUI session.

10.4 XYZ-Boxes or Arbitrary Mesh Tallies

Flexible histograming – called `XYZ` – is implemented in boxes arbitrary positioned in a 3D-system modeled. This histograming is totally independent of geometry description, both type (standard, non-standard, extended and MCNP-type) and details, including materials distribution. By default, the macro box (detector) axes are parallel to those of the mother volume, and can be rotated using `MARS2BML` and `BML2MARS` routines.

The histograms can overlap, contrary to the volume (RZ) and surface histograming. Arbitrary voxels (the smallest distinguishable box-shaped part of a three-dimensional space) can be defined by individual binning NX, NY, NZ in each macro box. Arbitrary cut-off energies can be defined for each flux-type histogram. Histogram ID reserved: $701 < ID < 1000$. These histograms are normalized to $AINT(p/s)$, contrary to all previous histograms and most values in `MARS.OUT` and `MTUPLE` which are normalized to one incident particle. Residual dose histogram DRE is scored for the irradiation `TIRH` and cooling `TICH` times (days) defined on the `HTIR` card (default is 30 and 1 days, respectively). Up to 35 different histogram types can be defined in a run as described in Table 12. First 32 types are two-dimensional and can be filled for particle energies above certain cutoffs, *e.g.*, $FLN > 0.03$, $DET > 0.001$, $DRE > 0.02$, etc. This cutoff technique is not applicable to the gas production histograms, HYD, HEL and TRI. Defaults are 0.03 GeV for star density and energy transport cutoff energies defind on the `ENRG` card. Last 6 types are one-dimensional energy spectra. **Note that the DPA-type histograms are calculated only if `IND(17)=T`. DET through DEE histograms (prompt dose) are calculated in the course of Monte Carlo from**

energy-dependent fluxes and flux-to-dose conversion coefficients; they can have non-zero values in vacuum assuming that a tissue-equivalent phantom - used for definition of flux-to-dose conversion coefficients - is placed there. Contrary, the absorbed dose DAB and power density type histograms PDT through PDE are calculated in the course of Monte Carlo directly from energy deposited on a step in a non-vacuum region.

Table 12: Histogram types in XYZ-histograming. DLT is currently not activated.

Codename	Value	Units
STA	star density	$cm^{-3}s^{-1}$
DRE	residual dose on contact	mSv/hr
FLT	total flux of hadrons	$cm^{-2}s^{-1}$
FLP	flux of protons	$cm^{-2}s^{-1}$
FLN	flux of neutrons	$cm^{-2}s^{-1}$
FLK	flux of charged pions/kaons	$cm^{-2}s^{-1}$
FLM	flux of muons	$cm^{-2}s^{-1}$
FLG	flux of photons	$cm^{-2}s^{-1}$
FLE	flux of e- and e+	$cm^{-2}s^{-1}$
DAB	absorbed dose	Gy/yr at 2×10^7 s/yr
DPA	DPA total	DPA/yr at 2×10^7 s/yr
DPH	DPA by NIEL hadrons and muons	DPA/yr at 2×10^7 s/yr
DPN	DPA by neutrons at E<14 MeV	DPA/yr at 2×10^7 s/yr
DPE	DPA by electrons and photons	DPA/yr at 2×10^7 s/yr
HYD	Hydrogen gas capture	$cm^{-3}s^{-1}$
HEL	Helium gas capture	$cm^{-3}s^{-1}$
TRI	Tritium gas capture	$cm^{-3}s^{-1}$
DET	prompt dose equivalent, total	mSv/hr
DEP	prompt dose equivalent, protons	mSv/hr
DEN	prompt dose equivalent, neutrons	mSv/hr
DEK	prompt dose equivalent, ch. pion/K	mSv/hr
DEM	prompt dose equivalent, muons	mSv/hr
DEG	prompt dose equivalent, photon	mSv/hr
DEE	prompt dose equivalent, e- and e+	mSv/hr
PDT	power density, total	mW/g or Gy/s
PDP	power density, protons	mW/g or Gy/s
PDN	power density, neutrons	mW/g or Gy/s
PDK	power density, ch. pion/kaons	mW/g or Gy/s
PDM	power density, muons	mW/g or Gy/s
PDG	power density, photons	mW/g or Gy/s
PDE	power density, e- and e+	mW/g or Gy/s
DLT	instantaneous temperature rise	degC per AINT (ppp)
SPN	neutron energy spectrum	GeV^{-1} cm^{-2} s^{-1}
SPP	proton energy spectrum	GeV^{-1} cm^{-2} s^{-1}
SPK	pion/kaon energy spectrum	GeV^{-1} cm^{-2} s^{-1}
SPM	muon energy spectrum	GeV^{-1} cm^{-2} s^{-1}
SPG	photon energy spectrum	GeV^{-1} cm^{-2} s^{-1}
SPE	e+e- energy spectrum	GeV^{-1} cm^{-2} s^{-1}

In any run, the total number of histograms can not exceed 300, i.e. a sum over all histogram types in all macro boxes (detectors) must be less than 300. Maximum number of voxels in any macro box (bins in each

histogram) can not exceed 10^5. To save CPU-time, keep the number of macro boxes (detectors), histogram types and voxels (bins) as small as possible!

Histograms are defined in an input file `XYZHIS.INP`, where the following conventions are used:

1. The first line is a textual problem description.
2. The last line, `STOP` or `stop`, terminates the `XYZHIS.INP` data card list, i.e. everything after this line is ignored.
3. Blank lines can be inserted between data cards, for file organizational purposes.
4. Comments are allowed at the line end starting from an exclamation sign.
5. The data is unformatted and separated either by blank space or by one of the following delimiters ", ; : &".
6. A macro box (detector) is defined by a XYZ-card of the following format:

   $XYZ\ X1\ X2\ Y1\ Y2\ Z1\ Z2\ NX\ NY\ NZ\ TEXT,$

   where XYZ or xyz is a card ID;
   $X1, X2, Y1, Y2, Z1, Z2$ are the minimum and maximum Cartesian coordinates defining the box;
   NX, NY, NZ are the number of bins along each axis;
   $TEXT$ is a word describing the box (optional).
7. Macro box (detector) categories are

   $nxyztp = 1 -> X - Y$ in a Δz slice ($x = v, y = h$), defined by $NZ = 1$
   $nxyztp = 2 -> X - Z$ in a Δy slice ($x = v, z = h$), defined by $NY = 1$
   $nxyztp = 3 -> Y - Z$ in a Δx slice ($y = v, z = h$), defined by $NX = 1$
   $nxyztp = 4 ->$ Energy spectra in the macro box, defined by $NX = NY = NZ = 1$
8. A card following a XYZ-card contains a list of histogram types for this macro box: any combination of the above 38 types. An arbitrary cut-off energy can be defined for each 2-D histogram (except HYD, HEL and TRI as mentioned above) using the following syntax $FLN > 0.03$, which means that neutron flux in this particular histogram is scored above 0.03 GeV, not above the default threshold or that defined in the `MARS.INP` file.

A list of histograms to be filled in the run is generated at the completion of the initialization stage and shows up in an `XYZHIS.TAB` file which contains all attributes for each histogram. Here is the file created with the above `XYZHIS.INP`.

Here is an example of a `XYZHIS.INP` file:

```
 XYZ histo test 01-May-2007

xyz -20. 20. -0.4 0.4 0. 50. 40 1 50 XZ-scan_at_|y|<0.4
STA DRE FLT>0.02 DET DEN>0.1 DAB PDT

xyz -0.4 0.4  -10. 10.  0. 50. 1 20 50 YZ-scan_at_|x|<0.4_fluxes
STA FLT>0.02 DET DEG>0.05 DAB>0.02 FLP FLE DPA

xyz -5. 5. -2. 8. 12. 13. 10 5 1 XY-scan_in_a_z=12-13_cm_slice
FLT DPA

xyz -5. 5. -5. 5. 12. 13. 10 5 1 ! Muon flux profile at shower max
FLM
```

```
xyz 0. 5. 2. 7. 10. 15. 1 1 1 Spectrum-1_in_a_5x5x5_cm_cube
SPP SPN SPK SPM SPG SPE

!xyz -5. 5. -5. 5.. 10. 20. 1 1 1 ! Muon-spectrum
!SPM

stop
```

10.5 User-Defined Histograms and Ntuples

Arbitrary one- and two-dimensional histograms and ntuples can additionally be booked in a user routine `MHSETU` and filled in `MFILL`.

11 Output of the Simulation

MARS produces many output files. The main output file, `MARS.OUT`, is always produced, and contains startup initialization information in addition to results. Other output files are produced only when certain zone types are set up, or when certain histogram options are requested, or when certain flags are set in the `MARS.INP` file. This section covers all these different output options, how to request them, modify them where possible, and what the contents are.

`MARS.OUT` and several other output files are "flat" text files which can be opened and examined with any text editor. They contain tables of data which are typically, but with some exceptions, arranged by the MARS zone numbers. These are described further below in Section 11.1.

There are a few other specialized output files, such as those generated by use of the **`TAPE`** card in the `MARS.INP` file, discussed in Section 11.2. While these are text files which can be opened and examined, they are not intended to be used in this way, but instead as input files for other programs or for multi-stage running of MARS. For example, the `TRACK.PLOT` file is used in conjunction with the GUI visualization tool, to examine samples of generated events. If the user wishes to extract particle data for his own specialized purpose, then certain user subroutines can be used to create a custom output file – see Section 5 for more information.

All the filenames used here are the MARS default names. These file names can be changed in the main calling routine `marsmain.f`; see Section 5.1.

11.1 Data Table Output Files

The files `VOLMC.NON`, `MARS.OUT`, `MTUPLE`, `MTUPLE.NON`, `MTUPLE.EXG`, and `MTUPLE.MCNP` are described in this Section. These files report results for different zone types. The `MARS.OUT` file, Section 11.1.2, lists all the tables of results for Standard zones. If the user requests "expanded output" by setting the `MARS.INP` card **`INDX`** with flag `1=T`, then the `MTUPLE` file is also produced (no suffix on this default filename). The `MTUPLE` file, Section 11.1.3, contains results for Standard zones and duplicates some of the information in the `MARS.OUT` file, but the results are arranged in a single large table rather than several separate ones. When non-Standard zones are present, their results are printed to the `MTUPLE.NON` output file, Section 11.1.4; these include zones from imported geometry files. If Extended Geometry zones are used, their output appears in the `MTUPLE.EXG` file, Section 11.1.5. If using the MARS-MCNP mode *and* also using MCNP geometry cards to create some of the geometric zones, then the results for these zones is reported in the `MTUPLE.MCNP` file, Section 11.1.6.

Keep in mind that if the users model description uses more than one type of zone (Standard and non-Standard, Standard and Extended, or any other combination) then the results will be spread around among the different files. If the results in an "area of interest" are split between files because of the use of different zone types, then the user may wish to rethink his model's geometry description in order to more efficiently organize the results.

The format of data tables in these files is defined within non-user code, and cannot be changed by the user. The tabular format of these files is intended to make it easy to look up the results for certain key zones of interest. The rows and columns can also be cut-and-pasted into other software plotting packages such as, GNUPLOT, XMGR, KALEIDAGRAPH and EXCEL. Copied data can also be read into PAW, although in most cases if PAW is being used it is simpler to use the histogram options to create such a file directly.

Each of the output files has it's own sub-Section, with each table of data defined.

11.1.1 The `VOLMC.NON` File

The `VOLMC.NON` file is a "pre-run" output file. The file is created only when MARS is run in "volume MC" mode; see Section 4.2.2 for the switch settings to use this mode. The file lists Zones, by their MARS zone number, and gives the volume for each of these zones as measured by the program. These volume values are cut and pasted into the user subroutine `VFAN`; see Section 5.6 for a more detailed discussion of the process. The contents of a `VOLMC.NON` file for a simple beam dump geometry is given here – In this example, there is a single "container" Standard zone, in which all the other non-Standard zones are embedded; this is zone #1. The Standard zone's volume is calculated along with the other zones, and the message explains that the container's actual volume is likely not to be that of a simple cylinder, due to the presence of the embedded zones. Following the text message are the volumes for the non-Standard zones, stated as `VOLUME( nnn) = 0.rrrE+nn`. The text after the `!` gives the error on the volume calculation, the assigned non-Standard or Extended Geometry zone number, and the zone name. The entire statement can be pasted directly into the `VFAN` user subroutine.

11.1.2 The `MARS.OUT` File

The `MARS.OUT` file is always produced, even when MARS is run in GUI mode or in "volume MC" mode where, in both cases, no particle interaction simulations are performed. This is because the `MARS.OUT` file records startup and initialization information in addition to the simulation results for Standard zones.

Initialization Information

At startup, Mars echos many of the settings in the MARS.INP file to the MARS.OUT file. The user should always take a look at these to check that the program settings were as expected.

The first few lines – give the Mars code version, the date and time the program was started, and the users title exactly as taken from the first line of the MARS.INP file.

Next the settings of the cards from the MARS.INP file are echoed, with the first few lines as follows – There is an entry for each card, even those which have been left at their default value and are not listed by the user in the MARS.INP file. The materials declared in the MARS.INP file are not merely echoed but appear along with the energy thresholds and tracking step sizes that will be applied to each material, and additional information such as the radiation length. The global thresholds and step sizes appear at the start of the materials list, labeled as the material "global vacuuum" – At the end of the materials data list is a table listing each element used by the model. Following that are tables of calculated material dependent quantities : After the materials, the echoing of the MARS.INP cards continues, and concludes.

Additional tables of calculated material-dependent parameters can be requested by setting the value of `IPRINM` on the **`NEVT`** card $= 1$. Tables of hadron and photon nuclear cross-sections and of dE/dx are printed out for each material and as a function of energy.

Output Tables

The first output table in the MARS.OUT file is a mapping of the MARS Standard zone numbers to their r, z and (if used) ϕ locations. Nearly all subsequent data tables list results by their Standard zone number. So, for example, the following lines in a MARS.INP file set up a set of Standard zones: This is a z-sandwich geometry `(INDX(2)=T)` so the material content of the zones is defined on the **`ZSEC`** card. The resulting map table in the MARS.OUT file is: The integer number is the Standard zone number, and the number in parentheses is the index for the material in that zone. For example, for $25.0 < z < 35.0$ and $10.0 < r <$

20.0, the Standard zone number is 14 and it contains material number 2. Notice that the r and z boundaries listed in the map table are not evenly distributed, but correspond to the boundaries of the declared Standard zones.

The tables which report results by Standard zone number are all laid out in a similar manner. The table has ten columns, with the columns numbered 1:10; the rows are labeled 10, 20, 30, etc. The data being reported in the table is listed in order by zone, labeled also "Region Number", starting at the first row through 10 columns, then to the second row, third, and so on for as many rows are required. So, for example, imagine if the model was a target, simplified to a group of Standard zone cylinders. One wishes to know the location which will have the largest energy deposition. Go to the table which lists total energy deposition by zone, and look for the largest value; identify, based on the row and column, which Standard zone number the value is in. Then return to the Standard zone map table to determine the r and z value which that zone number corresponds to.

Many results have a statistical error reported with them. Please read Section 3.8 for a brief discussion on how one should use these errors as a guide to the quality of the results. All results are normalized to one incident particle (i.e. to one event), except for temperature rise and residual dose, which are normalized to the total incidents as declared by the value of `AINT` on the **`VARS`** card.

The following paragraphs itemize each data table in the order in which they appear in the `MARS.OUT` file. The table label is highlighted, and the values it contains are defined. Not all the listed tables may be present. If `INDX 1=T` more tables will be present than otherwise. If "special region" histograms were requested, using the **`NOBL`** card, then additional tables of results will be present for the declared "special regions".

`LONGITUDINALLY INTEGRATED (ZMIN-ZMAX) VALUES`

This table is not arranged by zone number, but, as the title implies, has combined the results along the z-axis. Each column holds the summed results for all z within the r values given at the tops of the columns. Each type of integrated result is listed for those r on each row. The first row, `SCHAR`, is the charged star density; `STOT` is the total star density; `FLTOT` is the total hadron flux; `ELEN` is energy deposited by low energy neutrons, those below the hadron energy threshold as given by `EM` on the **`ENRG`** card; `ELCH` is energy deposited by low-energy charged hadrons from nuclear de-excitation processes; `EEMS` is energy deposited by electro-magnetic showers; `EDEX` is energy deposited through ionization losses from charged hadrons and muons; `ETOT` is the total energy deposition combined from all these sources in the given r ranges.

`DIRECT ENERGY DEPOSITION, GEV`

This table is not arranged by zone number, but has combined the results across r bins. Each row lists various energy deposition results for all r between the z values given in the first two columns. The third column, `LEN`, is energy deposited by low energy neutrons, those below the hadron energy threshold as given by `EM` on the **`ENRG`** card; the next column, `LCH`, is energy deposited by low-energy charged hadrons, also below energy `EM`, from nuclear de-excitation processes; `EMS` is energy deposited by electro-magnetic showers; `DEX` is the energy deposited through ionization losses from all charged hadrons and muons. The `TOTAL IN SLAB` adds all the contributions for that z row. `TOTAL IN BLOCK` sums each z row with the one before, making a cumulative sum. A last row, below the table, gives the sums, for all z, of the values in each column.

`TOTAL DOSE EQUIVALENT (ICRP51->MAXIMUM OR 1-CM), mSv PER 1 INC. PARTICLE`

This is the first of many tables arranged by Standard zone number, called "Region Number" in the file. The table presents, as labeled, the dose equivalent that tissue-like material would experience

if it were located here, instead of the material which is actually present. This is an estimate of the potential "beam-on" or prompt dose, not a residual rate. Each row contains two values, the result for that zone, and the statistical error on that result.

STAR DENSITY, NO./CM3 P.P.**

The table presents, arranged by zone number, the star density within each zone. Each row holds three values: the star density due to the charged hadrons (primarily protons, πs and Ks), the total star density, and the statistical error on the total star density. The standard definition of a star is a hadron with kinetic energy greater than 0.03 GeV; in MARS this is the default value for the star production threshold, EPSTAM in the **ENRG** card. If this threshold is changed, then the results reported in this table will likewise change, and will no longer correspond to the standard definition of a star.

DPA, DISPLACEMENTS PER ATOM P.P

The table presents, arranged by zone number, the nuclear displacements per atom within each zone, which can be used to assist in estimates of radiation damage to the material located in this zone. Elastic and inelastic nuclear interactions as well as electromagnetic interactions contribute to DPA in this version of the code.

HYDROGEN GAS PRODUCTION (1/cm3) P.P**

The table presents, arranged by zone number, the density of hydrogen gas production within each zone, which can be used to assist in estimates of radiation damage to the material located in this zone.

HELIUM GAS PRODUCTION (1/cm3) P.P**

The table presents, arranged by zone number, the density of helium gas production within each zone, which can be used to assist in estimates of radiation damage to the material located in this zone.

TRITIUM GAS PRODUCTION (1/cm3) P.P**

The table presents, arranged by zone number, the density of tritium gas production within each zone, which can be used in a corresponding radiation safety analysis.

TRACK-LENGTH HADRON FLUENCE IN HIGH-ENERGY SECTOR 1/CM2**

The table presents, arranged by zone number, the total hadron flux in each zone, as calculated from the total track-lengths of these particles passing through this zone. The hadrons used for the tabulation have a kinetic energy greater than the value of EM, the hadron threshold energy, in the **ENRG** card; the default threshold value is 0.0145 GeV. The statistical errors are also given.

CHARGED HADRON FLUENCE AT E>ETH, HADRONS/CM2

The table presents, arranged by zone number, the total hadron flux in each zone, for charged hadrons with a kinetic energy greater than the value of the threshold EMCHR set in the **ENRG** card, default value 0.0002 GeV; *OR* with a kinetic energy greater than EM if EM has been set to greater than it's 0.0145 GeV default value. The statistical errors are also given.

NON-NEUTRON NEUTRAL HADRON FLUENCE AT E>ETH, HADRONS/CM2

The table presents, arranged by zone number, the neutral hadron flux, but not including neutrons, in each zone, for those particles with a kinetic energy greater than the value of the threshold EM set in the **ENRG** card; the default threshold value is 0.0145 GeV. The statistical errors are also given.

HEAVY ION FLUENCE AT E>ETH, HADRONS/CM2

The table presents, arranged by zone number, the flux of heavy ions in each zone, for ions with a kinetic energy greater than the value of the threshold EMCHR set in the **ENRG** card, default value 0.0002 GeV; *OR* with a kinetic energy greater than EM if EM has been set to greater than it's 0.0145 GeV default value. The statistical errors are also given.

ELECTRON FLUENCE AT E>ETE, ELECTRONS/CM2

The table presents, arranged by zone number, the flux of electrons and positrons in each zone, for those particles with a kinetic energy greater than the value of the threshold EMIEL set in the **ENRG** card, default value 0.0002 GeV; *OR* with a kinetic energy greater than EM if EM has been set to greater than it's 0.0145 GeV default value. The statistical errors are also given.

GAMMA FLUENCE AT E>ETG, GAMMAS/CM2

The table presents, arranged by zone number, the flux of gammas in each zone, for those particles with a kinetic energy greater than the value of the threshold EMIGA set in the **ENRG** card, default value 0.0002 GeV; *OR* with a kinetic energy greater than EM if EM has been set to greater than it's 0.0145 GeV default value. The statistical errors are also given.

MUON FLUENCE AT E>ETM, MUONS/CM2

The table presents, arranged by zone number, the flux of muons in each zone, for those particles with a kinetic energy greater than the value of the threshold EMCHR set in the **ENRG** card, default value 0.0002 GeV; *OR* with a kinetic energy greater than EM if EM has been set to greater than it's 0.0145 GeV default value. The statistical errors are also given.

TOTAL STAR NUMBER

Not a table of values, but a single value summed over all Standard zones.

======= LATERAL INTEGRATED =======

A table of several values, all of which are summed across all Standard zone r values for the given z range. The first two columns give the lower and upper z boundaries for each z range. The third column, S/CM, is the integrated stars. Note the units - stars$/cm^2$ for each zone in the z range have been integrated over r, so that the units are now stars$/cm$. The fourth column, FTOT, is the total hadron flux, which includes neutrons above the threshold EM, default value 0.0145 GeV; here the flux values from each zone have been summed, so the units are $/cm^2$. The fifth column, F1, is charged hadron flux for hadrons with energy greater than 0.0001 GeV. The sixth column, F2, is charged hadron flux for hadrons with energy greater than 0.02 GeV. The seventh column, F3, is the $e^{\pm}$ flux for $e^{\pm}$ with energy greater than 0.00010 GeV. The eighth column, F4, is the $e^{\pm}$ flux for $e^{\pm}$ with energy greater than 0.02 GeV. The ninth column, F5, is the muon flux for muons with energy greater than 0.00010 GeV. The tenth column, F6, is the muon flux for muons with energy greater than 0.02 GeV.

ENERGY DEPOSITION DUE TO L.E.NEUTRONS, GEV/G P.P.

The table presents, arranged by Standard zone number, the energy deposited in each zone by low energy neutrons, those below the EM value set in the **ENRG** card; the default threshold is 14.5 MeV. The units are GeV per g per primary. Statistical errors are not given.

ENERGY DEPOSITION DUE TO L.E.CHARGED, GEV/G P.P.

The table presents, arranged by Standard zone number, the energy deposited in each zone by low energy charged hadrons, those below the `EM` value set in the **`ENRG`** card; the default threshold is 14.5 MeV. The units are GeV per g per primary. Statistical errors are not given.

`ENERGY DEPOSITION DUE TO EMS, GEV/G P.P.`

The table presents, arranged by Standard zone number, the energy deposited in each zone by electromagnetic showers; this includes the γs produced by π^0, which decay immediately at their production point. The units are GeV per g per primary. Statistical errors are not given.

`ENERGY DEPOSITION DUE TO CH.HADRONS AND MUONS, GEV/G P.P.`

The table presents, arranged by Standard zone number, the energy deposited in each zone by charged hadrons and muons with energy above the `EM` value set in the **`ENRG`** card; the default threshold is 14.5 MeV. The units are GeV per g per primary. Statistical errors are not given.

`DIRECT ENERGY DEPOSITION, GEV/G P.P.`

The table presents, arranged by Standard zone number, the total energy deposited in each zone by all contributing particles. The units are GeV per g per primary. Statistical errors are given.

`DOSE EQUIVALENT (mSv per 1 inc.  particle)`

The table presents, arranged by Standard zone number, the radiation dose which a tissue-like material would experience if it were present in this zone instead of the material which is actually present. This is not a residual rate, but an estimate of equivalent "beam-on" exposure. The units are mSv per primary. No statistical errors are given.

`TEMPERATURE RISE (K) AT TO= 3.000E+02 PER 1.000E+12 PPP`

The table presents, arranged by Standard zone number, the instantaeous temperature rise the material in this zone would experience, in Kelvin, due to the impact of beam. It is assumed that the material is initially at standard room temperature, but this can be changed by setting the `TEMPO` value in the **`VARS`** card. The temperature rise is calculated on a per pulse basis, rather than per primary, where a pulse contains by default 10^12 primaries; this pulse-size value can be changed by setting the `AINT` value in the **`VARS`** card. No statistical errors are given.

`LEAKAGE: BACKWARD FORWARD SIDE`

This table, and the values following it, present information on particles which have passed out of the model's geometrical limits. The table is arranged in 3 columns labeled by the indicated directions. The first row, `TOTAL SURFACE CHARGED CURRENT` is the number of charged hadrons; the second row, `TOTAL SURFACE NEUTRAL CURRENT` is the number of neutrons. This is followed by `HADRON LEAKAGE ENERGY, GEV`, a sum of the energy carried by these exiting particles in the indicated directions. This table is followed by single values for the leaked energy from various particle types, along with the total of all types. Finally, there is the `VISIBLE ENERGY BALANCE` which compares the total energy detected within the geometry to a projected total energy which is defined on the following line.

For geometries similar to beam dumps with thick shielding, one expects most or all of the energy to be contained within the model. For geometries which simulate a beamline, one expects a good fraction of the incoming beam energy to pass through. These leakage values help one determine if the model is behaving in an expected way, and help give indicators of how energy is being carried out of the model.

`HADRON LEAKAGE SPECTRA`

This table presents the energy spectra for different groups of hadrons which have leaked out of the model. The data are arranged in 3 main columns for the 3 labeled directions, `UPSTREAM DOWNSTREAM SIDE`. Each of these main columns holds in turn 3 sub-columns labeled `P,PBAR` for p and $\overline{p}$, `N,NBAR` for n and $\overline{pn}$, and `PI+/-,K+/-` for $\pi^{\pm}$ and $K^{\pm}$. The rows are bins in energy with the indicated values. Two spectra data entries are given for each row/column, dF/dE in units per GeV-cm^2 and $dF/d(lnE)$ in units per cm^2. At the end of the table are three additional results, combining the specified particle groups into a single value for each of the three leakage directions. The first result is `TOTAL (1/CM**2)`, the sum of the $dF/d(lnE)$ spectra; the second result is the mean energy `<E> (GEV)`; the third result is the dose equivalent `HADRON DEQ (mSv)` due to the leaked hadrons.

OTHER NON-LEPTON LEAKAGE SPECTRA

This table is similar in set-up to the one described above, but for different particle groupings. In this table, all charged hadrons are one particle group, all neutral hadrons, and all heavy ions. *Does this table include the particles from the above table, just in different combination? Or is it a summary of anything NOT included in the above table? In my sample output, this table is all zeros, while the above table has some data results.*

GRAPH: HADRON LEAKAGE SPECTRA

This table repeats the $dF/d(lnE)$ data from the `HADRON LEAKAGE SPECTRA` table, but reformatted so that the columns of data can easily be cut-and-pasted into an external graphics or spreadsheet program.

GRAPH: PHOTON-ELECTRON LEAKAGE SPECTRA

This table presents the energy spectra for photons and electrons/positrons which have leaked out of the model. The data are arranged in 3 main columns for the 3 labeled directions, `FACE BACK SIDE`. Each of these main columns holds in turn two sub-columns, one with the photon data and the other with the e^+/e^- data. Each row is an energy bin, with the bin value in the left-most column. The data is in units dF$/d$E $1/GeV - cm^2$. The table is formatted so that the columns of data can easily be cut-and-pasted into an external graphics or spreadsheet program. At the end of the table are three additional results, obtained by combining the photon and electron/positron spectra data. The first result, labeled `TOTAL (1/CM**2)`, the integral of the dF$/d$E spectra; the second result is the mean energy `<E> (GEV)` from each spectrum; the third result is the dose equivalent `DEQ (mSv)` due to the leaked particles.

GRAPH: HADRON SPECIAL REGIONS SPECTRA

The "special regions" are user-defined cylindrical volumes specified by the **NOBL** and **RZOB** cards in the `MARS.INP` file. There can be no more than three of these regions declared, and there is a column printed for each, filled with zeros if the region is not used. Declaring a NOBL region triggers the production of HBOOK histograms of a variety of results for the zones within the region; see Sec 10.1 for a definition of the histograms. This table holds the same data used to fill the histograms, although as in this case, split into the contributions from three particle types; each main column for a region holds three sub-columns labeled `P N PI`. The data is dF$/d$(lnE) in units hadrons per cm^2 and given in energy bins as labeled by the left-most column (note the energy bins are in MeV). The columns of data are arranged to be easily cut-and-pasted into an external graphics or spreadsheet program.

GRAPH: PHOTON-ELECTRON SPECIAL REGION SPECTRA

This table is similar to the one described above, being data accumulated in the volumes specified by the **NOBL** and **RZOB** cards. This table gives data for γ and $e^{\pm}$ in two sub-columns for each special

region. The data here is $d\mathrm{N}/d\mathrm{E}$ in units per GeV-cm^2, plotted in energy bins as labeled by the left-most column. The columns of data are arranged to be easily cut-and-pasted into an external graphics or spreadsheet program. At the end of the table are three additional results, summed or accumulated from the spectrum data. The first result, labeled `TOTAL (1/CM**2)`, the integral of the $d\mathrm{N}/d\mathrm{E}$ spectra; the second result is the mean energy `<E> (GEV)` from each spectrum; the third result is the dose equivalent `DEQ (mSv)` due to the particles passing through each volume region.

GRAPH: TOTAL DOSE EQUIVALENT

This table presents data values versus location in the model volume, ignoring zone and region boundaries. The data is instead mapped into 5 r bins and 10 z bins. The maximum r and z bin values describe the maximum extent of the model - the Standard zone "mother volume"; the bins are created by equal divisions between zero and the maximums. The left-most column gives the mean z coordinate in each bin; the top two rows give the lower and upper r range for each of those bins. The data in this table is the total dose equivalent in units of mSv per incident primary particle. The columns of bins and data are arranged to be easily cut-and-pasted into an external graphics or spreadsheet program.

GRAPH: TOTAL STAR DENSITY

This table presents data values versus location in the model volume, ignoring zone and region boundaries. The data is instead mapped into 5 r bins and 10 z bins. The maximum r and z bin values describe the maximum extent of the model - the Standard zone "mother volume"; the bins are created by equal divisions between zero and the maximums. The left-most column gives the mean z coordinate in each bin; the top two rows give the lower and upper r range for each of those bins. The data in this table is the total star density, in units of stars per cm^3 per incident primary particle. The columns of bins and data are arranged to be easily cut-and-pasted into an external graphics or spreadsheet program.

GRAPH: DIRECT ENERGY DEPOSITION

This table presents data values versus location in the model volume, ignoring zone and region boundaries. The data is instead mapped into 5 r bins and 10 z bins. The maximum r and z bin values describe the maximum extent of the model - the Standard zone "mother volume"; the bins are created by equal divisions between zero and the maximums. The left-most column gives the mean z coordinate in each bin; the top two rows give the lower and upper r range for each of those bins. The data in this table is the total direct energy deposited into the materials of the model, in units of GeV per gram per incident primary particle. The columns of bins and data are arranged to be easily cut-and-pasted into an external graphics or spreadsheet program.

GRAPH: TEMPERATURE RISE

This table presents data values versus location in the model volume, ignoring zone and region boundaries. The data is instead mapped into 5 r bins and 10 z bins. The maximum r and z bin values describe the maximum extent of the model - the Standard zone "mother volume"; the bins are created by equal divisions between zero and the maximums. The left-most column gives the mean z coordinate in each bin; the top two rows give the lower and upper r range for each of those bins. The data in this table is the temperature rise of the materials due to beam from a single pulse of 10^12 primaries, in units of K; it is assumed that the starting temperature of the materials is 300K unless the user changed this via the `TEMP0` value on the **`VARS`** card in the `MARS.INP` file. The columns of bins and data are arranged to be easily cut-and-pasted into an external graphics or spreadsheet program.

GRAPH: DOSE EQUIVALENT

This table presents data values versus location in the model volume, ignoring zone and region boundaries. The data is instead mapped into 5 r bins and 10 z bins. The maximum r and z bin values describe the maximum extent of the model - the Standard zone "mother volume"; the bins are created by equal divisions between zero and the maximums. The left-most column gives the mean z coordinate in each bin; the top two rows give the lower and upper r range for each of those bins. The data in this table is the tissue dose equivalent, in units of mSv per incident primary particle. The columns of bins and data are arranged to be easily cut-and-pasted into an external graphics or spreadsheet program.

GRAPH: TOTAL HADRON FLUENCE IN HIGH-ENERGY SECTOR

This table presents data values versus location in the model volume, ignoring zone and region boundaries. The data is instead mapped into 5 r bins and 10 z bins. The maximum r and z bin values describe the maximum extent of the model - the Standard zone "mother volume"; the bins are created by equal divisions between zero and the maximums. The left-most column gives the mean z coordinate in each bin; the top two rows give the lower and upper r range for each of those bins. The data in this table is the total hadron particle flux for hadrons with energy greater than the value of `EM`, the hadron threshold energy, in the **ENRG** card; the default threshold value is 0.0145 GeV. Units are hadrons per cm^2 per incident primary particle. The columns of bins and data are arranged to be easily cut-and-pasted into an external graphics or spreadsheet program.

******** LOW ENERGY NEUTRON SECTOR *******

This delimiter labels a group of tables which hold data on low energy neutrons, those with energy below ??. A few single data values are given here: the total neutron collisions per primary, the total energy leaked due to this class of particles, and the number of low energy neutron which exit the model volume.

TOTAL FLUENCE AT E < 14.5 MEV, N/CM2

The table presents, arranged by Standard zone number, the low energy neutron flux, in particles per cm^2. Each data value is accompanied by it's statistical error.

ENERGY DEPOSITION AT E < 14.5 MEV

The table presents, arranged by Standard zone number, the energy deposited by low energy neutrons, in units of GeV per gram. No statistical errors are given.

NEUTRON LEAKAGE SPECTRA PER CM2**

This table presents the energy spectra for the low energy neutrons which have leaked out of the model. The data are arranged in 3 main columns for the 3 labeled directions, `UPSTREAM DOWNSTREAM SIDE`. Each of these main columns holds in turn 3 sub-columns labeled `NEUTRONS` for ?? the overall spectrum, `N/UNIT LET` for ??, and `N/MEV` for ??. The rows are bins in energy with the indicated values. At the end of the table are three additional results, integrating the results to a single value for each of the three leakage directions. The first result is `NEUTRON FLUENCE(N/CM**2)`; the second result is the mean energy `<E> (GEV)`; the third result is the dose equivalent `NEUTRON DOSE EQIVAL (mSv)` due to the leaked neutrons.

GRAPH: L.E. NEUTRON LEAKAGE SPECTRA: dF/d(lnE) per cm2**

This table is similar to `NEUTRON LEAKAGE SPECTRA PER CM**2` table, but reformatted so that the columns of data can easily be cut-and-pasted into an external graphics or spreadsheet program.

SPECIAL REGIONS NEUTRON SPECTRA PER CM2**

The "special regions" are user-defined cylindrical volumes specified by the **NOBL** and **RZOB** cards in the `MARS.INP` file. There can be no more than three of these regions declared, and there is a column printed for each, filled with zeros if the region is not used. Declaring a NOBL region triggers the production of HBOOK histograms of a variety of results for the zones within the region; see Sec 10.1 for a definition of the histograms. This table holds the same data used to fill the histograms, although as in this case, split into three quantities; each main column for a region holds three sub-columns labeled `NEUTRONS N/UNIT LET N/MEV`. The data is given in energy bins as labeled by the left-most column (note the energy bins are in MeV). The columns of data are arranged to be easily cut-and-pasted into an external graphics or spreadsheet program. At the end of the table are three additional results for each special region, summed or accumulated from the spectrum data. The first result is `NEUTRON FLUENCE(N/CM**2)`; the second result is the mean energy `<E> (GEV)`; the third result is the dose equivalent `NEUTRON DOSE EQIVAL (mSv)`.

GRAPH: L.E. NEUTRON SPECIAL REGION SPECTRA: dF/d(lnE) (1/cm2)**

This table is similar to the one described above, being data accumulated in the volumes specified by the **NOBL** and **RZOB** cards. The data here is the low energy neutron spectrum $d\mathrm{N}/d\mathrm{E}$ in units per GeV-cm^2, plotted in energy bins as labeled by the left-most column. The columns of data are arranged to be easily cut-and-pasted into an external graphics or spreadsheet program.

GRAPH: LOW ENERGY NEUTRON FLUENCE (N/CM2 PER 1 PPP) AS A FUNCTION OF DEPTH(DOWN) AND RADIUS(ACROS) IN CM

This table presents data values versus location in the model volume, ignoring zone and region boundaries. The data is instead mapped into 5 r bins and 10 z bins. The maximum r and z bin values describe the maximum extent of the model - the Standard zone "mother volume"; the bins are created by equal divisions between zero and the maximums. The left-most column gives the mean z coordinate in each bin; the top two rows give the lower and upper r range for each of those bins. The data in this table is the low energy neutron flux, in units of neutrons per cm^2 per incident primary particle. The columns of bins and data are arranged to be easily cut-and-pasted into an external graphics or spreadsheet program.

GRAPH: TOTAL NEUTRON FLUENCE AT E > 1.000E-01 MEV (N/CM2 PER INC. PARTICLE) AS A FUNCTION OF DEPTH(DOWN) AND RADIUS(ACROS) IN CM

This table is similar to the one described above, but for all neutrons below the stated energy cutoff.

GRAPH: NEUTRON FLUENCE AT E > 0.1 MEV (N/CM2 PER INC.PARTICLE) AS A FUNCTION OF DEPTH(DOWN) AND RADIUS(ACROS) IN CM

This table appears to be identical to the one above.

GRAPH: NEUTRON FLUENCE AT E > 1 MEV (N/CM2 PER INC.PARTICLE) AS A FUNCTION OF DEPTH(DOWN) AND RADIUS(ACROS) IN CM

This table is similar to the one described above, but for all neutrons below the stated energy cutoff.

GRAPH: NEUTRON FLUENCE AT E > 1.450E+01 MEV (N/CM2 PER INC.PARTICLE) AS A FUNCTION OF DEPTH(DOWN) AND RADIUS(ACROS) IN CM

This table is similar to the ones described above, but for all neutrons below the stated energy cutoff.

30days/1day RESIDUAL DOSE RATE AT CONTACT (mSv/hr) FOR I= 1.0000E+12 P/SEC

The table presents, arranged by Standard zone number, the residual dose predicted at the surface of each Zone after a canonical exposure. The exposure used is 30 days with $1E^{12}$ primaries per second entering the model, followed by 1 day of "cool-down" decay. The residual estimates are valid only for relatively thick zones, those with a lateral thickness $\geq \lambda_{in}$, the Zone material's interaction length.

GRAPH: 30days/1day RESIDUAL DOSE RATE AT CONTACT (mSv/hr)

This table presents data values versus location in the model volume, ignoring zone and region boundaries. The data is instead mapped into 5 r bins and 10 z bins. The maximum r and z bin values describe the maximum extent of the model - the Standard zone "mother volume"; the bins are created by equal divisions between zero and the maximums. The left-most column gives the mean z coordinate in each bin; the top two rows give the lower and upper r range for each of those bins. The data is identical to that in the table described above, but re-binned.

MUON FLUENCE (MUON/CM2) AT E > 0.00020 GEV**

The table presents, arranged by Standard zone number, the muon flux in each Zone, for muons above the stated energy cut-off. Each main row holds four sub-rows: the first tabulating only μ^+, the second tabulating only μ^-, the third tabulating all muons, and the fourth being the statistical error on the total muon flux.

The table presents, arranged by Standard zone number, . No statistical errors are given.

The table presents, arranged by Standard zone number, . No statistical errors are given.

The table presents, arranged by Standard zone number, . No statistical errors are given.

The table presents, arranged by Standard zone number, . No statistical errors are given.

The table presents, arranged by Standard zone number, . No statistical errors are given.

The table presents, arranged by Standard zone number, . No statistical errors are given.

The table presents, arranged by Standard zone number, . No statistical errors are given.

The table presents, arranged by Standard zone number, . No statistical errors are given.

The table presents, arranged by Standard zone number, . No statistical errors are given.

The table presents, arranged by Standard zone number, . No statistical errors are given.

The table presents, arranged by Standard zone number, . No statistical errors are given.

The table presents, arranged by Standard zone number, . No statistical errors are given.

The table presents, arranged by Standard zone number, . No statistical errors are given.

1. Crude estimation of three-dimensional distribution of residual dose rate, in *Rad/hr*, after 30-day irradiation at the mean beam intensity `AINT` particles per sec and 1-day cooling. These are valid only for sufficiently thick systems, beyond a lateral thickness of $\geq \lambda_{in}$.
2. Three-dimensional distribution of the $relative$ statistical errors of star densities, fluxes and energy deposition densities.
3. Leakage data: number and energy of albedo hadrons, of punchthrough had-
   rons and of hadrons that escaped the sides of the system; leakage energy of low-energy neutrons and of electromagnetic showers; total leakage energy; energy balance.
4. Leakage energy spectra of different particles for the upstream plane, downstream plane and for the rest of the system.
5. If `NOB`$\geq$`1`, energy spectra of different particles in the pre-determined special regions.
6. Tables of particle spectra, of star density, of hadron, muon and low-energy fluence, of total energy deposition density, and of temperature rise, ready for use with graphics packages. The same tables are saved in separate .GRA files if activated in MAIN.

11.1.3 The `MTUPLE` File

`IND(1)=T` – ALWAYS. The program calculates and prints distributions of e^+e^- and photon fluxes and of energy deposition density ϵ and related values: dose equivalent, instantaneous temperature rise ΔT at given initial temperature T_0=`TEMPO` and number of particles per beam N_0=`AINT` (see `VARS`), and contact dose due to induced radioactivity at N_0 beam intensity. Other values discussed in Section 11.1.2 are calculated independently of the `IND(1)` meaning.

11.1.4 The `MTUPLE.NON` File

If the user defines geometrically complex insertions defined with REG1 routine as a supplement to the $standard$ and $extended$ geometries, additionally the program prints in a compact form most of the above quantities with corresponding statistical errors for all $non - standard$ regions.

11.1.5 The `MTUPLE.EXG` File

11.1.6 The `MTUPLE.MCNP` File

11.2 Other Data Output Files

have a less portable format which is fixed by non-user MARS routines which cannot be modified. a few files to be used by graphics packages and others (.GRA, .PLOT) and to be used in a consecutive run in a multi-stage case; the files are defined in the **??**

11.2.1 The `TRACK.PLOT` file

11.2.2 The `VERTEX.PLOT` file

12 Graphical-User Interface

A Graphical-User Interface, MARS-GUI-SLICE, has been developed. It is based on *Tcl/Tk* and is linked in to the user's executable, however it is active only when specific flags are set in the input file. When the interface is active, no events are generated, but the user's encoded geometry is probed and visually displayed. The interface displays all the details of the encoded geometry, showing the encoded zone numbers, materials and magnetic fields; it is a valuable tool for checking complex geometries before executing event generation. During event generation runs, the user can specify output files holding histograms and particle tracks; these output files can be opened by the *GUI* interface, post-run, and projected onto the visual display of the geometry.

12.1 Main Features

The main MARS-GUI-SLICE features are:

- Two-dimensional geometry slice and magnetic field view on a graphical display panel (GDP). Maximum and minimum coordinates along each axis and maximum field components are provided for the given view in corresponding entry fields (EF). They are changed automatically by grabbing a desirable view box on the GDP holding a *CTRL* key and clicking with the mouse left button at the two diagonal box corners. Alternatively, the lower and upper view boundaries can be typed in the EF along with a binning of the magnetic field grid seen on the GDP. There is a *1:1 scale* check field (CF) to return to a *natural* scale.

- A slice plane is chosen by a corresponding radio-button. A magnetic field view can be interactively turned ON and OFF in a corresponding CF.

- Materials distribution in a given view is represented in a color or black and white *wire-frame (contour) mode* or in a *color region-filled mode*, with the mode chosen by a corresponding radio-button. By clicking a corresponding button, a *Materials* window is created, with the CF displaying material index and name and select boxes (SB) showing color of each material in the given view on the GDP. The pre-set materials colors can arbitrary be modified in the corresponding SB individually for each material. The colors can be reset individually or globally. Changing the view automatically adjusts the material info in this window.

- By clicking a left mouse button at any point of the GDP, a *Point Info* window is created with information display fields (IDF) containing coordinates, region number, material name and index, magnetic field module and a value of histogram (see below) for this point. This window keeps the position intact.

- Particle tracks in the given view can be displayed on the GDP by loading a *.PLOT file generated by MARS. Similar to materials, by clicking a corresponding button, a *Particles* window is created, with information similar to the *Materials* window: particle ID, name, color and SB displaying color of each particle and allowing color modification. A corresponding CFs allow turning ON and OFF any ID and global track visibility. One can examine tracks by clicking a middle mouse button at any track point on the GDP. A *Track Info* window is created with IDF containing the particle ID, name, as well as the current energy, statistical weight and coordinates at the point.

- After the run, a variety of calculated 2-D histograms can be loaded and overlaped with the geometry view on the GDP. A *.HBOOK file is loaded in the *Load Hist* window and a desirable histogram is selected there from the IDF list by its ID or name. The geometry/histogram view is now handled as

a whole. The *Point Info* window allows now for a detailed examination of the histogram values even within the same decade (color).

- The view can be inverted both vertically and horizontally.
- One can add arbitrary texts all over the GDP with a *Text* window activated by the mouse right button. Variety of fonts can be chosen there. Fonts, subscripts and superscripts are handled as in the XMGR plotting tool [105]. Text can be modified, moved around the GDP or deleted.
- Up to 20 of the GDP views can be stored and restored back and then viewed by clicking $<<$ or $>>$ buttons.
- The GDP can be saved as an Encapsulated Postscript file by clicking the *Print* button. The entire window or its arbitrary fraction can be XV-grabbed by clicking the *Grab* button.
- A version exists for a 3-D solid body representation [106].

12.2 New Advanced Features

Substantial improvements have recently been made to the MARS-GUI-SLICE package. All the following features apply to the MARS15 graphics mode selected through `CTRL 1` in `MARS.INP`.

1. Selection of four GUI window sizes, respectively Extra-Large, Large, Medium, Small: XL (1000x840), L (700x700), M (580x640), S (480x540). The GUI pops up by clicking on any of the above buttons. No further change on the GUI window size is allowed further on. In order to change the window size the user must terminate the current work session and restart MARS15 in graphics mode.
2. The GUI panel has been compacted leaving some space for future developments. Resizing of the GUI window is inhibited.
3. Loading the geometry first by clicking on the $< Draw >$ button is now enforced by disabling both $< LoadHist >$ and $< LoadTrack >$ buttons that are grayed out prior to loading the geometry.
4. Either a histogram or the tracks can be loaded. In order to load the tracks, when a histogram is being displayed on the canvas, the user must click on the relative $< ON >$ toggle. Consequently the toggle turns to $< OFF >$ and the button $< LoadTrack >$ is enabled for selection. Similarly, in order to load a histogram, when the tracks are being displayed on the canvas, the user must click on the relative $< ON >$ toggle. Consequently the toggle turns to $< OFF >$ and the button $< LoadHist >$ is enabled for selection.
5. A button $< HistNorm >$ and relative entry have been added to the GUI panel. When a histogram is loaded the entry field shows the data normalization parameter (`AINT` or 1) applied by MARS15 when operated in simulation mode. The user has the possibility to renormalize on-line a histogram displayed in GUI, i.e., to vary the current histogram data normalization by typing a new factor in the entry field and then clicking on the ¡Draw¿ button. Useful to display results without a new Monte-Carlo run for another beam intensity or for another units, say, “mrem” instead of “mSv” (by increasing the normalization parameter by a factor of 100).
6. An arbitrary histogram value `HBK` can now be displayed in the $< PointInfo >$ widget. It was limited to the $2 \times 10^{-9} < HBK < 2 \times 10^{+9}$ in the old version.

7. A histogram data representation uniquely corresponds to a choice of the variable $< Number - Of - Decades >$. Acceptable values for such a variable are: 2, 4, 8, 16. The upper bound is 16. Upon loading a histogram the data are coded into 16 colors and 16 decades. The user can improve the resolution of the data representation by reducing the data range, hence changing the $< Number - Of - Decades >$ by entering the appropriate range end points in the relative entry field.

8. A $< Reset >$ button has conveniently been added to the GUI panel. If selected, all the current work session data are lost and the GUI is reset to its start-up state.

9. The MARS15 particle ID JJ has been added to the $< TracksPoint >$ widget.

10. The behavior of the histogram $< ON/OFF >$ toggle has been forced to be the same as the behavior of the tracks $< ON/OFF >$ toggle.

11. When a histogram is being displayed, the "aspect ratio" text string, the axes-symbol, and the maximum/minimum data values on either side of the color bar can be dragged all over the canvas by point&click using the mouse $< Button - 3 >$. They are also singularly deletable by point&click using the key stroke $< Ctrl >< Button - 3 >$.

12. The added $< ViewFormat >$ button allows for switching between the `1:1` and `16:9` canvas aspect ratios. The former view is intended for working purposes, whereas the latter one is useful for generating publications and documentations inserts. Some white space at the bottom of the canvas allows the user to include some add-on text strings. This new GUI state is fully integrated with the other GUI features.

13. The tracks environment has been integrated with the navigation system $<<$ and $>>$.

14. User's add-on text strings and arrows have been integrated with the navigation system $<<$, $>>$ and view format.

15. The geometry display modes have been reduced to two, namely $< Fill >$ and $< WireFrame >$. The former is currently the default at GUI start-up.

16. The horizontal `H` and vertical `V` rotation radio-buttons have been compacted. The rotation initial (default) values depend upon the slice/view selection according to Table 12.2. Remember, MARS15 uses the right-handed cartesian coordinate system. Therefore, the default settings shown in the table correspond to the top, side and cross-sectional (looking along the z-axis) views, respectively. The user can further vary the current slice rotation by acting directly on the `H` and `V` radio-buttons.

Table 13: Default GUI rotation settings.

Button	$Y - Z$	$X - Z$	X-Y
H	π	0	0
V	0	0	0

12.3 Installation and usage

This package requires to have `Tcl/Tk` installed. `Tcl/Tk` is scripting language and graphical user interface designer, developed by Dr. J. Ousterhout. It is free software and anyone can get copy from Scriptics web

site [107]. At least version 8.2 is required. Six variables in GNUmakefile might require changes in order to accommodate local installation:

- TCL_VER - `Tcl` version. Version 8.2 is known to work and it is the default value.
- TCL_INC - location of the `Tcl` include files.
- TCL_LOC - location of the `Tcl` include files.
- TK_VER - `Tk` version. Version 8.2 is known to work and it is the default value.
- TK_INC - location of the `Tk` include files.
- TK_LOC - location of the `Tk` library files.
- TCLTK_LST - list of `Tcl/Tk` library files. As default it consists of tcl and tk library -ltk$(TK_VER) -ltcl$(TCL_VER). Remember to put `Tk` library in linkage script before `Tcl` one.

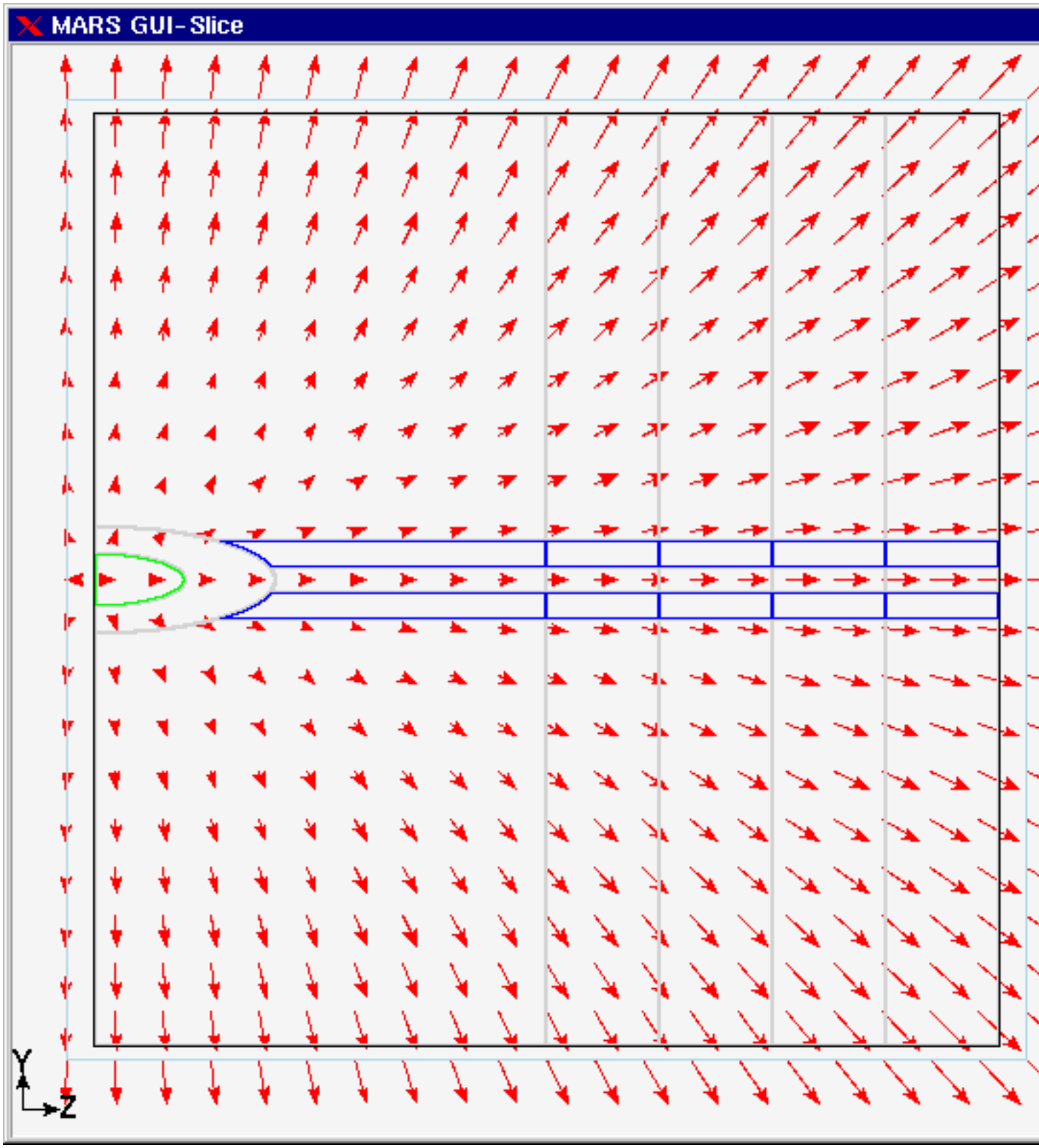
All interface is inside one `Tk/Tcl` script called `mf.tcl`, which should be stored into MARS data directory.

In order to run just set second control flag **CTRL** to 1 in the MARS.INP and run MARS executable - interface window will show up. Remember to set DISPLAY environment variable properly if you're working on remote computer via network.

12.4 Interface features

Interface main features:

- Draw magnetic field and two-dimensional geometry slice. Activated by pressing *Draw* button or *d* or *D* key. The picture of the main window is shown below



- The panel upper fields inform user about maximum field components and maximum field value

BH_max(T)	BV_max(T)	B_max(T)
20.600	36.050	41.521

- User can change the minimum and maximum values for each coordinate axis together with number of points used to draw magnetic field. It can be done using entries located on the right side of the interface. It is worth to mention also that

NBx	NBy	NBz
20	20	20
Xmin(cm)	Ymin(cm)	Zmin(cm)
-36.05	-36.05	-0.6
Xmax(cm)	Ymax(cm)	Zmax(cm)
36.05	36.05	20.6
◆ Y-Z	◇ X-Z	◇ X-Y

- User can set what axis slice will be activated. It can be done using radio buttons on the right side of the interface or pressing *x,X*, *y,Y* or *z,Z* keys respectively. Also the slice coordinate can be set using respective entry fields.

◆ Y-Z	◇ X-Z	◇ X-Y
X= 0.	Y= 0.	Z= 0.

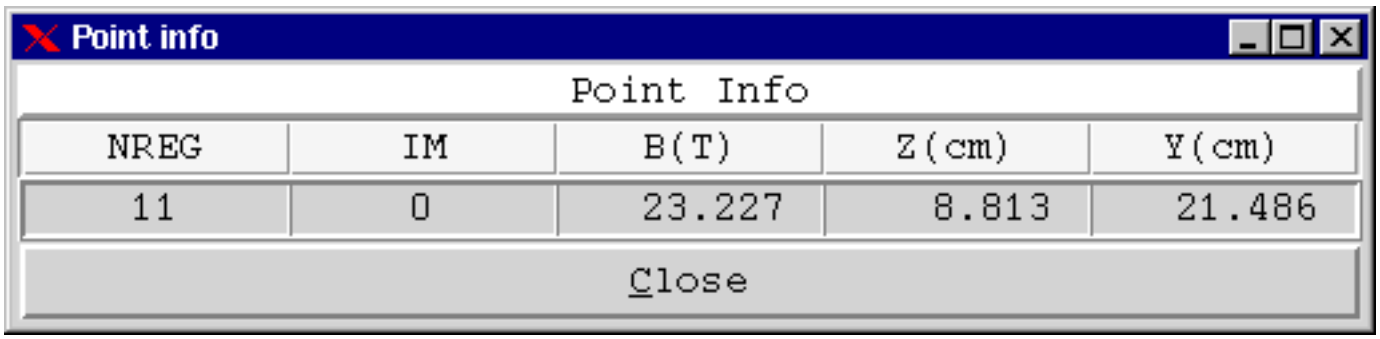
- The magnetic field could be interactively turned ON and OFF by user using the button.Magnetic field is ON by default

Magnetic field	ON

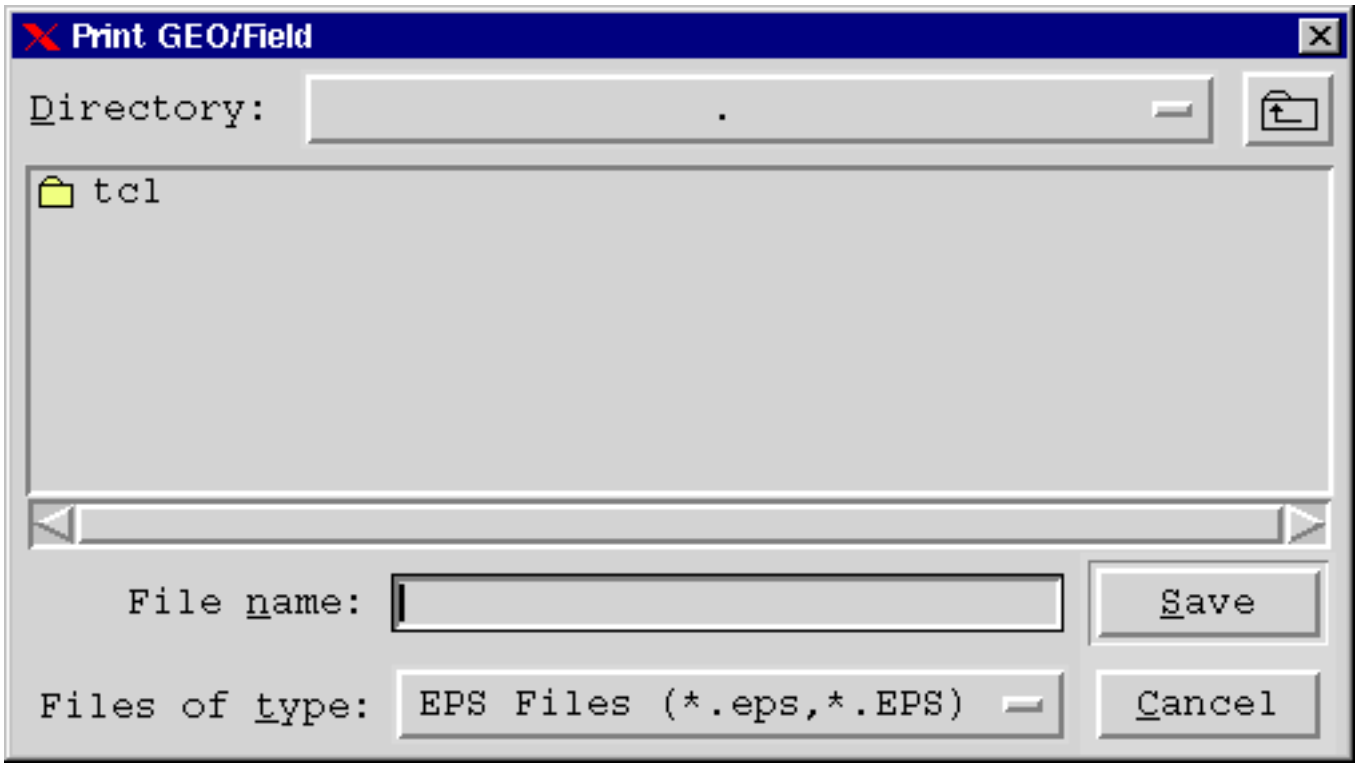
- Switch from so-called ”Natural” to separately scaled view. In ”Natural” mode program tries to preserve the same scale over horizontal and vertical axes. The horizontal axis counts as primary one,

therefore when button ”Natural” is pressed, vertical axis boundaries will be changed to accommodate changes in horizontal axis thus keeping the same overall scale. When ”Natural” is turned OFF, you can set horizontal and vertical boundaries separately. Natural view is OFF by default.

- Checking coordinates, region number, material index and magnetic field module just clicking left mouse button on field/geometry map. It keeps the position intact and updates

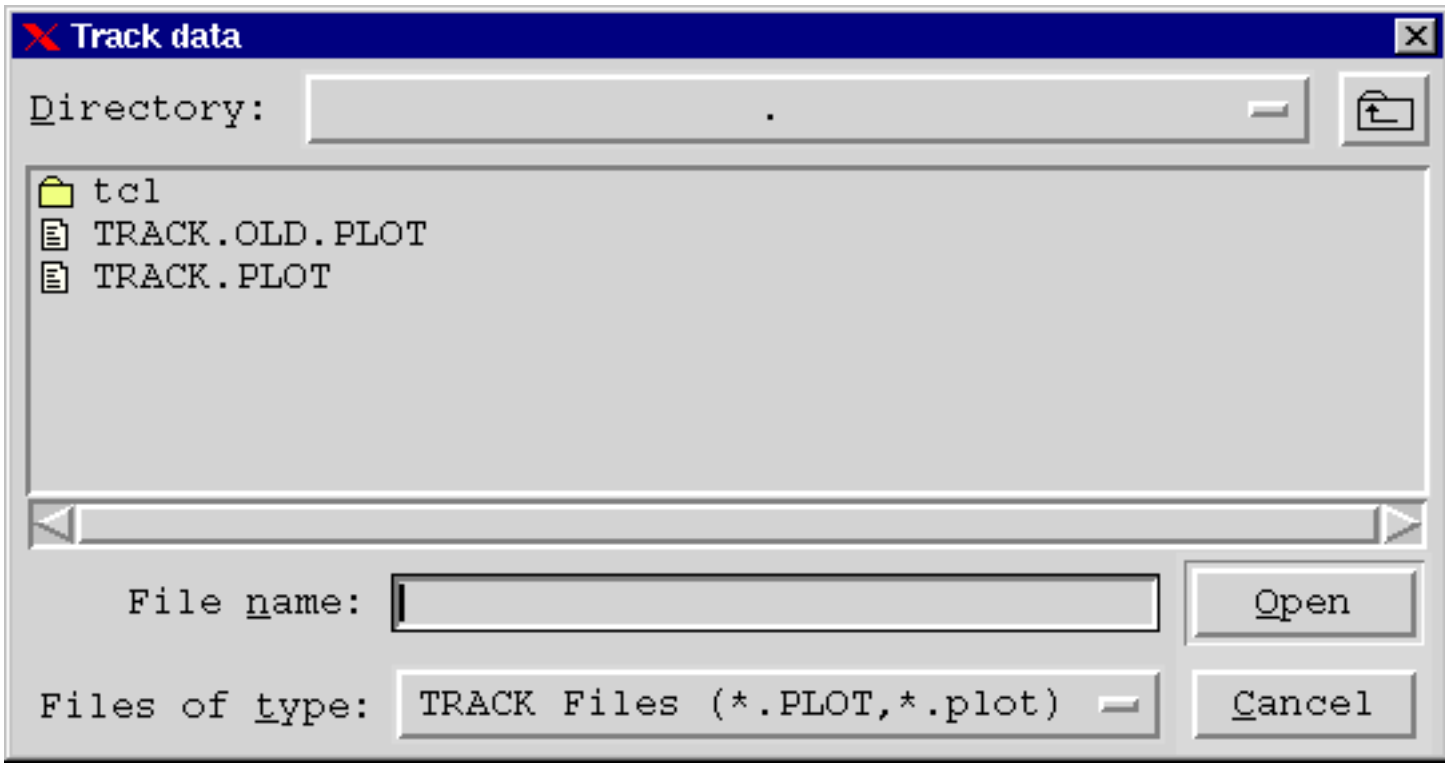


- Saving the drawing as well as ran external X grabber to create a picture as Encapsulated Postscript [108]. Saving the image as Encapsulated Postscript could be done by pressing *Print* button. User will get the dialog where he or she will be able to set EPS file name. You also can start external image grabber (XV by default) just pressing *Grab* button.



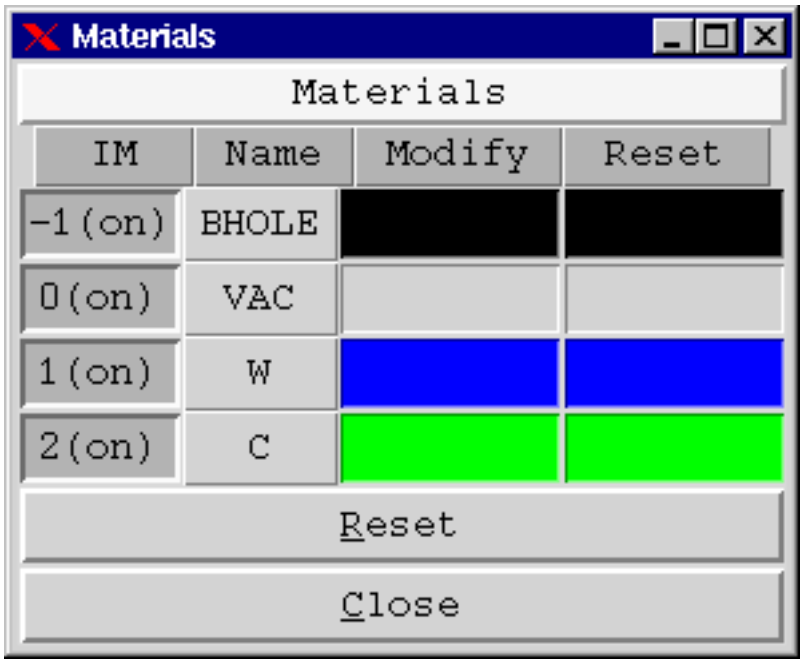
- User is able to load particle tracks as produced by MARS. Stored tracks file usually has **.PLOT** extension. Also there is possibility to turn on or off the tracks visibility. All of that can be done in main panel using "Load Track" button and switch.

The same as "Print" dialog will appear with prompt to enter track file name



- There are two information panels can be started - one for materials and another for particles.

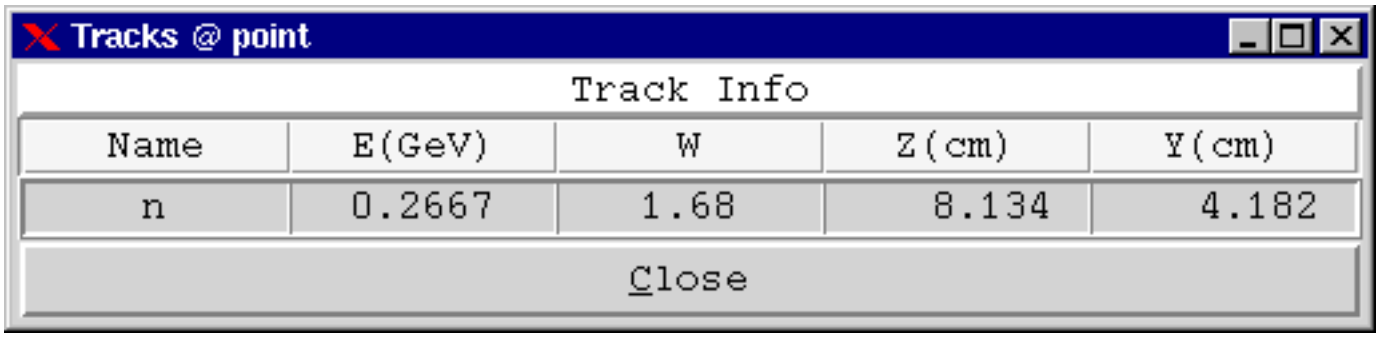
- Materials info panel allows you to examine which material is drawn in which color, turn ON and OFF interactively what material to show. It can be done by pressing the buttons on the panel left. The individual material color can be altered by pressing the colored square on the panel right.



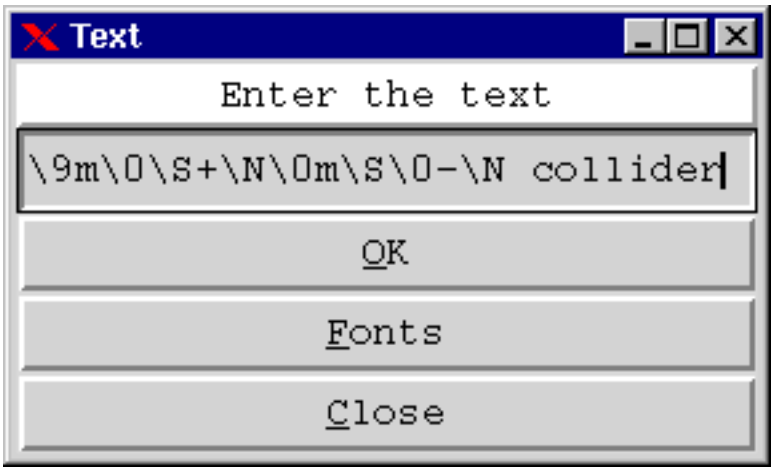
Of course, if you change the view and number of materials will be changed, the panel will adjust automatically

- Particles panel has the same properties and allows user to pick particular particles to show and change tracks color interactively

- User can examine tracks by pointing mouse and pressing middle button. panel with track information will appear and particle type, energy, weight and position will be shown.



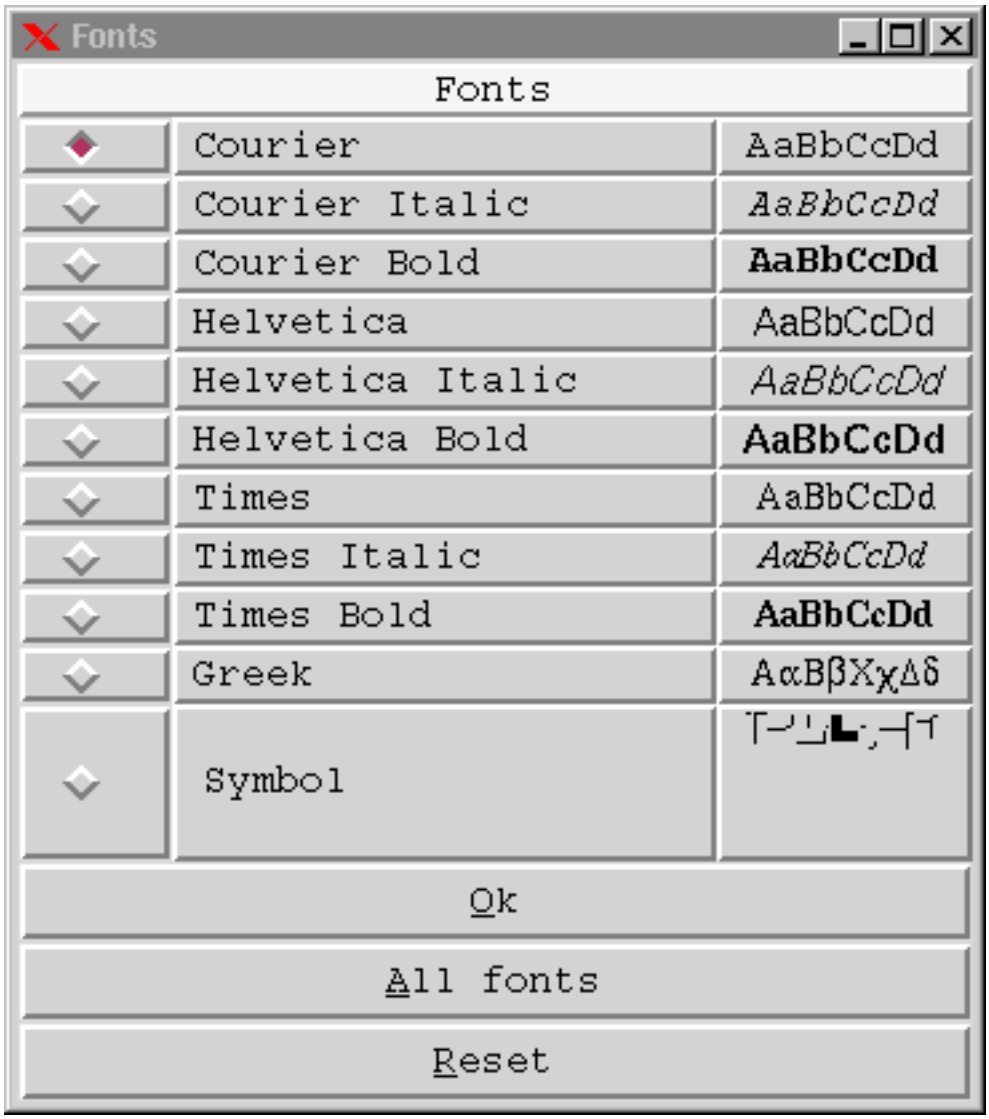
- User can add texts all over the picture. One can do it by holding Shift key and pressing mouse right button simultaneously. Text will be drawn where the mouse pointer is positioned at this moment. Dialog window will appear where text can be entered



The same convention as in well known XMGR drawing tool [105] is used. Namely, \n will switch you to font number n, \S will switch text to superscript mode, \s will switch text to subscript mode and \N will return text back to normal position. Example above will produce the next text

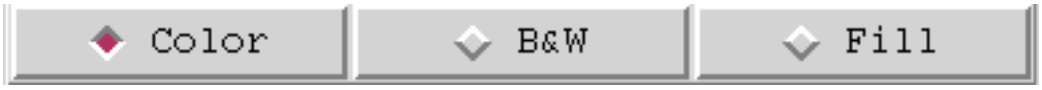
$\mu^+ \mu^-$ collider

- You also can choose fonts interactively by pressing Fonts button. Dialog will occur which will show you what fonts are available. Font numeration in the dialog is exactly the same used in text entry field



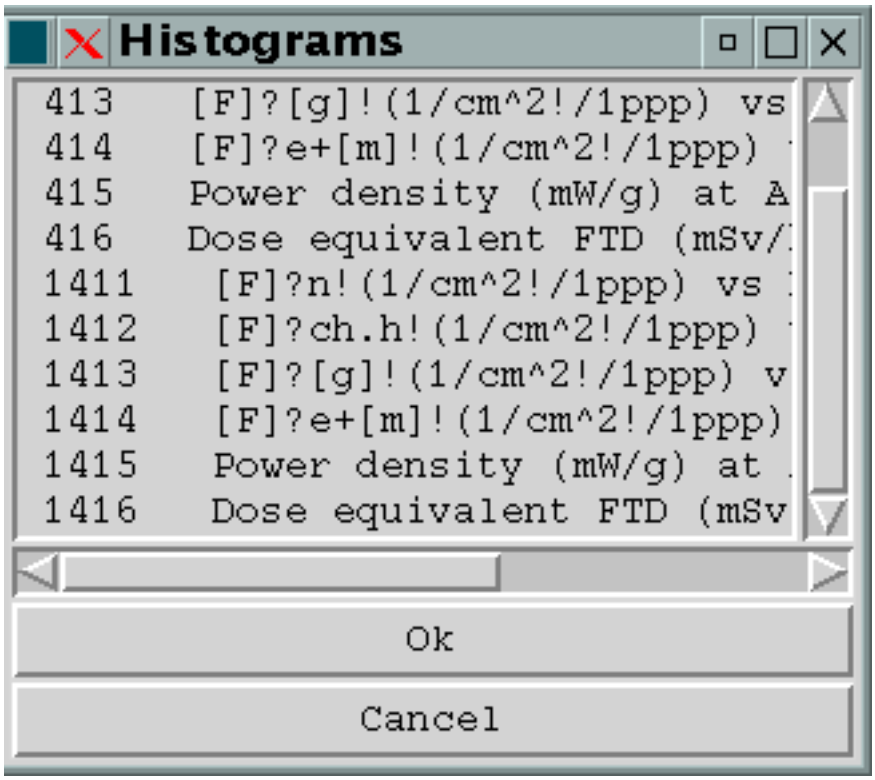
- Texts can be moved around the picture by grabbing them with third mouse button and dragging them around. User can delete texts by holding Ctrl key and pressing third mouse button on text fragment to be deleted.
- Ability to zoom in part of the picture using mouse. Just hold `Crtl` key while marking the opposite corner of zooming box and then redraw the geometry.
- Possibility to view the inverted image. You can use the following buttons to set either horizontal or vertical inversion and then redraw the geometry.

- The ability to draw the geometry with filled or transparence colors, or even in colorless (black&white) mode using next radiobuttons panel.



- The possibility to overlap the geometry view and histogram view. User could open histogram after MARS batch runs and draw the result of calculations on top of geometry

- Note, that it is user responsibility to make picture and histogram coherent. One can open .hbook file, check the histogram list and choose what histogram to load.



- The possibility to store and restore back number of previous geometry views and then move between them using one mouse click. That can be done using next buttons

13 Additional Code Topics

13.1 Rules-Of-Thumb

It is obvious that the quality of the typical output of any code, for given physics model, depends on how the user built the calculational model and how he/she handled the code. The geometry description is of primary importance. Then, as stated above, the most essential parameters to control the calculational accuracy for given physics model are the number of incidents (primary events) `NEVT` and the inclusive/exclusive switches. Some other parameters also affect the computing efficiency: the accuracy of boundary localization in iterative transport algorithm `STEPEM`, `IND(6)` option, cutoff energies, geometry details, histogramming etc. Naturally, the higher `NEVT`, the better the result will be. But here we come to a contradiction with both CPU time t_1 and time t_2 alotted for the whole problem. The strategy would be to keep both t_1 and t_2 as small as possible. There are a few rules for a user to get the best from the MARS code. These rules are:

1. Required `NEVT` is determined by a statistical error in a phase-space or geometrical region of interest. The calculated results in a given region `N` are statistically valid only if a R.M.S. statistical error $\delta \leq$20%. So, run until this condition is satisfied. Do short runs first to estimate required `NEVT` and play with DUMP output.

2. Before a long run, try to understand what a combination of inclusive/exclusive options provides highest computing efficiency.

3. Use as few as possible geometrical regions described in all the $standard$, $extended$ and $user-supplied$ sectors: CPU time grows almost linearly with the number of regions in the direction of predominant propagation of the particles.

4. Use as little as possible volume detector histogramming: it is rather time-consuming. At least minimize the number of bins.

5. Keep `STEPEM`$\simeq$`0.1`$\times t_{min}$, where t_{min} is a smallest linear size of the smallest region in the considered group of zones.

6. Use cutoff energies for each particle class as high as possible (to not damage result, of course), especially in bulk regions far from your regions of interest.

7. Use `IND(15)=T` for thick shielding calculations.

8. Use `IND(6)=T` and `DLEXP`$\neq$`1` options for thick shielding calculations. Be carefull as with any biasing techniques, do short tests first.

9. Use `IND(1)=F` in a routine run to reduce amount of the output.

In some of these rules, the code takes care of those components in some effective manner anyway, but the user can reduce the CPU time drastically if he/she turns off the corresponding options.

13.2 Biasing and Other Control of Physics Processes

This is a new section, intended to describe in detail how to control or modify the physics processes applied to various particle types as they are transported; for example what thresholds or other input deck settings modify which physics processes.

13.3 Supplementary Routines

Section on supplementary routines available from the MARS libraries: ATNM, RM48, RNDM, POISSN, NORRAN, NORMCO, FINEXT, EINT, FINT2D, SGINT1, ISOTR,

13.4 Multiprocessing

A parallel processing option has been developed and implemented into MARS15 [13]. It is based on the Message Passing Interface (MPI) libraries. Parallelization is job-based, i.e. the processes, replicating the same geometry of the setup studied, run independently with different initial seeds. A unique master process – also running event histories – collects intermediate results from an arbitrary number of slaves and calculates the final results when a required total number of events has been processed. Intermediate results are sent to the master on its request generated in accordance with a scheduling mechanism. The algorithm was tested on Unix and Linux clusters and demonstrated good performance.

13.5 External Packages Required

13.6 Interfaces to Other Programs

13.6.1 STRUCT

13.6.2 ANSYS

14 Additional Information

14.1 MARS Web Page and World-Wide Support

MARS is under active development, with new versions released every year or so. Registered users will be sent email announcing new releases, with a list of the features which have been implemented. Many examples with corresponding write-ups are posted on the site. A question-and-answer sections holds user questions with expert response.

14.2 Benchmarking of the Simulation

Outline the bench-marking to data process. Refer to Data-Simulation comparisons in published references.

14.3 Future Developments

Outline on-going developments

15 Acknowledgements

We express our gratitude to P. Aarnio, Y.I. Eidelman, K.K. Gudima, M.A. Kostin, O.E. Krivosheev, S.G. Mashnik, M.E. Monville, I.L. Rakhno, A.J. Sierk, and S.I. Striganov for their contribution to this version of the MARS code.

References

[1] N. V. Mokhov, Mars15 overview, Recent Mars15 developments: nuclide inventory, DPA and gas production, Technical Report Fermilab-Conf-10/518-APC, 2010.

[2] P. Aarnio, Decay and transmutation of nuclides, Technical Report CMS-NOTE-1998/086, CERN, 1998.

[3] A. Isotalo, Modifications to DeTra, Technical Report September 18, 2008.

[4] N. V. Mokhov, I. L. Rakhno and S. I. Striganov, Simulation and verification of DPA in materials, in *Applications of High Intensity Proton Accelerators, World Scientific Proc. pp. 128-131*, 2010.

[5] I. L. Rakhno, Modeling heavy ion ionization energy loss at low and intermediate energies, Technical Report Fermilab-FN-0835-APC, 2009.

[6] N. V. Mokhov and S. I. Striganov, Mars15 overview, Technical Report Fermilab-Conf-07/008-AD, 2007.

[7] N. V. Mokhov et al., Physics models in the mars15 code for accelerator and space applications, in *Int. Conf. on Nuclear Data for Science and Technology, AIP Conf. Proc. 769, pp. 1618-1623*, 2004.

[8] S. G. Mashnik, K. K. Gudima, A. J. Sierk, M. I. Baznat, and N. V. Mokhov, Cem03.01 user manual, Technical Report LANL LA-UR-05-7321, 2005.

[9] N. V. Mokhov, E. I. Rakhno, and I. L. Rakhno, Residual activation of thin accelerator components, Technical Report Fermilab-FN-0788-AD, 2006.

[10] I. L. Rakhno, N. V. Mokhov, and S. I. Striganov, Modeling heavy ion ionization loss in the mars15 code, Technical Report Fermilab-Conf-05/019-AD, 2005.

[11] N. V. Mokhov et al., Recent enhancements to the mars15 code, Technical Report Fermilab-Conf-04/053, 2004.

[12] N. V. Mokhov, K. K. Gudima, S. G. Mashnik, I. L. Rakhno, and S. I. Striganov, Towards a heavy-ion transport capability in the mars15 code, Technical Report Fermilab-Conf-04/052, 2004.

[13] M. A. Kostin and N. V. Mokhov, Parallelizing the mars15 code with mpi for shielding applications, Technical Report Fermilab-Conf-04/054, 2004.

[14] M. A. Kostin, O. E. Krivosheev, N. V. Mokhov, and I. S. Tropin, An improved mad-mars beam line builder: User's guide, Technical Report Fermilab-FN-738-rev, 2004.

[15] S. I. Striganov, On the theory and simulation of multiple coulomb scattering of heavy charged particles, Technical Report Fermilab-Conf-04/056, 2004.

[16] N. V. Mokhov, Nauka, in *Proc. IV All-Union Conference on Charged Particle Accelerators*, volume 2, page 222, Moscow, 1975.

[17] N. V. Mokhov and V. V. Phrolov, Sov. J. Atomic Energy **38**, 226 (1975).

[18] R. P. Feynman, Phys. Rev. Lett. **23**, 1415 (1969).

[19] A. V. Ginneken, Weighted monte carlo calculations in thick targets, Technical Report FN-250, Fermilab, 1972.

[20] A. V. Ginneken, Casim – program to simulate transport of hadronic cascades in bulk matter, Technical Report FN-272, Fermilab, 1975.

[21] W. R. Nelson, H. Hirayama, and D. Rogers, The egs4 code system, Technical Report SLAC-265, Stanford Linear Accelerator, 1985.

[22] A. Fasso, A. Ferrari, J. Ranft, and P. Sala, Fluka92, in *Proceedings of the SARE Workshop, Santa Fe, New Mexico, January 1993*, volume A349, Nucl. Instruments and Methods, 1994.

[23] A. V. Ginneken, Calculation of the average properties of hadronic cascades at high energies (casim), in *Computer Techniques in Radiation Transport and Dosimetry*, edited by W. R. Nelson and T. M. Jenkins, 1978.

[24] N. V. Mokhov, Sov. J. Part. Nucl. **18**, 408 (1987).

[25] N. V. Mokhov, Inclusive simulation of hadronic and electromagnetic cascades in the ssc components, Technical Report SSC-SR-1033, SSC Central Design Group, 1988.

[26] A. N. Kalinovsky, N. V. Mokhov, and Y. P. Nikitin, *Passage of High-Energy Particles through Matter*, AIP, New York, 1989.

[27] N. V. Mokhov, S. I. Striganov, and A. V. Uzunian, Technical Report 87-59, IHEP, 1987.

[28] I. L. Azhgirey, N. V. Mokhov, and S. I. Striganov, Antiproton production for tevatron, Technical Report TM-1730, Fermilab, 1991.

[29] N. V. Mokhov, The mars12 code system, in *Proceedings of the SARE Workshop*, Santa Fe, New Mexico, 1993.

[30] S. L. Kuchinin, N. V. Mokhov, and Y. N. Rastsvetalov, Technical Report 75-74, IHEP, Serpukhov, 1975.

[31] I. S. Baishev, S. L. Kuchinin, and N. V. Mokhov, Technical Report 78-2, IHEP, Serpukhov, 1977.

[32] M. A. Maslov and N. V. Mokhov, Particle Accelerators **11**, 91 (1980).

[33] N. V. Mokhov, Energy deposition in targets and beam dumps at 0.1-5 tev proton energy, Technical Report FN-328, Fermilab, 1980.

[34] N. V. Mokhov, Technical Report 82-168, IHEP, Serpukhov, 1982.

[35] N. V. Mokhov, The mars10 code system: Inclusive simulation of hadronic and electromagnetic cascades and muon transport, Technical Report FN-509, Fermilab, 1989.

[36] N. V. Mokhov and J. D. Cossairt, Nucl. Instruments and Methods **A244**, 349 (1986).

[37] I. S. Baishev, I. A. Kurochkin, and N. V. Mokhov, Technical Report 91-118, IHEP, Protvino, 1991.

[38] I. L. Azhgirey, I. A. Kurochkin, M. A. Maslov, V. V. Talanov, and A. V. Uzunian, Technical Report 93-19, IHEP, Protvino, 1993.

[39] D. C. Wilson, C. A. Wingate, J. C. Goldstein, R. P. Godwin, and N. V. Mokhov, Hydrodynamic calculations of 20-tev beam interactions with the ssc beam dump, in *Proceedings of the 1993 Particle Accelerator Conference, IEEE*, pages 3090–3092, 1993.

[40] J. Ftacnik and M. Popovic, Parallel version of mars10 and mars12 codes, in *The 1994 April APS Meeting*, 1994.

[41] O. E. Krivosheev and N. V. Mokhov, A new mars and its applications, 1998.

[42] N. V. Mokhov et al., Mars code developments, 1998.

[43] N. V. Mokhov, Mars code developments, benchmarking and applications, in *Proc. of ICRS-9 International Conference on Radiation Shielding, October 17-22, 1999*, volume 1, pages 167–171, Tsukuba, Ibaraki, Japan, 2000, J. Nucl. Sci. Tech.

[44] N. V. Mokhov and A. V. Ginneken, in *Proc. of ICRS-9 International Conference on Radiation Shielding, October 17-22, 1999*, volume 1, pages 172–179, Tsukuba, Ibaraki, Japan, 2000, J. Nucl. Sci. Tech.

[45] N. V. Mokhov and O. E. Krivosheev, Mars code status, 2000.

[46] O. E. Krivosheev and N. V. Mokhov, Status of mars electromagnetic physics, 2000.

[47] N. V. Mokhov, S. I. Striganov, and A. V. Ginneken, Muons and neutrinos at high-energy accelerators, 2000.

[48] I. Baishev, A. Drozhdin, and N. Mokhov, Struct program user's reference manual, Technical Report SSCL-MAN-0034, SSC Laboratory, 1994.

[49]

[50]

[51]

[52] P. D. Group, Phys. Rev. (1998).

[53]

[54] V. S. Barashenkov, Particle and nucleus cross-sections, Technical report, Dubna, 1993.

[55]

[56]

[57]

[58] N. V. Mokhov and S. I. Striganov, Beam dynamics and technology issues for $\mu^+\mu^-$ colliders, in *AIP Conference Proceedings*, page 234, Montauk, NY, 1995.

[59]

[60]

[61]

[62] N. V. Mokhov and S. I. Striganov, Model for pion production in proton-nucleus interactions, 1998.

[63]

[64] N. V. Mokhov and A. V. Ginneken, to be published.

[65]

[66]

[67]

[68] J. Ranft, Phys. Rev. **D51** (1995).

[69] J. Ranft, Gran sasso report, Technical Report AE-97/45, INFN, 1997.

[70] N. V. Mokhov, Muons at lhc. part 1: Beam dumps, Technical Report TIS–RP/TM/95–27, CERN, 1995.

[71] Y. Tsai, Rev. Mod. Phys. **46**, 815 (1995).

[72] N. V. Mokhov, G. I. Semenova, and A. V. Uzunian, Nucl. Instruments and Methods **180**, 469 (1981).

[73] M. A. Maslov, N. V. Mokhov, and A. V. Uzunian, Nucl. Instruments and Methods **217**, 419 (1983).

[74] S. I. Striganov, Technical Report 94-14, IHEP, 1994.

[75] G. W. Foster and N. V. Mokhov, Backgrounds and detector performance at a 2×2 tev $\mu^+\mu^-$ collider, in *Proceedings of the 2nd Workshop on Physics Potential and Development of $\mu^+\mu^-$ Colliders, Sausalito, California, November 17-19, 1994*, AIP, 1995.

[76] P. D. Group, Phys. Rev. **D50** (1994).

[77]

[78]

[79] I. S. Baishev, N. V. Mokhov, and S. I. Striganov, Sov. J. Nucl. Physics **42**, 1175 (1985).

[80]

[81]

[82] J. Briesmeister, Mcnp - a general monte carlo n-particle transport code, Technical Report LA-12625-M, LANL, 1993.

[83] N. V. M. M. Huhtinen, A cross-comparison of mars and fluka simulation codes, Technical Report FN-697, Fermilab, 2000.

[84]

[85]

[86] I. S. Baishev, M. A. Maslov, and N. V. Mokhov, in *Proc. VIII All-Union Conference on Charged Particle Accelerators*, volume 2, page 167, Dubna, 1983.

[87] M. A. Maslov and N. V. Mokhov, Technical Report IHEP 85-8, Serpukhov, 1985.

[88] N. V. Mokhov, Nucl. Phys. B (Proc. Suppl.) **51A**, 210 (1996).

[89] C. Johnstone and N. Mokhov, Optimization of a muon collider interaction region with respect to detector backgrounds and the heat load to the cryogenic systems, Technical Report Conf–96/366, Fermilab, 1996.

[90] A. Drozhdin, M. Huhtinen, and N. Mokhov, Nucl. Instruments and Methods **A381**, 531 (1996).

[91] A. I. Drozhdin, M. Harrison, and N. V. Mokhov, Study of beam losses during fast extraction of 800 gev protons from the tevatron, Technical Report FN-418, Fermilab, 1985.

[92] I. S. Baishev, A. I. Drozhdin, and N. V. Mokhov, Beam loss and radiation effects in the ssc lattice elements, Technical Report SSCL-306, SSC Laboratory, 1990.

[93] A. Drozhdin, N. Mokhov, R. Soundranayagam, and J. Tompkins, Toward design of the collider beam collimation system, Technical Report SSCL-Preprint-555, SSC Laboratory, 1994.

[94] A. Drozhdin, N. Mokhov, and B. Parker, Accidental beam loss in superconducting accelerators: Simulations, consequences of accidents and protective measures, Technical Report SSCL-Preprint-556, SSC Laboratory, 1994.

[95] J. M. Butler et al., Reduction of tevatron and main ring induced backgrounds in the d0 collision hall, Technical Report FN-629, Fermilab, 1995.

[96] IEEE, *IEEE Standard 754-1985 for Binary Floating-Point Arithmetic*, 1985.

[97] G. Marsaglia and A. Zaman, Toward a universal random number generator, Technical Report FSU-SCRI-87, Florida State University, 1988.

[98] I. S. Baishev, Technical Report IHEP 87-149, Serpukhov, 1987.

[99]

[100]

[101] F. Iselin, The MAD program (methodical accelerator design), version 8/13/8, Technical Report SL/92, CERN, 1992.

[102] D. M. et al, MAD parsing and conversion code, Technical Report TM-2115, Fermilab, 2000.

[103] R. Brun, O. Couet, C. Vandoni, and P. Zanarini, Paw physics analysis workstation, Technical Report Q121, CERN Program Library.

[104] R. Brun and D. Lienart, *HBOOK User Guide*, CERN Program Library, y250 edition.

[105]

[106] O. E. Krivosheev and N. V. Mokhov, Oo geometry engine for monte-carlo simulation, in *Proc. of Radiation Protection and Shielding Topical Meeting, No. Falmouths, MA, April 21–25*, pages 487–493, 1996.

[107]

[108]